智能系统学报
CAAI TRANSACTIONS ON INTELLIGENT SYSTEMS

空间关键字个性化语义近似查询方法

李盼，张霄雁，孟祥福，赵路路，齐雪月

引用本文:
李盼, 张霄雁, 孟祥福, 等. 空间关键字个性化语义近似查询方法[J]. 智能系统学报, 2020, 15(6): 1163–1174.
LI Pan, ZHANG Xiaoyan, MENG Xiangfu, et al. Spatial keyword personalized and semantic approximate query approach[J]. *CAAI Transactions on Intelligent Systems*, 2020, 15(6): 1163–1174.

在线阅读 View online: https://dx.doi.org/10.11992/tis.201903033

DOI: 10.11992/tis.201903033

空间关键字个性化语义近似查询方法

李盼, 张霄雁, 孟祥福, 赵路路, 齐雪月
(辽宁工程技术大学 电子与信息工程学院, 辽宁 葫芦岛 125105)

摘 要: 现有的空间关键字查询处理模式大都仅支持位置相近和文本相似匹配, 但不能将语义相近但形式上不匹配的对象提供给用户; 并且, 当前的空间−文本索引结构也不能对空间对象中的数值属性进行处理。针对上述问题, 本文提出了一种支持语义近似查询的空间关键字查询方法。首先, 利用词嵌入技术对用户原始查询进行扩展, 生成一系列与原始查询关键字语义相关的查询关键字; 然后, 提出了一种能够同时支持文本和语义匹配, 并利用 Skyline 方法对数值属性进行处理的混合索引结构 AIR-Tree; 最后, 利用 AIR-Tree 进行查询匹配, 返回 top-k 个与查询条件最为相关的有序空间对象。实验分析和结果表明, 与现有同类方法相比, 本文方法具有较高的执行效率和较好的用户满意度; 基于 AIR-Tree 索引的查询效率较 IRS-Tree 索引提高了 3.6%, 在查询结果准确率上较 IR-Tree 和 IRS-Tree 索引分别提高了 10.14% 和 16.15%。

关键词: 空间关键字查询; 词嵌入; 语义近似查询; 文本; 数值属性; 索引结构; 查询匹配
中图分类号: TP311 **文献标志码:** A **文章编号:** 1673−4785(2020)06−1163−12

中文引用格式: 李盼, 张霄雁, 孟祥福, 等. 空间关键字个性化语义近似查询方法 [J]. 智能系统学报, 2020, 15(6): 1163–1174.
英文引用格式: LI Pan, ZHANG Xiaoyan, MENG Xiangfu, et al. Spatial keyword personalized and semantic approximate query approach[J]. CAAI transactions on intelligent systems, 2020, 15(6): 1163–1174.

Spatial keyword personalized and semantic approximate query approach

LI Pan, ZHANG Xiaoyan, MENG Xiangfu, ZHAO Lulu, QI Xueyue
(School of Electronic and Information Engineering, Liaoning Technical University, Huludao 125105, China)

Abstract: Most spatial keyword query processing models only support the location proximity and text similarity matching. However, in terms of text information processing, spatial objects with similar semantics but mismatched forms cannot be filtered out and provided to query users. Furthermore, the current spatial-text index structure cannot process the numerical attributes. To solve the above problem, this paper proposes a spatial keyword query method that can support the semantic approximate query processing. Word embedding technology is used to expand the users' original queries and generate a series of query keywords semantically related to the original query keywords. Then, a hybrid index structure AIR-tree that can support text and semantic matching and use the Skyline method to process numerical attributes is proposed. Finally, AIR-tree is used for query matching to return the top-k ordered spatial objects most closely related to the query conditions. Experimental analysis and results show that compared with similar methods, this method has a higher execution efficiency and better user satisfaction. The query efficiency based on the AIR-tree index is 3.6% higher than that of the IRS-tree index. In terms of accuracy, IR-tree and IRS-tree are increased by 10.14% and 16.15%, respectively, compared with AIR-tree.

Keywords: spatial keyword query; word embedding; semantic approximate query; text; numerical attribute; index structure; query matching

移动网络的普遍应用和空间 Web 对象的大量出现, 使得空间关键字查询成为 LBS(location-based system) 的重要支撑技术。现有的空间关键字查询处理模式主要有 top-k 范围查询 (top-k range query) 和 top-k k 近邻查询 (top-k kNN query), 这两类查询处理模式主要是根据空间对象与空间关键字查询之间的文本相似度和位置相近度构建结果评分函数, 进而利用文本和空间混合索引技术提高查询效率。现有的空间数据和文本信息相混合的空间−文本索引技术主要有 IR-Tree[1]、IR^2-Tree[2]、QuadTree[3]、R*-Tree[4]、S2I[5] 等; 文本搜索

收稿日期: 2019−03−25.
基金项目: 国家自然科学基金面上项目 (61772249).
通信作者: 孟祥福. E-mail: marxi@126.com.

的索引技术主要有倒排文件 (inverted file)、签名文件 (signature file) 和位图索引 (bitmap) 等。上述文本索引以及空间−文本索引主要适用于查询关键字的严格形式匹配，但由于现实中文本表达形式多样，如果采用关键字严格形式匹配，可能导致返回的查询结果太少或没有结果。针对上述问题，相关研究如文献 [6-8] 中提出了一些新的索引技术以处理文本中的拼写错误，然而这些方法并没有考虑文本之间的语义相似/相关度。尽管最近有少数工作研究了空间关键字查询的语义匹配[9]，但空间对象除包含位置信息和文本信息外，还包含了价格、用户评分等数值数据。目前还没有相关工作同时考虑空间对象与空间关键字查询在位置、文本、语义和数值上的综合相关度，进而也就没有同时支持上述综合查询的混合索引结构。

针对上述问题，本文工作的重点是建立一种同时融合位置信息、文本信息、语义信息和数值信息的空间关键字查询处理模式，并提出一种有效的混合索引结构来提高查询效率。

本文的主要贡献如下：

1) 提出了基于 Word Embedding 技术对初始查询关键字进行语义扩展的方法，能够为用户提供语义相关的近似查询结果；

2) 在文本信息匹配方面，考虑了查询关键字可能会出现拼写错误情况，提出了基于编辑距离的字符串相似度度量方法，尽量避免因查询关键字拼写错误而导致没有匹配结果的情况；

3) 提出了基于 Skyline 的数值属性处理方法，使得空间关键字查询处理模式能够处理数值属性，令查寻结果更能满足用户的个性化需求。

4) 构建了一种新型的混合索引结构 AIR-Tree，该索引结构能够直接从每个节点中获取该节点对应的数值属性的 Skyline 集合，并能同时对位置信息、文本信息和语义信息进行索引。

1　相关工作

随着移动网络的普遍应用，网络上出现了越来越多的空间 Web 对象 (spatial web object)。一个空间对象通常包含位置信息 (如经纬度)、文本信息 (如空间对象的名称、设施、类别等) 以及数值信息 (如价格、用户评分等)，数值信息有时也归为文本信息。现有工作将数值信息作为文本关键字来进行处理，但实际上数值信息的处理方法与文本信息匹配的处理方法有本质区别 (如文本信息的主要处理方法是统计和字符串匹配，而数值信息可以进行数值大小比较、数值计算等操作)。

