软件学报 ISSN 1000-9825, CODEN RUXUEW
*Journal of Software*,2012,23(8):2018−2031 [doi: 10.3724/SP.J.1001.2012.04166]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

# 一种基于学习的高维数据 *c*-近似最近邻查询算法*

袁培森[1+], 沙朝锋[1], 王晓玲[2], 周傲英[2]

[1](上海市智能信息处理重点实验室(复旦大学),上海 200433)

[2](上海市高可信计算重点实验室(华东师范大学),上海 200062)

## *c*-Approximate Nearest Neighbor Query Algorithm Based on Learning for High-Dimensional Data

YUAN Pei-Sen[1+], SHA Chao-Feng[1], WANG Xiao-Ling[2], ZHOU Ao-Ying[2]

[1](Shanghai Key Laboratory of Intelligent Information Processing (Fudan University), Shanghai 200433, China)

[2](Shanghai Key Laboratory of Trustworthy Computing (East China Normal University), Shanghai 200062, China)

+ Corresponding author: E-mail: peiseny@fudan.edu.cn

**Yuan PS, Sha CF, Wang XL, Zhou AY. *c*-Approximate nearest neighbor query algorithm based on learning for high-dimensional data. *Journal of Software*, 2012,23(8):2018−2031 (in Chinese).** http://www.jos.org.cn/1000-9825/4166.htm

**Abstract**: Under the filter-and-refine framework and based on the learning techniques, a data-aware method for *c*-approximate nearest neighbor query for high-dimensional data is proposed in this paper. The study claims that data after random projection satisfies the entropy-maximizing criterion which is needed by the semantic hashing. The binary codes after random projection are treated as the labels, and a group of classifiers are trained, which are used for predicting the binary code for the query. The data objects are selected who's Hamming distances between the query satisfying the threshold as the candidates. The real distances are evaluated on the candidate subset and the smallest one is returned. Experimental results on the synthetic datasets and the real datasets show that this method outperforms the existing work with shorter binary code, in addition, the performance and the result quality can be easily tuned.

**Key words**: random projection; *c*-approximate nearest neighbor query; SVM classifier; high-dimensional data; entropy maximizing criterion; locality sensitive hashing

**摘 要**: 针对高维数据近似最近邻查询,在过滤-验证框架下提出了一种基于学习的数据相关的 *c*-近似最近邻查询算法.证明了数据经过随机投影之后,满足语义哈希技术所需的熵最大化准则.把经过随机投影的二进制数据作为数据的类标号,训练一组分类器用来预测查询的类标号.在此基础上,计算查询与数据集中数据对象的海明距离.最后,在过滤后的候选数据集上计算查询的最近邻.与现有方法相比,该方法对空间需求更小,编码长度更短,效率更高.模拟数据集和真实数据集上的实验结果表明,该方法不仅能够提高查询效率,而且方便调控在查询质量和查询处理时间方面的平衡问题.

**关键词**: 随机投影;*c*-近似最近邻查询;支持向量机分类器;高维数据;熵最大化准则;位置敏感哈希

* 基金项目: 国家自然科学基金(60925008, 60903014); 国家重点基础研究发展计划(973)(2010CB328106); "核心电子器件、高端通用芯片及基础软件产品"国家科技重大专项(2010ZX01042-002-003-004)

收稿时间: 2011-01-24; 修改时间: 2011-04-28, 2011-08-23; 定稿时间: 2011-11-17

中图法分类号: TP311 文献标识码: A

最近邻查询(nearest neighbor query)广泛应用于文本信息检索、搜索引擎、基于图像内容的信息查询[1]以及数据重复性检测等领域.在这些领域,数据的高维性质使得快速、有效的最近邻查询已经变得非常困难.然而在许多查询应用中,查询的响应时间是重要的,通常一个近似的结果也可以满足查询的要求.因此,研究者提出了近似最近邻查询方法来平衡查询结果的质量和查询效率之间的矛盾.

近似最近邻查询一般又称为 *c*-近似最近邻查询(*c*-approximate nearest neighbor query),它返回查询与其真实最近邻半径在 *c* 倍误差之内的任何一个对象作为结果.目前,近似最近邻查询的常用技术是位置敏感的哈希(locality sensitive hashing,简称 LSH)[2].LSH 基本原理是,首先把相似的数据对象以高概率哈希到相同的冲突桶中,非相似的数据对象几乎不被哈希到相同的冲突桶中.对于查询,首先使用一组相同的哈希函数把查询哈希到桶空间,然后把与查询哈希到的冲突桶中的数据对象作为最近邻候选,进而在候选集上计算查询的近似最近邻.然而 LSH 没有考虑具体数据的性质,对不同的数据集,哈希函数族取自相同的分布,是一种数据无关的查询(data independent)技术.为了获得较好的查询效率,需要较大的空间开销.

最近,研究者提出了数据相关的相似性查询(data-aware similarity search).它使用数据挖掘中的学习技术,对查询数据首先进行预处理,在此基础上进行相似性查询.基于学习的方法首先把数据对象转变为二进制编码,使得相似的数据对象,其二进制编码也相似,最后使用二进制编码在海明空间进行距离计算.由于降低了维度,并且计算机对二进制的数据计算效率很高,因此大幅地提高了查询速度.为此,如何设计较好的方法,以获得高质量的二进制编码,对于计算效率和结果的质量具有重要的决定性作用.因此,Baluja[3]和 Weiss[4]等人提出了熵最大化准则来衡量二进制编码的质量.最近,Zhang 等人[5]在相似图矩阵基础上提出了自学习的哈希方法(self-taught hashing,简称 STH).为了获得满足熵最大化的二进制编码,STH 方法首先对数据集构建 *k*-NN 相似性矩阵,并对进行矩阵分解和特征值求解等操作.因此,该方法的时间和空间复杂度较高,并且对于数据的更新就要重新进行处理.

本文在作为 LSH 技术之一的随机投影(random projection)技术基础上,提出了一种基于学习的、数据相关的 *c*-近似最近邻查询的近似处理算法.不同于文献[3–5],本文使用随机投影技术对高维数据进行二进制编码,并且证明了随机投影技术的二进制编码能够满足熵最大化准则.使用随机投影技术对数据编码之后,把投影之后的二进制签名向量作为原始数据的类标号集,训练一组分类器.对于近似最近邻查询,是在过滤-验证框架(filter-and- refine framework)下完成的.首先,对查询使用训练的分类器进行类标号预测;随后,在海明空间计算查询与数据库中数据对象之间的海明距离;选择数据库中与查询对象的海明距离小于一定阈值的数据对象的子集作为候选,继而在候选集上验证,计算查询真实的最近邻.

本文所提出的方法使得数据经过随机投影之后满足熵最大化的准则.本文的方法在空间上减少了 LSH 所需要的二进制向量的长度.与现有方法相比,该方法在相同的查询质量下提高了查询的效率,同时降低了二进制编码的长度.

本文第 1 节介绍相关的研究工作.第 2 节将详细介绍本文所使用的基础知识.第 3 节给出本文方法的处理流程和相关算法.第 4 节描述整个实验设置及其结果分析.在第 5 节对本文工作进行总结.

## 1 相关工作

最近邻查询返回目标数据中与查询距离最近的数据对象.它应用于数据库、信息检索等多个领域并得到了广泛的关注.在低维空间中,研究者提出了基于树索引的数据划分和空间划分的技术,例如 R-tree[6],*k*-d 树[7]等.但是,文献[8]指出,当数据的维度超过 8 时,使用的基于数据划分或者空间划分的技术,其性能变得还不如线性扫描.

对于高维空间的数据,近似最近邻查询能够以较高的效率返回近似的结果而满足查询需求,获得了人们的青睐.Indyk 等人[2]提出的位置敏感哈希是目前近似最近邻查询的最新技术,它在理论上能够在次线性

(sublinear)时间内求解近似最近邻问题.针对不同的应用需求和相似性度量或者距离度量,研究者提出了多种位置敏感的哈希技术.针对高维数据的余弦相似度计算,Charikar[1]提出了随机投影位置敏感哈希函数族;对于基于集合的 Jaccard 相似度,Broder 等人提出了 Min-Hashing 位置敏感哈希技术[9]用于近似计算集合数据之间的相似度;对于范式距离 $L_p$,当 $p\in(0,2)$时,Datar 等人证明了存在哈希函数族,可用于在次线性时间内计算最近邻查询[10].目前,位置敏感哈希技术已经在多个应用领域用于处理数据的近似性查询问题.

Tao 等人利用空间 $z$-曲线填充[11]、LSH 技术及 $B^+$-树索引,提出了 LSBtree[12]处理 $k$-最近邻查询.在基于图像内容的数据的最近邻查询方面,Min[1]提出了 Compact Projection 用于图像数据近似计算最近邻查询.该方法证明了数据的弱可分性($\Delta$-weakly separable),首先通过随机投影技术,把高维数据投影为二进制的编码表示,通过分析具体的数据筛选数据作为候选集,其中,候选集的大小可以表示为 $O(an^l)$.但是,该方法仍需较长的编码来保证查询的质量.

Salakhutdinov 等人[13]提出了基于语义的哈希技术(semantic hashing).该方法通过把高维数据转换为二进制编码,基于相似的数据对象具有相似的二进制编码的原理,使用二进制编码来有效地计算最近邻.如何生成高质量的二进制编码成为提高语义哈希技术性能和查询效果的关键.基于语义的哈希提出了熵最大化准则(entropy maximizing criterion),用以刻画二进制编码的质量问题.Zhang 等人[5]提出了基于学习的相似性查询技术 STH.STH 使用 $k$-NN 图构造数据库的相似性矩阵.为了得到满足熵最大化准则的二进制编码,STH 对相似性矩阵进行特征值和特征向量求解,使用特征向量的中位数作为阈值,把特征向量转换为二进制编码,以满足熵最大化.数据的二进制编码作为该对应数据的类标号,训练一组分类器来计算相似性查询.但是,该方法的数据预处理需要很高的空间和时间代价.

## 2 基于学习的近似性查询

在本文中,数据对象(data object)也称为数据记录(data record),是对一个实体的刻画和数量性质描述,通常表示为一个 $d$ 维实向量空间中的点.本文中的记法如下:数据库 $D$ 具有 $n$ 个数据对象,记为数据对象集合 $O=\{o_1,\ldots,o_n\}$,且 $O\subseteq R^d$.本文中给定两个对象 $o_1,o_2$,它们之间的距离记为 $dist(o_1,o_2)$且 $dist(o_1,o_2)\geqslant 0$;它们的相似性度量记为 $sim(o_1,o_2)$且 $sim(o_1,o_2)\in[0,1]$.除非特别指出,本文中的距离度量指的都是欧氏距离,相似性度量为余弦相似度.

### 2.1 问题定义

给定数据库中数据对象集合 $O$ 和该数据集合上的查询 $q$,最近邻查询(nearest neighbor query)返回满足 $dist(q,o^*)=\min_{o\in O}dist(q,o)$的数据对象 $o^*$.对精确最近邻查询,最简单的处理方法是对数据集执行线性扫描并计算与查询之间的距离,对结果排序并返回距离最小的一个数据对象.这样,对于具有 $n$ 个 $d$ 维数据记录的数据集,需要 $O(nd)$的时间复杂度.对于大量且高维的数据集,线性扫描的方式效率不高.