现有的空间关键字查询处理模式主要可以分为 4 类：布尔范围查询、布尔 k 近邻 (kNN) 查询、top-k 范围查询以及 top-k k 近邻查询。上述四类方法呈递进式发展，后者是对前者的改进。布尔范围查询的缺点是不能控制查询结果规模，且没有对查询结果进行排序；布尔 k 近邻查询是通过兴趣点与查询点之间的距离对查询结果排序，排序前后与距离大小成反比。布尔范围查询和布尔 k 近邻查询方法都需要兴趣点的文本描述中包含所有查询关键字，这很可能导致查询不到结果或只有少量查询结果，或是得到的查询结果距离查询点的位置很远。top-k 范围查询和 top-k 近邻查询模式不要求兴趣点的描述信息包含所有的查询关键字，查询结果也可以是仅包含部分关键字的查询结果。然而，top-k 范围查询的排序方法仅考虑了兴趣点的文本相关性而没有考虑位置相近性，top-k k 近邻查询同时考虑了兴趣点与查询的位置相近性和文本相关性，但没考虑语义相关性。

空间关键字查询[10-11] 通常需要将空间索引和文本索引相结合起来构建混合索引结构，从而达到高效地检索空间对象的目的。当前主要的空间数据混合索引结构可归结为表 1 所示的几类。

表 1　混合索引结构

Table 1　Hybrid index structure

索引名称	组合模式	优点	缺点
两阶段索引[12]	R-tree, Inverted file	结构简单	存储代价高，无法确定第一阶段产生的候选对象个数
IR^2-tree	R-tree+Signature	存储代价低、搜索效率高	查询关键字必须严格匹配
IR-tree	R-tree+Inverted file	存储空间小，提高了搜索效率，允许查询关键字部分匹配	未考虑查询的语义相关性
bR*-tree[13]	R*-tree+Bitmap	存储空间小，关键字匹配效率高	关键字多，I/O代价高
Light-Weighted索引[14]	R*-tree和Inverted file分开存储	可扩展性较强，搜索效率高	存储代价高，频繁插入操作的计算代价过高
Quadtree索引	Quadtree+Inverted file	区域搜索效率高	树结构不平衡，存储代价较高
G-index索引[15]	聚类标准+聚类操作	通用性强	存储代价高，泛化计算代价高

近年来, AirBnB、TripAdvisor、hotels.com、Craigslist、Yelp 以及 Zillow 等 LBS 系统, 都存在布尔或分类属性, 也包括大量数值属性。但是在大多数情况下, 这些数值属性一般被离散化和转换为分类属性[16], 然而这样的处理并不能满足用户的查询需求。Liu 等[17] 提出的 IRS-Tree 是一种拥有 Synopses 的倒排 R-Tree 的混合索引结构, 能够有效处理一组位置敏感等级查询。然而, 基于 IRS-Tree 的查询算法要求提供数值属性的精确范围, 故数值属性的精确匹配也可能会导致过少甚至没有查询结果返回。针对 IRS-Tree 存在的问题, 本文利用 Skyline 方法[18], 对兴趣点的数值属性进行处理, 将处于被支配地位的元组从 Skyline 中删除, 使查询结果更满足用户的个性化需求。

文本之间的语义相关性度量方法可大致分为 3 类: 第一类是通过使用本体来定义术语/概念之间的距离, 进而定义拓扑相似性估计语义相似性; 第二类是使用如向量空间模型等统计手段估计语言单元 (例如单词、句子) 之间的语义相关性; 第三类是采用概率主题模型对文本信息进行语义近似处理。词嵌入方法是近年来自然语言理解领域出现的新方法, 该方法是自然语言处理 (NLP) 中的一组语言建模和特征学习技术的集合名称, 其中词汇表中的单词或短语映射到实数向量。从概念上讲, 它涉及从每个单词具有多个维度的空间到具有更低维度的连续向量空间的数学嵌入。生成这种映射的方法包括神经网络、单词共现矩阵上的降维[19-21]、概率模型[22]、可解释知识库方法[23] 以及单词出现上下文的显式表示[24], 单词和短语嵌入作为底层的输入表示。该方法已经被证明可以提高 NLP 任务的性能, 比如语法分析和情感分析[25]。本文通过 Skip-gram 模型[26] 的 Word Embedding 技术对查询条件进行语义近似扩展, 即根据查询关键字, 利用 Skip-gram 技术生成与其语义相关的关键字信息。Skip-gram 模型的训练目标是找到对预测句子或文档中的周围单词有用的单词表示。给定一系列训练单词 $\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_T$, Skip-gram 模型可以预测出这些单词的上下文关系, 从而实现语义近似查询, Skip-gram 结构如图 1 所示。

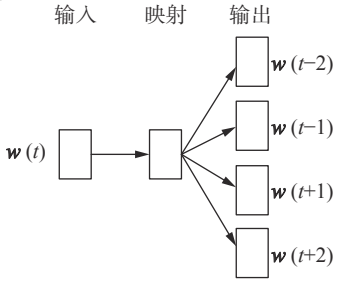


图 1　Skip-gram 模型架构

Fig. 1　Skip-gram model architecture

Skyline 查询技术是一种典型的偏好查询方法, 由于它具有从多维数据集中提取用户感兴趣信息的能力, 因此受到了广泛研究。Skyline 在涉及多准则决策的应用中被广泛使用[27], 进一步应用到 top-k 查询[28-29]、偏好收集和最近邻搜索[30]。Skyline 是指在数据集中不受任何其他元组支配的元组集合[18]。如果 q 在至少一个维度上优于 p, 并且在所有其他维度上不比 p 差, 则称 q 支配 p。此外, 如果一对元组 p 和 q 都不支配彼此, 则元组 p 和 q 都应该在 Skyline 中。例如, 一个客户想要寻找一个度假村, 假设他综合考虑 3 个条件: 价格、酒店级别和停车位数量。价格低, 级别高, 停车位多无疑是更好的选择。因此, 如果 p 在 Skyline 中, 则没有其他的不在 Skyline 中的 q 都比 p 拥有更高的价格、更低的级别、更少的停车位。因此 Skyline 方法在对查询结果进行个性化处理方面具有很大的优势。然而, Skyline 方法只能对数值属性进行计算, 并不能对文本信息等其他非数值属性进行处理。故本文在处理查询结果的过程中利用 Skyline 方法来对数值属性进行处理, 从而实现个性化查询的目标, 使其更加满足用户的需求。

2　问题定义和解决方案

2.1　问题定义

本文先通过一个例子说明要解决的问题。图 2 给出了 9 个空间对象, 空心圆代表空间对象, 每个空间对象包含的文本关键字和数值属性信息如表 2 所示。

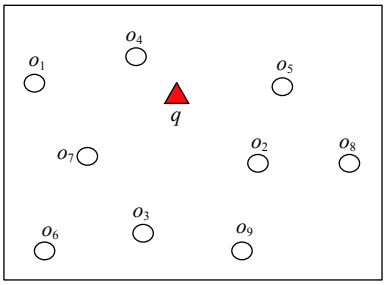


图 2　空间对象的地理位置信息

Fig. 2　Geographic location information for spatial objects

对于一个给定的空间关键字查询 q(图 2 中的三角形表示), 目的是寻找距离查询位置最近, 提供“chicken”食品, 具有“价格低”、“噪声小”且“不拥挤”等特点的“KFC”店。如果进行严格的文本匹配, 则没有满足条件的对象。但实际上, KFC 与 McDonald 语义相似, 故 o_2、o_4、o_7 可作为考虑对象; 进而, 这三者相比, o_4 距查询位置最近, o_7 在数值属性上优于 o_4, o_7 在价格属性上优于 o_2, 但 o_2 在噪声和拥挤度上明显优于 o_7。在这种情况