在大多数的查询应用中,用户并不需要精确的结果,近似的结果也可以满足用户的查询需要.比如在基于内容的图像检索应用中,一个内容近似的结果就可以满足查询的要求.因此,研究者提出了近似最近邻查询,又称为 $c$-近似最近邻查询($c$-approximate nearest neighbor query).针对近似最近邻查询问题,本文提出了基于随机投影学习的 $c$-近似最近邻查询技术.$c$-近似最近邻查询的定义如下:

**定义 1($c$-近似最近邻查询)**. 给定数据对象集合 $O\subseteq R^d$,构造算法 $A$,对于每一个查询 $q\in R^d$,返回数据对象 $o^*\in O$,满足条件 $dist(q,o^*)\leqslant c\cdot\min_{o\in O}dist(q,o)$,其中,$c$ 称为近似因子且 $c>1$.

给定包含 5 个数据对象的集合 $O=\{o_1,\ldots,o_5\}$和 $O$ 上的查询 $q$.查询 $q$ 在集合 $O$ 上的 $c$-近似最近邻查询如图 1 所示.在数据集合 $O$ 上,查询 $q$ 的最近邻对象是 $o_1$,$q$ 与 $o_1$ 之间的距离为 $r$.因此,在以 $q$ 为圆心、以 $c\cdot r$ 为半径的圆内的任何一个数据对象$\{o_1,\ldots,o_4\}$,均可作为结果返回,其中,$c$ 为近似因子.

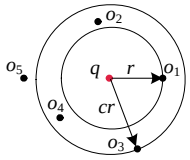



Fig.1 $c$-Approximate nearest neighbor query

图 1 $c$-近似最近邻查询

### 2.2 熵最大化准则

对于二元随机变量 $X$,假设 $X$ 的概率分布为 $\Pr(X=1)=p$,$\Pr(X=0)=1-p$.二元随机变量的信息熵又称为二元熵,定义为 $H(X)=-p\log p-(1-p)\log(1-p)$.当 $p=1/2$ 时,$H(X)$取得最大值.为了刻画二进制编码的质量,基于学习的语义哈希技术提出了熵最大化准则(entropy maximizing criterion)[4],即要求哈希值中 0 和 1 出现的概率均等且各个比特位无关.对于熵最大化准则,形式化为性质 1.

**性质 1(熵最大化准则)**. 语义哈希的二进制编码需满足熵最大化准则,即编码中的 0 和 1 出现的概率相同且相互无关.

熵最大化准则的意义在于,使编码的每一位最大化地表示目标数据的信息,可以使用最短的二进制编码长度表示原始数据.

### 2.3 LSH随机投影

Indyk 等人[2]提出的 LSH 已被广泛应用于高维数据近似相似性查询.针对不同的相似度和距离函数,研究者提出了不同的 LSH 哈希函数族.其中,针对高维数据的余弦相似度近似计算,提出了随机投影(random projection)[14]LSH 技术.随机投影把高维向量投影到低维的随机超平面,进而在低维的海明空间内近似地计算数据对象之间的余弦相似性.

给定向量 $\boldsymbol{u}\in R^d$,随机投影的哈希函数族 $H$ 的每一个函数 $h_{\boldsymbol{r}}$ 如等式(1),其中,随机向量 $\boldsymbol{r}\in R^d$,其每一项随机独立地取自标准正态分布 $N(0,1)$.

$$h_r(u)=\begin{cases}1, & \text{if } r\cdot u\geqslant 0\\ 0, & \text{otherwise}\end{cases},h_r\in H \tag{1}$$

给定两个向量 $\boldsymbol{u},\boldsymbol{v}$ 和随机投影哈希函数族 $H$,设 $\theta(\boldsymbol{u},\boldsymbol{v})$为两个向量 $\boldsymbol{u},\boldsymbol{v}$ 之间的夹角,则可以得到等式(2)[14].

$$\Pr[h_r(u)=h_r(v)]=1-\frac{\theta(u,v)}{\pi} \tag{2}$$

通过等式(2)可以看出,向量之间的余弦相似度可以通过它们投影之后的哈希函数值近似地计算.本文中,投影之后的哈希函数值也称为签名(signature).

对于基于随机投影的 LSH,为了改进查询质量,通常选取 $L$ 个 $d$ 维随机向量 $\boldsymbol{r}_1,\ldots,\boldsymbol{r}_L$ 构成随机矩阵,记为 $M^{d\times L}$.对于向量 $\boldsymbol{u}$,经过 LSH 随机投影之后,可以计算出其签名 $s(\boldsymbol{u})=\{h_{r_1}(\boldsymbol{u}),...,h_{r_L}(\boldsymbol{u})\}\in\{0,1\}^L$ .对数据库中的每一个对象,计算其二进制签名之后,可以获得一个签名矩阵 $S$,如等式(3)所示.

$$S=\begin{pmatrix}h_{r_{11}}(u_1) & \cdots & h_{r_{1L}}(u_1)\\ \vdots & \ddots & \vdots\\ h_{r_{d1}}(u_n) & \cdots & h_{r_{dL}}(u_n)\end{pmatrix} \tag{3}$$

与 STH 方法[5]的处理类似,本文把数据经过随机投影之后的二进制签名作为数据的二进制编码,在此基础上,使用学习算法来有效地计算近似最近邻查询.从直观上理解,相似的数据对象,其随机投影之后的编码相似的概率也较高.这一点可以从等式(2)看出来,与 Salakhutdinov 等人提出的基于语义的哈希二进制编码的原理相似.下面证明数据经过随机投影之后,其二进制签名满足熵最大化准则.

### 2.4 证 明

设向量 $\boldsymbol{u}\in R^d$ 为 $d$ 维实空间正规化向量,即$\|\boldsymbol{u}\|=1$.设随机向量 $\boldsymbol{r}\in R^d$,其中,$\boldsymbol{r}$ 的每一分量均随机地取自标准正态分布 $N(0,1)$,则有定理 1.

**定理 1**. 设变量 $v=\boldsymbol{u}\cdot\boldsymbol{r}$,则 $v\sim N(0,1)$.

证明:由 $v=\boldsymbol{u}\cdot\boldsymbol{r}$,则 $v=\sum_{i=1}^{d}\boldsymbol{u}_i\cdot\boldsymbol{r}_i$. 由于随机变量 $\boldsymbol{r}_i$ 随机地取自 $N(0,1)$,则 $\boldsymbol{u}_i\cdot\boldsymbol{r}_i\sim N(0,\boldsymbol{u}_i^2)$.

所以,根据正态分布的可加性,则随机变量 $v\sim N(0,\sum_{i=1}^{d}\boldsymbol{u}_i^2)=N(0,1)$. □

**定理 2**. 设 $f(x)=\mathrm{sgn}(x)$是是定义在实数集 $R$ 上的符号函数,当 $x\geqslant 0$ 时,$f(x)=1$;否则,$f(x)=0$.设随机变量 $v'=f(v)$,其中,$v=\boldsymbol{u}\cdot\boldsymbol{r}$,则 $\Pr(v'=0)=\Pr(v'=1)=1/2$.

证明:$\Pr(v'=0)=\Pr(f(v)=0)=\Pr(v\leqslant 0)$.根据定理 1,随机变量 $v$ 服从标准正态分布 $N(0,1)$,可得 $\Pr(v\leqslant 0)=\Phi(0)=1/2$.当 $v'=1$ 时,$\Pr(v'=1)=1-\Pr(v'=0)=1/2$. □

定理 2 表明,随机变量 $v$ 取值为 0 和 1 的概率均为 1/2,即向量 $\boldsymbol{u}$ 的 $L$ 次随机投影的位向量中的 0 和 1 出现的概率是一样的.

根据定理 1 可知:投影之后的位向量中,每个分量是相互独立的;并根据定理 2,由此可以得到定理 3.

**定理 3**. 随机投影之后的二进制编码 0,1 出现的概率均等且相互无关,可以使得信息熵最大化,即满足熵最大化准则.

本文方法的两个依据是:(1) 随机投影把高维的向量投影到低维的海明空间,使得高维空间中相似的数据向量,其在海明空间的二进制位向量之间的海明距离较小,因而具有较多的相同的位编码;(2) 根据数据挖掘中最近邻分类的原理[15],距离越近的数据对象,则它们属于同一个类的可能性越高.

因此,在本文中把二进制签名矩阵 $S$ 作为数据库中数据的标号集,在此基础上训练分类器.这样避免了 STH 方法[5]中矩阵处理所需的极高的时间和空间复杂度,极大地提高了效率,并且对于数据的更新,本文的方法不需要对已经投影之后的数据重新进行投影.

## 3 流程和算法

### 3.1 处理框架

本文采用了过滤-验证的处理架构.投影之后的二进制编码近似地反映了数据之间的距离,首先在海明空间近似地计算它们之间的海明距离,在海明空间的过滤操作去除海明距离大于阈值的数据对象,最后在过滤之后的数据对象集合上计算 $c$-近似最近邻.

本文的处理框架如图 2 所示.首先是预处理阶段.该阶段对数据库中的所有数据进行 $L$ 次随机投影,以获得数据的二进制签名矩阵 $S$.接着,把签名矩阵每一列作为数据库中对应数据的类标号集,训练了一组分类器.对于用户的查询,首先使用分类器预测查询对应的类标号集,把类标号集转换为二进制编码,在海明空间计算查询与数据的签名之间的海明距离.选择海明距离小于一定阈值的数据对象作为候选子集,在候选子集上计算与查询距离最小的数据对象.

使用签名矩阵训练分类器的过程如图 3 所示.首先把二进制签名矩阵 $S$ 的每一个列向量作为数据集中对应数据的类标号,这样,$L$ 列共有 $L$ 种标号方法.在这些“伪标号”的数据集上,训练 $L$ 个 SVM 分类器.对于用户的查询输入,首先已训练的 $L$ 个分类器,为查询预测 $L$ 个类标号.到此,查询和数据集都转化到了海明空间.在此基础上,使用查询的类标号集与数据库中的签名矩阵之间的海明距离.本文中二进制编码{0,1}可以与类标号{−1,1}之间方便地转换,编码 0 转变为标号−1,编码 1 转变为标号 1.接着,选择与查询之间海明距离满足一定阈值的数据对象作为候选集,计算查询与候选对象之间的欧氏距离.

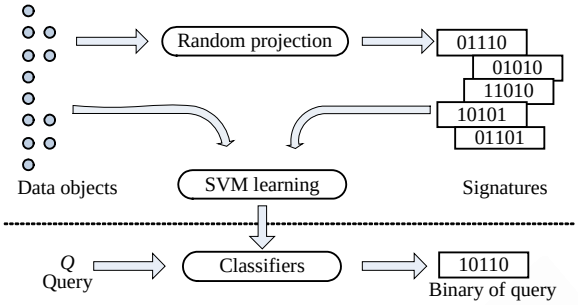



Fig.2 Processing framework

图 2 处理框架

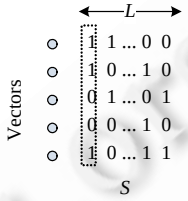



Fig.3 Training SVM classifiers with signature as the labels

图 3 把签名作为标号训练 SVM 分类器

### 3.2 算 法

数据库中的数据经过随机投影之后,由原来的高维的实数空间转变为使用二进制编码表示的海明空间.在具体介绍相关的算法之前,本文定义了海明空间中 $R$-半径球盖集的概念.