下，如果用户对价格的在意程度较低，则三者中最优的查询结果应该是 o_2，反之，最优的查询结果应该是 o_7。由此可见，空间对象中包含的数值信息以及用户对数值信息的重视程度，对于空间关键字查询也至关重要，并且其处理方法与文本关键字的匹配处理方法完全不同。

表 2　空间对象的文本和数值属性信息

Table 2　Text and numerical attribute information for spatial objects

空间对象	位置信息(latitude, longitude)	文本属性keywords	数值属性		
			噪声	价格	拥挤度
o_1	(33.330 690 2, −111.978 599 2)	pizza, steak	0.3	0.5	0.7
o_2	(41.119 534 6, −81.475 689 8)	chicken, McDonald	0.2	0.6	0.4
o_3	(33.524 902 5, −112.115 309 8)	tea, coffee	0.3	0.4	0.5
o_4	(40.291 685 3, −80.104 899 9)	chicken, McDonald	0.5	0.3	0.6
o_5	(33.383 146 8, −111.964 725 4)	shopping, market	0.8	0.7	0.9
o_6	(48.727 2, 9.147 95)	bar, beer, chicken	0.9	0.7	0.9
o_7	(40.615 102 244 5, −80.091 348 746 5)	chicken, McDonald	0.3	0.3	0.5
o_8	(36.197 484 4, −115.249 660 1)	bread, sandwich	0.4	0.3	0.4
o_9	(36.207 43, −115.268 46)	movie, drink	0.2	0.4	0.3
q	(34.2, −81.839)	chicken, KFC	low	low	low

给定一个空间数据集 $O=\{o_1, o_2,\cdots,o_n\}$，$O$ 中的每个对象 o_i 由一个三元组 (λ, K, A) 表示，其中 $o_i.\lambda$ 是对象 o_i 的位置信息，$o_i.K$ 是 o_i 中的文本关键字集合，$o_i.A$ 是 o_i 中的数值属性集合，$o_i.A$ 中的 $o.a_i$ 标准化到 [0,1] 之间。假设这些数值属性的值越小越好，如：噪声低、价格低等。如果数值属性值越高越好，例如：环境氛围、评分等信息，则可以通过 $a_i=1-a_i$ 将其转换。查询 q 由三元组 (λ, K, W) 表示，其中 $q.\lambda$ 是查询条件的位置信息，$q.K$ 是查询关键字集合，$q.W$ 是不同数值属性的权重的集合和用户对于这些数值属性的偏好，$q.w \in q.W$，$q.w \geqslant 0$ $(i=1,2,\cdots,|q.W|)$ 并且 $\sum_{i=1}^{|q.W|} q.w_i = 1$。为每个数值属性赋权重而不是限定属性的精确查询范围，目的是为了实现数值属性上的模糊查询。

2.2　解决方案

本文提出的解决方案如图 3 所示，可分为 2 步：

1) 对于用户发起的查询 $q=(\lambda, K, W)$，根据离线阶段计算的语义相关性和字符串相似度，将满足条件的关键字拓展到查询关键字的集合中；

2) 构建 AIR-Tree 混合索引结构，对查询条件与数据库中的空间对象进行位置相近性、文本相似度和数值接近度的计算，根据最终得分筛选出最符合用户需求的 top-k 个结果。

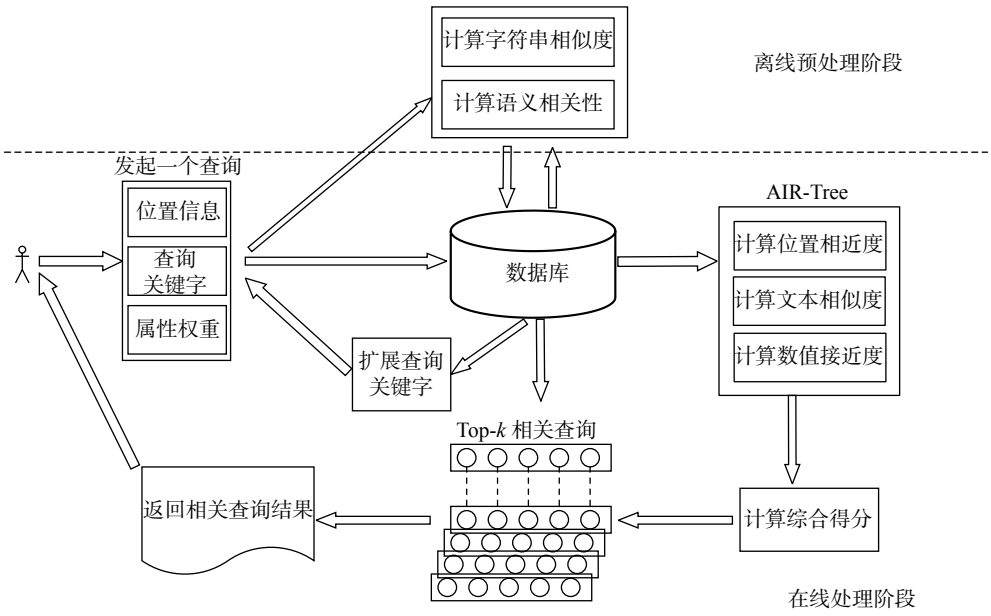


图 3　解决方案框图

Fig. 3　Solution block diagram

3 查询与结果的相关性评估

查询条件与空间对象的相关性, 主要包括位置相近度、字符串相似度、语义相关性、文本相似度和数值接近度。

3.1 位置相近度

给定一个查询 q 与空间对象 o, 它们的位置相近度计算方法如下:

$$S_D(q,o)=1-\frac{\mathrm{dist}(q.\lambda,o.\lambda)}{D_{\mathrm{Max}}} \tag{1}$$

式中: $\mathrm{dist}(q.\lambda,o.\lambda)$ 为空间对象 o 与查询 q 的欧氏距离; D_{Max} 为所有对象集合 O 中的最大距离。

对于表 2 中的空间对象, 可以计算出对象集合 O 中的最大距离 D_{Max} 为 92.1394, 故查询位置与所有空间对象之间的距离分别为: 0.6728、0.9248、0.6713、0.9313、0.6729、0.0000、0.9278、0.6367、0.6365。

3.2 字符串相似度

查询 q 与空间对象 o 中文本信息的字符串相似度可通过式 (2) 计算:

$$S_F=1-\frac{\mathrm{ld}(q.s,o.s)}{\max(\mathrm{len}(q.s),\mathrm{len}(o.s))} \tag{2}$$

式中: $\mathrm{ld}(q.s,o.s)$ 为 q 与 o 对应关键字之间的编辑距离; len() 是求字符长度的函数; $\max(\mathrm{len}(q.s),\mathrm{len}(o.s))$ 为 q 与 o 对应关键字长度的最大值。

对于查询条件“KFC, chicken”, 假设用户输入的是“KCF, chicken”, 则其字符串相似度为 0.8182。

3.3 语义相关度

首先通过 Skip-gram 模型的 Word Embedding 技术对查询关键字进行扩展。Skip-gram 模型的训练目标是找到对预测句子或文档中的周围单词有用的单词表示。给定一系列训练单词 $\boldsymbol{w}_1$, $\boldsymbol{w}_2$, …, $\boldsymbol{w}_T$, Skip-gram 模型的目标是最大化平均对数概率, 即

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leqslant j\leqslant c,j\neq 0}\log p(\boldsymbol{w}_{t+j}|\boldsymbol{w}_t) \tag{3}$$