**定义 2($R$-半径球盖集($R$-spherical cap))**. 给定二元向量集合 $\boldsymbol{V}_b$,半径 $R$ 和一个二元查询向量 $q$,$q$ 在集合 $\boldsymbol{V}_b$ 上的 $R$-半径球盖集定义为 $C(R,q)=\{v|v\in\boldsymbol{V}_b,dist_{\mathrm{H}}(q,v)\leqslant R\}$,其中,$dist_{\mathrm{H}}(q,v)$两个二进制向量 $q$,$v$ 之间的海明距离,即 $dist_{\mathrm{H}}=|\{i|q_i\neq v_i\}|$.

$R$-半径球盖集定义了数据库中的数据对象与查询之间的海明距离小于整数阈值 $R$ 的所有对象集合.

#### 3.2.1 预处理

在数据预处理阶段,本文方法与 STH 不同之处是采用随机投影之后的哈希值作为数据的标类签.本文中使用的位置敏感哈希随机投影算法流程如图 4 所示.

```
Input: Data vectors, O⊆R^d.
Output: Signature matrix S.
1:  S=∅;
2:  Generate the random matrix M^{d×L};  //矩阵的每一项随机取自标准正态分布
3:  for each u∈O do
4:      v=∅;
5:      for each column r of M^{d×L} do
6:          h=r·u;
7:          val=sgn(h);
8:          v.add(val);
9:      S.add(v);
10: return S;
```

Fig.4 Algorithm of random projection

图 4 LSH 随机投影

算法对输入的数据集首先生成 $d$ 个长度为 $L$ 的随机向量,向量的每一个数据项均随机地取自标准的正态分布 $N(0,1)$.由这 $d$ 个随机向量构成一个随机矩阵记为 $M^{d\times L}$(第 2 行).对数据库中的每一个数据对象 $\boldsymbol{u}$,通过随机投

影计算长度为 $L$ 二进制签名向量(第 3 行~第 9 行).对数据库的所有数据经过 $L$ 次的随机投影之后,可以得到一个二进制的矩阵 $S^{n\times L}$,称为签名矩阵.算法最后返回数据库的签名矩阵 $S$(第 10 行).

3.2.2 训练分类器

数据库经过随机投影之后,把获得的二进制签名矩阵的每一列作为相应数据的“人工标号”,然后使用这些带有人工标号的数据,训练一组分类器.对于分类器算法,本文采用了经典的 SVM 分类器算法.SVM 分类器是一种有监督学习分类器算法.它首先使用带标号的训练数据集,学习一个模型对未知的数据 $\boldsymbol{x}$ 进行分类.该数据模型如式(4)所示.

$$f(x)=\mathrm{sgn}(\boldsymbol{w}^T\boldsymbol{x}+b) \tag{4}$$

其中,参数 $\boldsymbol{w}$ 和 $b$ 是给定 $n$ 个训练集$(\boldsymbol{x}_i,y_i),i=1,\ldots,n,\boldsymbol{x}_i\in R^d,y_i\in\{0,1\}$学习而得到.参数的计算可以转化为等式(5)的优化问题.

$$\left.\begin{aligned}&\min_{w,b,\xi_i}\frac{1}{2}\boldsymbol{w}^T\boldsymbol{w}+C\sum_{i=1}^{n}\xi_i\\&\text{s.t. }y_i(\boldsymbol{w}^T\boldsymbol{x}_i+b)\geqslant 1-\xi_i,\ \xi_i\geqslant 0,\ i=1,\ldots,n\end{aligned}\right\} \tag{5}$$

使用随机投影的位向量作为数据标号的 SVM 训练算法流程如图 5 所示.算法的输入为数据库和该数据库的 LSH 投影之后的签名矩阵 $S$.算法把 LSH 随机投影之后的签名矩阵 $S$ 作为数据库数据的标号集,对于签名矩阵的每一列,作为对数据库的数据一次标号.使用这一列标号的数据,训练一个 SVM 分类器.保存训练好的每一个 SVM 分类器.最后,算法训练 $L$ 个分类器作为返回结果(第 1 行~第 5 行).

```
Input: Data vectors O⊆R^d, signature matrix S.
Output: L SVM classifiers.
1: for (i=1; i<L; i++) do
2:     for each u∈O do
3:         v_i=getLabel(S,u);   //获得向量 u 的签名数据
4:         SVMTrain(u,v_i[j]);
5: return (w_j,b_j);
```

Fig.5 SVM training algorithm

图 5 SVM 训练算法

3.2.3 $c$-近似最近邻查询

对于 $c$-近似最近邻查询 $q$,其处理流程是:使用已经训练的 $L$ 个 SVM 分类器,预测查询 $q$ 的 $L$ 个标号数据,并把这 $L$ 个标号转化为 0 和 1 编码.这样,把查询也转换到海明空间,同时查询的编码中获得了数据集的信息.$c$-近似最近邻查询算法流程如图 6 所示.

```
Input: Query q, L SVM classifiers, signature matrix S, Hamming ball radius R.
Output: The c-approximate nearest neighbor object o*.
1:  HammingDistList=∅;
2:  q_v=∅;   //查询的二进制向量编码
3:  for (i=1; i⩽L; i++) do
4:      bit=SVMPredict(q,svm_i);
5:      q_v.add(bit);
6:  for each(s∈S) do
7:      distVal=dist_H(s,q_v);
8:      HammingDistList=HammingDistList∪distVal;
9:  sort(HammingDistList);
10: Õ=getRCap(q,O);   //在数据集 O 上计算查询 q 的 R-半径球盖集
11: o*=argmin_{o∈Õ} dist(q,o);
12: return o*;
```