式中 c 是训练上下文的大小 (可以是中心词 $\boldsymbol{w}_t$ 的函数)。较大的 c 导致更多训练示例, 因此可以以训练时间为代价来获得更高的准确性。基本的 Skip-gram 公式使用 softmax 函数定义 $p(\boldsymbol{w}_{t+j}|\boldsymbol{w}_t)$, 其中 softmax 函数为

$$p(\boldsymbol{w}_O|\boldsymbol{w}_I)=\frac{\exp(\boldsymbol{v}'^{\mathrm{T}}_{w_O}\boldsymbol{v}_{w_I})}{\sum_{w=1}^{W}\exp(\boldsymbol{v}'^{\mathrm{T}}_{w}\boldsymbol{v}_{w_I})} \tag{4}$$

式中: $\boldsymbol{v}_{w_I}$ 和 $\boldsymbol{v}_{w_O}$ 是 $\boldsymbol{w}$ 的向量表示的输入和输出向量; W 是词汇表中的单词个数。由于其计算代价 $\nabla\lg p(\boldsymbol{w}_O|\boldsymbol{w}_I)$ 正比于 W, 该数一般情况下很大 (10^5~10^7)。

虽然噪声对比度估计 (noise comparison evaluation, NCE) 可近似地最大化 softmax 的对数概率, 但是 Skip-gram 模型只关心学习高质量的向量表示, 因此只要向量表示保持其质量, 就可以自由地简化 NCE。负采样 (NEG) 的目标函数为

$$\log\sigma(\boldsymbol{v}'^{\mathrm{T}}_{w_O}\boldsymbol{v}_{w_I})+\sum_{i=1}^{k}\mathrm{E}_{w_i\sim P_n(w)}[\log\sigma(-\boldsymbol{v}'^{\mathrm{T}}_{w_i}\boldsymbol{v}_{w_I})] \tag{5}$$

NCE 和 NEG 均将噪声分布 $P_n(\boldsymbol{w})$ 作为一个随机参数。文献 [26] 研究了 $P_n(\boldsymbol{w})$ 的多种选择, 发现 unigram 分布 $U(\boldsymbol{w})$ 提高到 $U(\boldsymbol{w})^{3/4}/Z$ 时在 NCE 和 NEG 的每个任务上都明显优于 unigram 和均匀分布。因此本文也采用该值作为 $P_n(\boldsymbol{w})$ 的默认值。

为了解决罕见词和频繁词之间的不平衡, 本文采用了一种简单的下采样方法: 将训练集中的每个单词 $\boldsymbol{w}_i$ 丢弃, 丢弃概率的计算公式为

$$P(\boldsymbol{w}_i)=1-\sqrt{\frac{t}{f(\boldsymbol{w}_i)}} \tag{6}$$

式中: $f(\boldsymbol{w}_i)$ 为单词 $\boldsymbol{w}_i$ 的频率; t 为所选阈值, 一般在 10^{-5} 左右。该下采样公式能够对频率大于 t 的单词进行下采样, 同时保留频率的排序。

对于图 2 中的例子, 利用词嵌入技术得到的扩展关键字查询为{Chicken, KFC, McDonald}。进而, 如果空间对象的文本中也包含了 Chicken 或 McDonald, 其在语义上也与初始查询关键字十分相关, 因此也可能是候选查询结果。

3.4 文本相似度

空间对象 o 与查询 q 的文本相似度评估的基本思想是, 先将空间对象文本和查询关键字进行向量化处理, 分别用 $\boldsymbol{V}_o$ 和 $\boldsymbol{V}_q$ 表示, 再利用 Cosine 相似度方法计算文本相似性, 计算方法为

$$S_T(o,q)=\frac{\sum_{i=1}^{n}\boldsymbol{V}_o[i]\cdot\boldsymbol{V}_q[i]}{\sqrt{\sum_{i=1}^{n}\boldsymbol{V}_o[i]^2}\cdot\sqrt{\sum_{i=1}^{n}\boldsymbol{V}_q[i]^2}} \tag{7}$$

现有的空间关键字查询仅在形式上匹配关键字, 通常仅考虑文本相似度而未考虑查询与文本的语义相关度和拼写错误的情况。

对于表 2 中的空间对象, 可以分别计算出查询关键字与其文本相似性为: 0.0000、0.4082、0.0000、0.4082、0.0000、0.3333、0.4082、0.0000、0.0000。

3.5 Skyline

假设关系 D 有 n 个元组和 m+1 个数值, 令 $A=\{A_1, A_2, \cdots, A_m\}$, $t[A_i]$ 为属性 A_i 上元组 t 的值。假设对于每一个属性, 在支配关系 dominate 中的值有一个关于偏好的总的排序 (例如, $a>b$ 表明值 a 优于值 b)。一个元组 $t\in D$ 支配另一个元组

$t' \in D$, 由 $t > t'$ 表示, 当且仅当 $\forall A \in A$, $t[A] \geqslant t'[A]$ 和 $\exists A \in A$, $t[A] > t'[A]$。另外, 如果一个元组 $t \in D$ 与另一个元组 $t' \in D$ 是不可比的, 则表示为 $t \sim t'$, 当且仅当 $t \not\succ t'$ 且 $t' \not\succ t$。

Skyline S 是 D 中不被其他元组支配的元组集合。对于图 2 中的对象 o_2、o_4、o_7, 其数值属性可分别表示为{0.2, 0.6, 0.4}, {0.5, 0.3, 0.6}, {0.3, 0.3, 0.5}。若属性值越小越好, 则由于 o_7 的第一、三个属性优于 o_4, 第二个属性与 o_4 的相等, 可判断 o_7 支配 o_4, 再比较 o_2、o_7, 可知 o_2 的第一、三个属性优于 o_7, 第二个属性次于 o_7, 故 o_2 和 o_7 是不可比的, 故都应该加入 S 中。

3.6　评价函数

本文首先使用式 (2) 对查询条件与空间对象关联的文档中的关键字进行字符串相似度计算, 若查询关键字与文档中的关键字高于给定的字符串相似度阈值 τ, 则将该关键字扩展到查询关键字的集合中。字符串相似度计算的目的是解决由于查询关键字拼写错误而导致空查询结果的情况。

其次, 根据离线阶段计算的语义相关性将满足给定阈值的关键字拓展到查询关键字的集合中, 从而进行语义近似扩展。目前, 普遍采用的空间对象 o 与查询 q 的相关度计算方法为

$$S_1(q,o) = \alpha \cdot S_D(q,o) + (1-\alpha) \cdot S_T(q,o) \tag{8}$$

式中 α 为调节参数。为了不失一般性, 本文将其设置为 0.5。

本文在此基础上加上了数值属性对查询结果的影响, 所采用的空间对象 o 与查询 q 之间数值属性的相关度计算方法为

$$S_2(q,o) = 1 - \sum_{i=0}^{|q.W_i|} (q.W_i \cdot o.a_i) \tag{9}$$

式中: 权重数组 $q.W_i$ 是查询 q 对空间对象 o 的数值属性的看重程度; $|q.W_i|$是权重个数 (也就是数值属性个数); $o.a_i$ 是空间对象 o 的数值属性值。