Fig.6 $c$-Approximate nearest neighbor query

图 6 $c$-近似最近邻查询

算法采用了过滤-净化框架(filter-and-refine framework),其流程可划分为 3 步:

第 1 步,对于输入的查询 $q$,算法首先使用 $L$ 个已经训练好的 SVM 分类器,预测查询 $q$ 的 $L$ 个类标号,这对应于算法的第 2 行~第 5 行;

第 2 步,在海明空间计算查询与数据库每一个对象的在签名矩阵中对应的签名向量之间的海明距离,这对应于算法的第 6 行~第 9 行.

第 3 步是从数据库中选择出 $q$ 的 $R$-半径球盖集 $\tilde{O}$,把它作为查询结果的候选子集,即与查询之间的海明距离小于阈值 $R$ 的数据对象作为候选集(第 10 行).最后为在候选集合上的验证阶段,即计算查询与候选子集之间的欧氏距离,返回最小的作为近似最近邻查询结果,这对应于算法的第 11 行、第 12 行.

### 3.3 复杂性分析

设具有 $n$ 个数据对象的数据集 $O\subseteq R^d$,对于本文的算法 1,生成有随机向量构成的矩阵 $M^{d\times L}$ 的时间和空间复杂度都为 $O(dL)$.设 $z$ 为每一个向量中非零项的个数,则训练 $L$ 个 SVM 分类器的复杂度最大为 $O(\ln z)$[16].

对于查询处理,用 $L$ 个 SVM 处理器计算 $L$ 个位的时间复杂度为 $O(\ln z\log n)$,对海明空间结果的排序代价为 $O(n\log n)$.设海明球包集的大小记为 $|C(R,q)|=C^*$,则查询与候选的计算代价为 $O(C^*L)$.

因此,查询的代价为 $O(\ln z\log n+C^*L+n\log n)$.

## 4 实验与分析

本文在模拟数据集和真实数据集上进行了实验验证,并在查询性能、查询质量和编码长度方面与本文最相关的工作 Compact Projection[1]进行了实验对比.

### 4.1 实验设置

本文的算法使用 Java SDK1.6 编程实现,实验运行机器配置 CPU 为 Intel Core™ 2 Duo 2.33 GHz,内存为 3.8GB.实验中,近似因子 $c$ 的值默认设置为 1.1.

对于模拟数据集,本文使用随机数生成向量的每一维,采用了两个最常见的分布类型:标准正态分布 $N(0,1)$;区间[0.0,1.0]之间的均匀分布.其中,模拟数据集的向量长度设置为 50.本文对模拟数据集进行了标准化处理,模拟数据集的统计信息见表 1.

**Table 1** Statistical information of artificial experimental datasets

**表 1** 模拟测试数据集统计信息

| Dataset | Vector number | Vector length | Query number |
|---|---|---|---|
| Gaussian | 10 000 | 50 | 50 |
| Normal | 10 000 | 50 | 50 |

实验测试的真实数据集为两个著名的文档数据集 WebKB(http://www.cs.cmu.edu/~webkb/)和 Reuters21578(简记为 Reuters(http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html)).经过去除停用词(stop-words)和词干化(stemming)处理之后,使用 TF-IDF 模型生成文档向量.数据集的统计信息总结见表 2.

**Table 2** Statistical information of real experimental datasets

**表 2** 真实测试数据集统计信息

| Dataset | Vector number | Query number | Vector length | Class number |
|---|---|---|---|---|
| WebKB | 2 785 | 50 | 7 287 | 5 |
| Reuters | 5 485 | 50 | 14 575 | 8 |

对于 SVM 分类器,本文使用 SVM Java 包 LibSVM[17].SVM 分类器的参数采用了默认的设置,核函数设置为线性核函数.在每一个数据集上,分别提交了 50 个查询,这 50 个查询均匀分布在文档集中的每一类文档上.

### 4.2 实验结果与分析

为了评价 $c$-近似最近邻查询结果的质量,本文定义了平均成功概率 Average Success Ratio(ASR)来衡量近

似查询的成功的概率.其定义如下:

**定义 3(平均成功概率)**. 对于一个查询集合 $Q$ 和海明球半径 $R$,对任一查询 $q_i \in Q$,

$$o^* = \{o \mid \min_{o \in C(R,q_i)} dist(q_i,o)\},$$

其中,$C(R,q_i)$为查询 $q_i$ 的 $R$-半径球盖.