最后使用评价函数, 即用式 (10) 来计算最后的综合得分:

$$S(q,o) = \beta \cdot S_1 + (1-\beta) \cdot S_2 \tag{10}$$

式中 β 为调节参数, 本文设置为 0.85。

4　查询匹配算法

4.1　索引结构

本文提出的索引结构主要分为语义层、AIR-Tree 索引层。语义层首先使用式 (2) 计算查询 q 与空间对象关联文档集合中的关键字之间的字符串相似度, 如果文档中的关键字与查询关键字的字符串相似度高于给定阈值, 则对查询关键字进行扩展。然后, 语义层首先通过 Word Embedding 技术, 找到语义相关的关键字, 然后对查询关键字进行语义扩展。扩展的查询关键字同时包含了字符串相似和语义相关的关键字。

在 AIR-Tree 索引层中, 构建了一种新的索引结构 (AIR-Tree), 即在 IR-Tree 的基础上为每个节点增加了 AttrFile 文件, 见图 4。AIR-Tree 的每个节点记录了以该节点为根的子树中所有对象的空间信息、文本信息概要 (从节点文本信息中抽取的关键字集合)、数值属性元组信息及指针, 如图 4 所示。AIR-Tree 中每个节点的信息分为 3 个部分: 前两部分是两个指针, 分别指向包含该节点所有关键字的倒排文件 (InvFile) 和数值属性文件 (AttrFile), 第三部分是该节点中的实体集合 (Entries)。每个非叶子节点和叶子节点都可能包含多个条目, 对于叶子节点, 它当中的每一个条目由一个四元组构成, 形式为<O, R, t, a>, 其中 O 代表空间对象, R 代表该对象的最小外接矩形 (MBR), t 是该对象的文本信息标识符, a 是该对象的数值属性元组的标识符; 对于非叶子节点, 它当中的每一项也由一个四元组构成, 形式为<p_N, R, p, a>, 其中, p_N 是该节点中孩子节点 N 的地址, R 是指能够包含该节点下所有孩子节点的 MBR, p 是该节点的文档标识符, 文档包含了该节点下所有子节点的信息概要, a 是该节点的数值属性标识符, 包含了该节点下所有子节点的数值属性元组的 Skyline 集合。

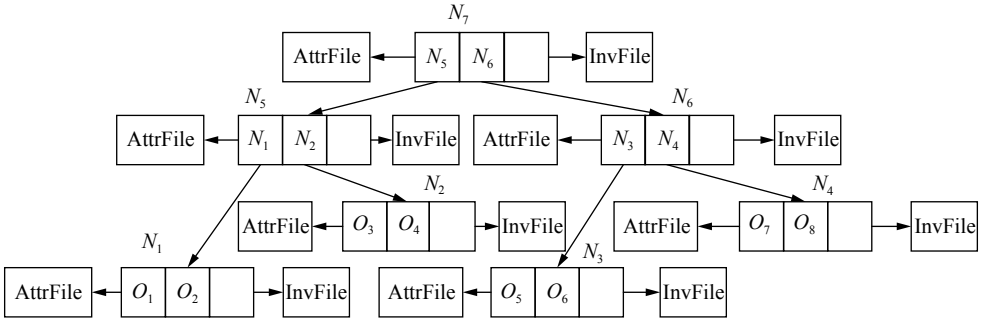


图 4　AIR-Tree 索引结构

Fig. 4　AIR-Tree index structure

在查询过程中, 利用算法 1 中 AIR-Tree 索引结构对 POI 兴趣点的位置信息、文本信息以及数值属性进行 top-k 查询。首先使用式 (8) 计算出与查询条件中位置信息以及关键字信息的相关度, 其次使用式 (9) 计算出数值属性相关性, 再使用式 (10) 计算出最后综合分数, 最后按照综合得分对查询结果进行排序, 使其筛选出最符合用户需求的前 k 个结果, 从而使得查询结果更个性化。

4.2 相关算法

本节给出 AIR-Tree 索引中用到的函数以及算法。首先, 需要在离线阶段计算 AIR-Tree 每个节点所对应的 Skyline 集合, 实现方法如算法 1 所示。

算法 1　skyline(L)

输入　数值属性列表 L;

输出　数值属性列表 L。

1) 对 L 按照第一个属性的值从小到大排序

2) 令 $i = 0$

3) for 每个在 L 列表中的元素元组 t

4) while i<t.size() do

5) if $t[i]$<=$t_1[i]$ then /*将 $t[i]$ 与 L 中其他元组 t_1 的第 i 个元素 $t_1[i]$ 相比*/

6) L.remove(t_1) /*将其他元组 t_1 从列表中移除*/

7) i += 1

8) else

9) L.remove(t)

10) return L

算法 1 可以为用户提供独立于其他数据项的数据项。在查询过程中, 数值接近度 S_2 由 Skyline 和用户指定的权重决定, 从而决定着哪个分支被选中而加快查找速度。对于一个给定的数值属性列表 L, 根据其第一个属性的大小正序排序。L 中的每一个元组的值都会被逐列比较, 如果一个元组的每一列都比其他元组的该列大, 则将该元组从 L 中移除, 直到不再存在这样的元组为止。

对于一个给定的空间关键字查询, 利用 AIR-Tree 索引结构获取候选查询结果的实现算法如算法 2 所示。

算法 2　候选集结果生成算法 search(q, k, α, β, $W[i]$)

输入　数据集 D, 扩展后的查询条件 q, 返回结果个数 k, 调节参数 α, β, 权重数组 $W[i]$;

输出　候选集列表 R。

1) $R\leftarrow\phi$, 最大堆 $H\leftarrow\phi$, $h_E\leftarrow\phi$

2) H.add(r) /*r 为根节点*/

3) while $H\neq\phi$ 并且 R.size()<k do

4) N=H.poll()

5) if N is an object then

6) R.add(N)

7) S = 1

8) else

9) for entry e in N

10) h_E.add(e)

11) if $q.K$ 中包含 h_E.getId().getKeyword() then

12) 用式 (8) 计算位置信息与关键字信息之间的耦合相关度的分数 S_1

13) for 每个 h_E.getId()

14) 根据节点的 AttrFile, 将其数值属性的元组信息添加到候选元组列表 c_t 列表中

15) $s_t\leftarrow$skyline(c_t) /*计算 skyline 元组列表 s_t 的 Skyline 集合*/

16) for 每个在 s_t 中的元素 t

17) $S_2(q,o) = 1-\sum_{i=0}^{n}(W[i]*t[i])$

18) 利用式 (10) 计算出 S

19) H.add(h_E)/*H 按照 S 排序*/

20) return R

算法 2 用于查找 top-k 个最符合用户需求的候选集。首先, 使用离线阶段训练好的 Word Embedding 技术和字符串相似度计算方法对查询关键字进行扩展, 然后使用离线阶段构建好的 AIR-tree 来查找与查询条件最接近的空间对象。首先, 根节点 r 被添加到最大堆 H 中, 如果该节点是空间对象 (即叶子节点), 则将其添加到 R 中, S 设置为 1。否则, 即为中间节点, 然后中序遍历该树, 将其孩子节点添加到 h_E 中。如果 h_E 中的叶子节点包含扩展后的查询关键字, 则使用式 (8) 计算 S_1, 根据其 AttrFile, 将其数值属性的元组信息添加到候选元组列表 s_t 中, 计算 s_t 的 Skyline 集合, 对于每个在 s_t 中的元素 t, 使用式 (9) 计算 S_2。最后使用式 (10) 计算最终得分 S。最后, 将 h_E 添加到 H 中, 并且迭代, 直到 H 为空或者 R 中元素个数大于 k。

算法 1 的复杂度分析: 令 n=H.size(), m=s_t.size(), 则由以上算法分析可得其时间复杂度为 $O(kmn)$, 其中 k 表示查询结果个数。