设 $O'$为与查询在最小半径 $c$ 倍之内所有的数据对象,即 $O'=\{o|o \in O, dist(q_i,o) \leqslant c \cdot \min_{o' \in O} dist(q_i,o')\}$,查询 $q_i$ 成功率为

$$s_i = \begin{cases} 1, & \text{if } o^* \in O' \\ 0, & \text{otherwise} \end{cases},$$

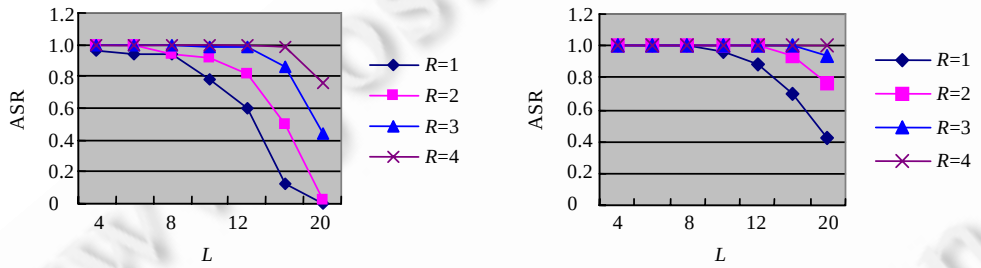
则对于查询集合 $Q$,ASR 定义为等式(6).

$$ASR = \frac{\sum_{i=1}^{|Q|} s_i}{|Q|} \tag{6}$$

平均成功的概率 ASR 值越大,表明查询的结果效果越好.

4.2.1 $c$-近似最近邻查询质量

对于本文的查询处理结果质量,本文在两个模拟数据集和真实的文本数据集上分别进行了测试.图 7 是在两个模拟数据集上分别提交 50 个查询测试的平均结果.其中,图 7(a)和图 7(b)分别是在标准正态分布和均匀分布模拟数据集上 ASR 随着投影长度 $L$ 以及球包集半径 $R$ 变换的情况.$L$ 的取值为 4~20,球半径 $R$ 的取值为 1~4.




(a) 正态分布数据集 (b) 均匀分布数据集

Fig.7 Query quality on synthetic datasets

图 7 在模拟数据集上的查询质量

在如图 7(a)所示的正态分布数据集上,当 $R$ 为 1 时,ASR 从 1 降低到 0 左右;当 $R$ 为 4 时,$L$ 增加到 20,ASR 从 1.0 降低到 0.8 左右;当投影长度 $L$ 一定时,随着半径 $R$ 的增加,ASR 增加,因为随着 $R$ 的增加,候选集中候选对象的个数增加.当 $L$ 小于 12 时,$R$ 为 1~3,ASR 从 1.0 降低到 0.8 左右,变化不大.在如图 7(b)所示的[0.0,1.0]区间上的均匀分布数据集上,ASR 呈现了类似的结果.但不同的是,对于相同的半径参数 $R$,$L$ 的增加,ASR 的降低幅度低于正态分布数据集.当 $L$ 小于 8,$R$ 取值为 1~4 时,ASR 的值均为 1,即 $c$-近似最近邻查询全部成功;当 $L$ 大于 10,$R$ 取值 1 时,ASR 变化较大,为 1~0.5 左右.

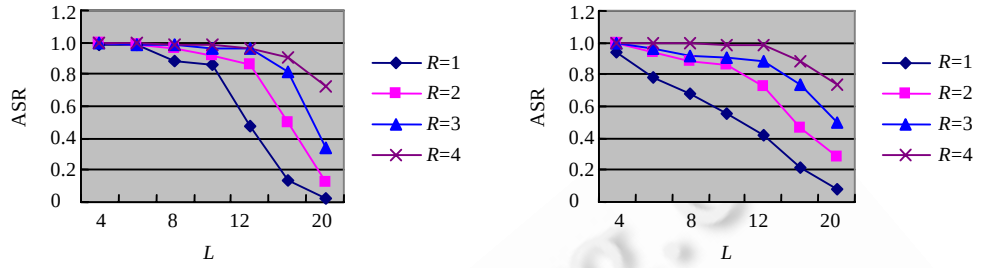
从模拟数据集上的实验结果可以得出如下结论:(1) 在正态分布数据集上,当海明球半径参数 $R$ 一定时,随着随机投影长度 $L$ 的增加,平均查询成功的概率 ASR 在下降.因为海明球半径阈值一定,投影长度的增加会导致候选集中候选对象个数的降低,进而降低查询成功的概率.(2) 均匀分布数据集上的查询平均成功概率高于标准正态分布数据集.

图 8(a)和图 8(b)分别是在真实数据集 WebKB 和 Reuters 上,$c$-近似最近邻查询质量 ASR 随着随机投影的二进制编码长度 $L$ 变化的示意图.横轴表示投影之后的二进制编码长度 $L$,纵轴为数据集上提交的 50 个查询的平均成功概率 ASR.$L$ 的取值变化为 4~20,海明球半径阈值 $R$ 的取值变化为 1~4.

从两个真实数据集上的实验图可以看出:(1) 随着 $L$ 的增加,平均成功比率 ASR 总体呈现下降的趋势,ASR

从 1 开始减小;(2) 随着半径 $R$ 的增加,查询质量也随之增加.

两个数据集上的实验显示:当编码长度 $L$ 小于 10 时,对于 $R$ 的变化,查询质量变化不大,平均查询成功的概率 ASR 的值都大于 0.95;当 $L$ 大于 10 时,半径 $R$ 的变化对结果的影响较大.半径 $R$ 的增加,使得 ASR 下降比较快,从 0.98 下降到 0.2 以下.但是当半径 $R$ 取值为 4 时,编码长度 $L$ 的变化对查询结果的质量影响比较小,平均查询成功的概率 ASR 从接近于 1 降低到 0.75 左右,质量比较稳定.