5 效果与性能实验评价

本节主要通过在真实数据集上进行实验来验证本文所提出算法的性能。

5.1 实验设置

本实验通过 Pytorch 实现 Word Embedding, 其

中负采样随机采样数量 K 为 100，指定周围单词数 C 为 3 进行预测，迭代轮数为 2，每轮迭代 1 个 batch 的数量为 128，词汇表的大小设置为 30 000，学习率为 0.000 1，词向量维度为 100 作为超参数。从文本文件中读取所有的文字，通过这些文本创建一个 vocabulary，由于单词数量可能太大，故只选取最常见的 30 000 个单词，同时添加一个<unk>单词表示最不常见的单词。本文使用 Adam 作为优化器来优化模型。

本文利用 Skip-gram 模型的词嵌入技术在 Yelp 数据集上进行训练，可以得到查询关键字与其语义相关的关键字，如表 3 所示。

表 3　词嵌入技术对查询关键字的语义扩展效果

Table 3　Effect of word embedding technology for query expansion

查询关键字	与查询关键字语义相关的关键字
arts entertainment	shopping, beauty & spas, home services, health & medical, automotive
charlotter	restaurants, shopping, beauty & spa, home service, health & medical
beauty & spas	las vegas, food, phoenix, home services, nightlife
tea rooms	beauty & shopping, beauty & spas, home services, local services, active life
coffee & tea	shopping, beauty & spas, home services, health & medical, local services
breakfast & brunch	home services, health & medical, active life, hair salons, home & garden
food	restaurants, food, nightlife, bars, sandwiches

根据表 3 的结果可以判定 Word Embedding 技术有较好的语义近似处理能力。

本实验所使用的第 1 个数据集为从 Yelp 商户点评网站上抓取的真实的 POI 数据集，Yelp 是美国著名商户点评网站，其网站包含了各地餐馆、购物中心、酒店等各个领域的商户信息以及用户评价和签到时间等信息。将这些真实 POI 数据处理成 174 567 个兴趣点，使得每个 POI 兴趣点都有一个 ID、位置信息 (以经纬度的形式表示)、文本信息、数值属性。将位置信息作为空间信息，用户评论信息和 category 作为文本信息，随机产生的 5 个 0~1 之间的随机数作为数值属性。

第 2 个数据集来自基于位置的服务平台 Foursquare。数据清理后，数据集包含 215 614 个与地理位置相关的对象、关键字列表以及数值属性的标准化值。每个空间对象包含经纬度信息、关键字信息，如牛排、披萨、咖啡等，以及 4 个数值属性，即价格、环境、服务和评级。

参数的默认值在表 4 中给出。在实验过程中，通过改变一个参数的值，固定其他参数的值来研究该参数对实验结果的影响。所有实验都采用 Java 实现，电脑配置为 CPU i7-8700K3.7 GHz，32 GB 内存，Ubuntu 18.04.1 操作系统。

表 4　参数的默认值

Table 4　Default values for parameters

参数	默认值	描述信息
k	10	top-k结果集个数
α	0.5	空间相近性与文本相关性的调节参数
β	0.85	空间相近性和文本相关性之间的耦合相关性与数值属性的调节参数
$\|o.A\|$	4	数值属性数量
$\|q.K\|$	7	查询关键字的数量
$\|D\|$	1	所有空间对象的数量
τ	0.55	字符串相似度的阈值

5.2　实验结果与分析

本节主要研究在相同数据集上，表 4 中各参数分别对 AIR-Tree、IR-Tree 以及 IRS-Tree 在查询效率和查询效果上的评估。“索引 (Y/F)”表示该索引结构在 Yelp/Foursquare 数据集上进行的实验。

5.2.1　查询效率方面的实验

1) k 对查询响应时间的影响

该实验的目的是通过设置 k 的值为 10~60 来观测查询结果个数对系统响应时间的影响。从图 5 可知，在相同的数据集上无论哪种算法，k 值越大，系统响应时间越久。这是因为 k 值越大，越多的候选对象被索引，返回越多条与查询条件相近的 POI 空间对象，因此查询时间越久。此外，IR-Tree 的查询响应时间最短，因为它既没有考虑数值属性，也没考虑拼写错误的情况，更无需考虑与查询关键字语义相关的关键字查询，因此查

询时间最短; 其次是 AIR-Tree, 原因是它需要利用 Skyline 查询方法对数值属性进行处理; 耗时最久的是 IRS-Tree, 因为它要求数值属性的精确范围来完成查询匹配, 增加了查询成本, 故耗时最久。综上所述, 当 k 值增大时, AIR-Tree 索引的查询效率较高, 相比于 IRS-Tree 提高了 3.6%。

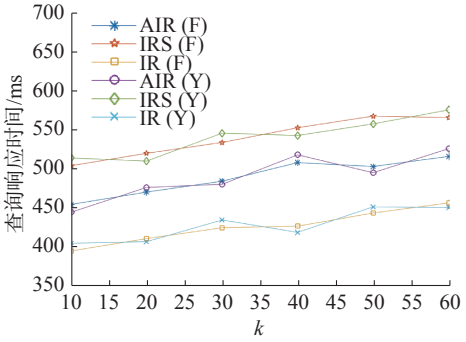


图 5　k 对查询响应时间的影响

Fig. 5　Effect of k on the query execution time

由于在 Foursquare 和 Yelp 上的实验结果相差甚微, 故本节剩余部分只展示在 Yelp 数据集上的实验结果。

2) $|o.A|$对查询响应时间的影响

该实验的目的是通过改变数值属性的个数来验证其对查询响应时间的影响。本文通过改变 $|o.A|$的值从 1~5 来观测其对查询响应时间的影响, 如图 6。实验结果表明, 随着数值属性个数的增加, 查询时间也逐渐增加。这是由于 AIR-Tree 在查询结果中需要对数值属性的元组进行 Skyline 计算, 在最坏情况下, Skyline 方法几乎会将每个元组中的每个元素进行比较, 因此数值属性个数越多, 越耗费时间。IRS-Tree 比 AIR-Tree 耗时, 因为 IRS-Tree 在处理数值属性时, 考虑数值属性的精确范围, 若范围设置过大, 则会非常耗时。综上可见, 当数值属性的个数增多时, AIR-Tree 查询效率最佳。

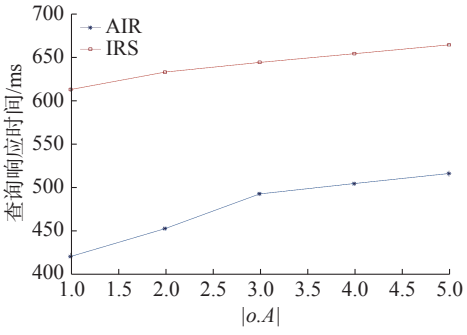


图 6　$|o.A|$对查询响应时间的影响

Fig. 6　Effect of $|o.A|$ on the query execution time

3) $|q.K|$对查询时间的影响

该实验的目的是通过设置查询关键字的数量从 1~7 来观测其对查询响应时间的影响。由图 7 可知, 查询响应时间与查询关键字的个数成正比增长。因为无论哪种索引结构, 当查询关键字增多时, 则索引到的包含查询关键字的对象越多, 因此查询时间会增加。

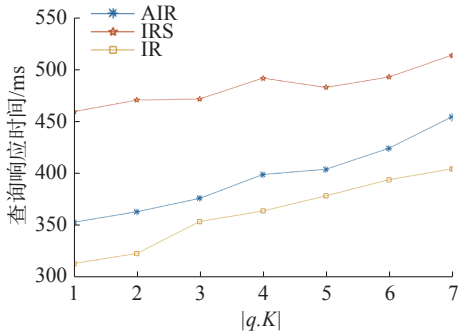


图 7　$|q.K|$对查询响应时间的影响

Fig. 7　Effect of $|q.K|$ on the query execution time

4) $|D|$对查询时间的影响

该实验的目的是通过设置数据集大小从 1, 2,⋯, 200 000 来观测数据集大小对查询响应时间的影响。从图 8 可知, 查询响应时间随着数据集的增加而增加。这是因为数据集越大, 需要索引的对象越多, 并且在处理数值属性的过程中可能需要耗费更多的时间。

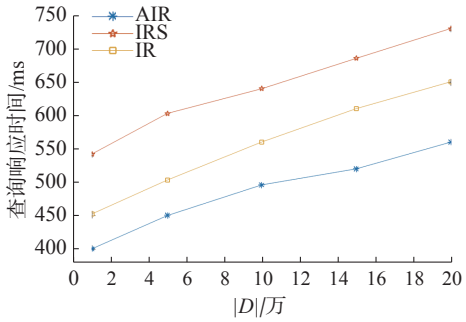


图 8　$|D|$对查询响应时间的影响

Fig. 8　Effect of $|D|$ on the query execution time

5.2.2　效果方面的实验

1) $|D|$与构建索引时间的关系

该实验的目的是测试在相同数据集上, 通过改变数据集的大小, 从 1, 2,⋯, 200 000 来比较以上几种算法构建索引所用时间, 以评价其性能。由图 9 可知, 索引构建时间与数据集的大小成正比。其中构建 IR-Tree 所用时间最少, 这是因为 IR-Tree 与 AIR-Tree、IRS-Tree 相比, 不需要构建 AttrFile 文件和 synopses, 因此其索引构建时间最短。而 IRS-Tree 需要将 synopses Tree 与其他索引结合来完成查询, 故其索引构建时间最长。

2) τ 与查询准确率的关系

AIR-Tree 是一个高维近似查询索引, 对于一

些给定的查询 q 可以得到一些语义相关结果。本文使用准确率 P 来评估查询效果，计算公式为

$$P = \frac{|I(q) \cap R(q)|}{|I(q)|} \tag{11}$$

式中：$I(q)$ 是距离查询 q 最近的 top-k 个对象的理想集合；$R(q)$ 是本文提出算法所返回的结果集。这里，为了获取更具有鲁棒性的准确率，将返回结果数设置为 100。由图 10 可知：当 τ<0.55 时，准确率随着字符串相似度的增大而增大；当 $\tau \geqslant$ 0.55 时，准确率不再发生改变并保持在 85% 左右。

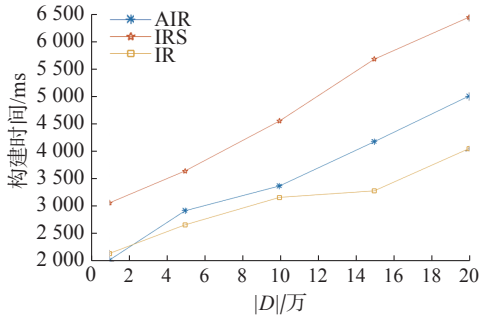


图 9　|*D*|与构建索引所用时间的关系

Fig. 9　Relationship between |*D*| and the time taken to build the index

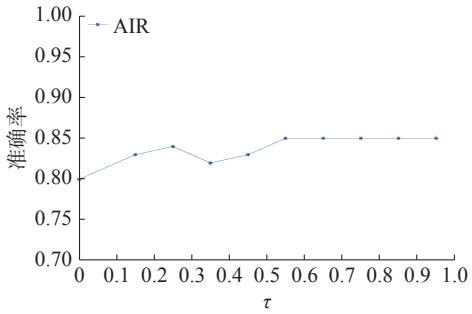


图 10　τ 与查询准确率的关系

Fig. 10　Effect of τ on the query accuracy

3) k 与查询准确率的关系

通过设置 k 来查看其对查询准确率的影响，本文设置 k 区间为 [10, 100]，步长为 10。由图 11 可知，本文所提算法 AIR-Tree 较 IR-Tree、IRS-Tree 分别在准确率上提高了 10.14% 和 16.15%。

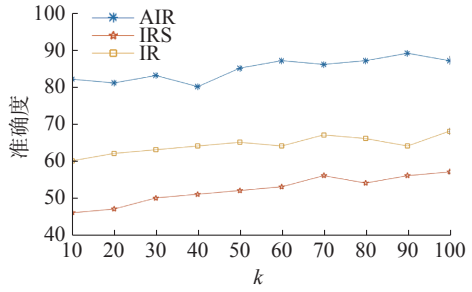


图 11　k 与查询准确率的关系

Fig. 11　Effect of k on the query accuracy

6　结束语

针对现有空间关键字查询处理模式仅支持位置相近和文本匹配，但未考虑语义信息，且不能处理数值属性的问题，提出了一种支持语义近似查询处理的空间关键字查询方法。本文研究了通过 Word Embedding 中的 Skip-gram 模型来实现空间对象的语义近似查询的方法，并且利用 Sky-line 查询方法实现了查询结果的个性化。实验结果表明，本文提出的算法不仅支持空间关键字的精确匹配，支持语义近似查询和形式上的近似匹配，且能处理文本信息中的数值属性，这样更符合用户查询要求，在一定程度上提高了用户体验以及满意度。

在未来的工作中，将会利用深度学习的方法研究路网上的移动对象的集合关键字查询。

参考文献：

[1] LU Ying, LU Jiacheng, CONG Gao, et al. Efficient algorithms and cost models for reverse spatial-keyword k-nearest neighbor search[J]. ACM transactions on database systems, 2014, 39(2): 13.

[2] DE FELIPE I, HRISTIDIS V, RISHE N. Keyword search on spatial databases[C]//Proceedings of 2008 IEEE 24th International Conference on Data Engineering. Cancun, Mexico: IEEE, 2008: 656−665.

[3] ZHANG Chengyuan, ZHANG Ying, ZHANG Wenjie, et al. Inverted linear quadtree: efficient top K spatial keyword search[J]. IEEE transactions on knowledge and data engineering, 2016, 28(7): 1706–1721.

[4] BECKMANN N, KRIEGEL H P, SCHNEIDER R, et al. The R*-tree: an efficient and robust access method for points and rectangles[J]. ACM SIGMOD record, 1990, 19(2): 322–331.

[5] ZHANG Dongxiang, OOI B C, TUNG A K H. Locating mapped resources in web 2.0[C]//Proceedings of 2010 IEEE 26th International Conference on Data Engineering (ICDE 2010). Long Beach, USA: IEEE, 2010: 521−532.

[6] LI Feifei, YAO Bin, TANG Mingwang, et al. Spatial approximate string search[J]. IEEE transactions on knowledge and data engineering, 2013, 25(6): 1394–1409.

[7] ROCHA-JUNIOR J B, VLACHOU A, DOULKERIDIS C, et al. Efficient processing of top-k spatial preference queries[J]. Proceedings of the VLDB endowment, 2010, 4(2): 93–104.

[8] YAO Bao, LI Feifei, HADJIELEFTHERIOU M, et al. Approximate string search in spatial databases[C]//Proceedings of 2010 IEEE 26th International Conference on Data Engineering. Long Beach, CA, USA: IEEE, 2010:

545−556.

[9] QIAN Zhihu, XU Jiajie, ZHENG Kai, et al. Semantic-aware top-k spatial keyword queries[J]. World wide web, 2018, 21(3): 573–594.

[10] 胡骏, 范举, 李国良, 等. 空间数据上 Top-*k* 关键词模糊查询算法 [J]. 计算机学报, 2012, 35(11): 2237–2246.
HU Jun, FAN Ju, LI Guoliang, et al. Top-*k* fuzzy spatial keyword search[J]. Chinese journal of computers, 2012, 35(11): 2237–2246.

[11] 刘喜平, 万常选, 刘德喜, 等. 空间关键词搜索研究综述 [J]. 软件学报, 2016, 27(2): 329–347.
LIU Xiping, WAN Changxuan, LIU Dexi, et al. Survey on spatial keyword search[J]. Journal of software, 2016, 27(2): 329–347.

[12] ZHANG Dongxaing, CHEE Y M, MONDAL A, et al. Keyword search in spatial databases: Towards searching by document[C]//Proceedings of 2009 IEEE 25th International Conference on Data Engineering. Shanghai, China: IEEE, 2009: 688−699.

[13] CONG Gao, JENSEN C S, WU Dingming. Efficient retrieval of the top-k most relevant spatial web objects[J]. Proceedings of the VLDB endowment, 2009, 2(1): 337–348.

[14] KWON H Y, WANG Haixun, WHANG K Y. G-index model: a generic model of index schemes for top-*k* spatial-keyword queries[J]. World wide web, 2015, 18(4): 969–995.

[15] LEVY O, GOLDBERG Y. Linguistic regularities in sparse and explicit word representations[C]//Proceedings of the Eighteenth Conference on Computational Natural Language Learning. Michigan, USA, 2014: 171−180.

[16] MORSE M, PATEL J M, JAGADISH H V. Efficient skyline computation over low-cardinality domains[C]//Proceedings of the 33rd International Conference on Very large Data Bases. Vienna, Austria: VLDB Endowment, 2007: 267−278.

[17] LIU Xiping, CHEN Lei, WAN Changxuan. LINQ: a framework for location-aware indexing and query processing[J]. IEEE transactions on knowledge and data engineering, 2015, 27(5): 1288–1300.

[18] BORZSONY S, KOSSMANN D, STOCKER K. The skyline operator[C]//Proceedings 17th International Conference on Data Engineering. Heidelberg, Germany: IEEE, 2001: 421−430.

[19] LEBRET R, COLLOBERT R. Word embeddings through Hellinger PCA[C]//Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics. Gothenburg, Sweden, 2014: 482−490.

[20] LEVY O, GOLDBERG Y. Neural word embedding as implicit matrix factorization[C]//Proceedings of the 27th International Conference on Neural Information Processing Systems. Montreal, Canada: ACM, 2014: 2177−2185.

[21] LI Yitan, XU Linli, TIAN Fei, et al. Word embedding revisited: a new representation learning and explicit matrix factorization perspective[C]//Proceedings of the Twenty-Fourth International Conference on Artificial Intelligence. Buenos Aires, Argentina, 2015: 3650−3656.

[22] GLOBERSON A, CHECHIK G, PEREIRA F, et al. Euclidean embedding of co-occurrence data[J]. The journal of machine learning research, 2007, 8: 2265–2295.

[23] QURESHI M A, GREENE D. EVE: explainable vector based embedding technique using Wikipedia[J]. Journal of intelligent information systems, 2019, 53(1): 137–165.

[24] SOCHER R, BAUER J, MANNING C D, et al. Parsing with compositional vector grammars[C]//Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics. Sofia, Bulgaria, 2013: 455−465.

[25] SOCHER R, PERELYGIN A, WU J, et al. Recursive deep models for semantic compositionality over a sentiment Treebank[C]//Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing. Washington, USA, 2013: 1631−1642.

[26] MIKOLOV T, SUTSKEVER I, CHEN Kai, et al. Distributed representations of words and phrases and their compositionality[C]//Proceedings of the 26th International Conference on Neural Information Processing Systems. Lake Tahoe, Nevada, USA, 2013: 3111−3119.

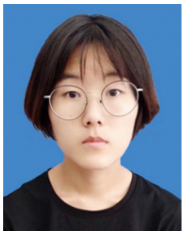
[27] GODFREY P, SHIPLEY R, GRYZ J. Maximal vector computation in large data sets[C]//Proceedings of the 31st International Conference on Very Large Data Bases. Trondheim, Norway, 2005: 229−240.

[28] ASUDEH A, THIRUMURUGANATHAN S, ZHANG Nan. Discovering the skyline of web databases[J]. Proceedings of the VLDB endowment, 2016, 9(7): 600–611.

[29] ILYAS I F, BESKALES G, SOLIMAN M A. A survey of top-*k* query processing techniques in relational database systems[J]. ACM computing surveys, 2008, 40(4): 11.

[30] KOSSMANN D, RAMSAK F, ROST S. Shooting stars in the sky: an online algorithm for skyline queries[C]//Proceedings of the 28th International Conference on Very Large Data Bases. Hong Kong, China, 2002: 275−286.

作者简介:



李盼, 硕士研究生, 主要研究方向为空间关键词查询和空间推荐系统。

张霄雁，博士研究生，主要研究方向为空间数据分析、城市计算和深度学习。提出了空间对象的聚类分析方法、空间–文本数据的语义近似查询和多样性推荐方法，主持辽宁省教育厅科学研究项目 1 项。发表学术论文 10 余篇。

孟祥福，教授，博士生导师，主要研究方向为空间数据管理、推荐系统和大数据可视化等。提出了空间对象的耦合关系分析模型，多样性兴趣点推荐方法和 Web 数据库 top-*k* 个性化检索方法，主持国家自然科学基金 2 项、辽宁省自然科学基金项目等 3 项，发表学术论文 30 余篇。

关于举办“吴文俊人工智能科学技术奖十周年颁奖盛典暨 2020 中国人工智能产业年会”的第二轮通知

为全面实施创新驱动发展战略，贯彻落实国家《新一代人工智能发展规划》，总结“吴文俊人工智能科学技术奖”十年来为我国科技自立自强发挥的积极效用，表彰在人工智能领域做出突出贡献的科研单位和科技工作者，调动广大智能科技领军人才的积极性和创造性，激励原创性科学成果不断涌现，推动我国智能科技与实体经济深度融合，随着国内疫情防控形势的持续好转，中国人工智能学会定于 2021 年 4 月 10 日—12 日在北京、苏州同期举办“‘智创十年 赋能未来’——吴文俊人工智能科学技术奖十周年颁奖盛典暨 2020 中国人工智能产业年会”。

一、时间地点：

2021 年 4 月 10 日—12 日（10 日、星期六上午为颁奖仪式）

北京、苏州工业园区

大会主题：智创十年·赋能未来

二、组织机构：

主办单位：中国人工智能学会

协办单位：苏州工业园区管委会

承办单位：中国人工智能学会吴文俊人工智能科学技术奖评选基地、苏州智博天宫人工智能产业研究院

三、联系方式

（一）中国人工智能学会吴文俊人工智能科学技术奖办公室（北京）

联系人：王老师、武老师

联系电话：010-52365722、52365896

电子邮箱：wuwenjunkejijiang@vip.163.com

活动官网：www.wuwenjunkejijiang.cn

（二）中国人工智能学会吴文俊人工智能科学技术奖评选基地（苏州）

联系人：俞老师、刘老师

联系电话：0512-62927273、62927535

电子邮箱：liuyunzhe@caaipalace.co