(a) WebKB 数据集 (b) Reuters 数据集

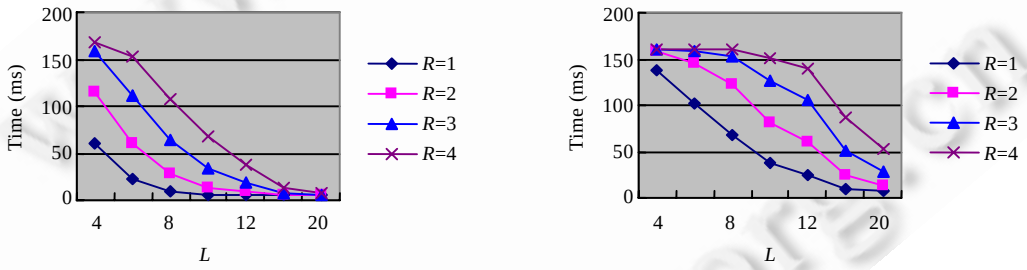
Fig.8 Query quality on real datasets

图 8 在真实数据集上的查询质量

另外,从模拟数据集和两个真实数据集上的查询结果可以看出,真实数据集上的质量与标准正态分布模拟数据集上的查询结果趋势相类似.

4.2.2 查询的性能

对于查询的性能,在两个模拟数据集上的实验结果如图 9 所示.图 9(a)和图 9(b)分别是在正态分布数据集和均匀分布数据集上的查询性能.图中的纵轴为时间,单位为 ms.




(a) 正态分布数据集 (b) 均匀分布数据集

Fig.9 Query performance on synthetic datasets

图 9 在模拟数据集上的查询性能

在正态分布数据集上,随着 $L$ 的增加,查询的时间快速地下降.当 $L$ 一定时,随着半径参数 $R$ 的降低,查询的时间也随之减少.当 $L$ 大于 12 时,无论 $R$ 在 1~4 内如何取值,查询时间都远远低于线性扫描的时间;当 $R$ 等于 4 时也仅为线性扫描的 20%.其中,$L$ 为 4 且 $R$ 等于 4 的查询时间为线性扫描时间.

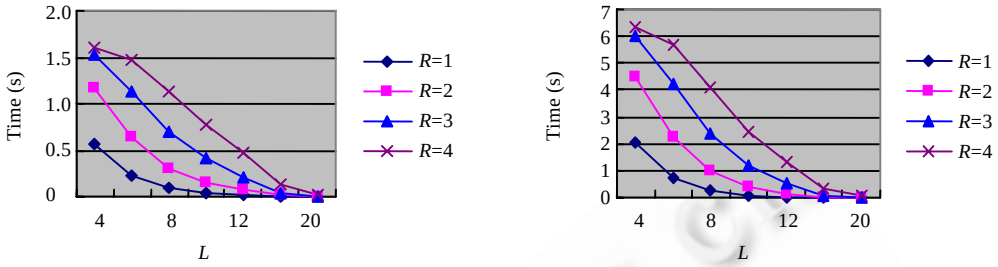
在均匀分布数据集上,随着 $L$ 的增加,查询的时间快速地降低.$R$ 越小,查询的时间越小.因为 $R$ 越小,候选集的个数越少.当 $L$ 大于 12,$R$ 等于 1 和 2 时,查询的时间远低于线性扫描,仅为线性扫描的 30%左右.

从两个模拟数据集上可以看出,两个数据集上的查询性能呈现相似的趋势,但是均匀分布数据集上的查询性能低于正态分布数据集.

图 10(a)和图 10(b)是本文的方法分别在两个数据集 WebKB 和 Reuters 上,50 个查询的平均查询性能随着编码长度 $L$ 和半径 $R$ 的变化而变化的情况.横坐标为二进制编码长度 $L$,纵坐标为查询的处理时间,单位为 s.在

数据集 WebKB 和 Reuters 上采用线性扫描方计算最近邻的时间花销分别是 1 696ms 和 6 366ms.

从图 10(a)和图 10(b)可以看出:(1) 近似查询处理的时间随着 $L$ 的增加,查询处理的时间迅速降低;(2) 随着半径 $R$ 的减少,查询时间减少.当 $L$ 超过 15 以后,查询时间趋向于一个很小的数,只有线性扫描的 10%左右.这表明本文的方法具有良好的过滤效果.




(a) WebKB 数据集 (b) Reuters 数据集

Fig.10 Query performance on real datasets

图 10 在真实数据集上的查询性能

通过图 7 和图 9 的模拟数据集上的查询结果的质量和查询性能可以得出:当 $R$ 等于 4,$L$ 的取值范围在 16~20 时,平均查询成功概率 ASR 在 0.8 以上,查询处理的时间低于线性扫描的 9%;当 $R$ 等于 3,$L$ 的取值范围在 12~16 时,ASR 的值在 0.86 以上,查询时间为线性扫描的 12%左右;当 $R$ 取值为 3,$L$ 的取值范围在 8~12 时,ASR 的值在 0.8 以上,查询时间为线性扫描时间的 20%左右;当 $R$ 等于 1,$L$ 取值为 4~8 时,平均成功概率 ASR 在 1 左右,且在正态分布数据集上,查询时间不到线性扫描时间的 25%.

综合图 8 和图 10 两个真实数据集上的近似查询的质量和查询性能实验可以看出,虽然查询的性能得到了极大的提高,但是查询的质量随着 $L$ 的增大而降低,因为 $L$ 的增加导致候选集大小的减少.根据两组实验可以看出:当二进制编码长度 $L$ 为 16,半径 $R$ 为 4 时,在两个数据集 WebKB 和 Reuters 上查询的平均成功概率在 0.9 以上;查询的时间分别为 131ms 和 300ms 左右,分别是其是线性扫描的 7%和 4%左右,远低于线性扫描的时间.

4.2.3 可扩展性

为了测试本文算法的可扩展性,本文在标准正态分布模拟数据集上测试了随着数据集大小的变化,算法的查询性能和对应的查询结果质量.实验中的数据集向量长度为 50,数据集的大小取值分别为 10 000,50 000 和 100 000.查询的结果如图 11 和图 12 所示.其中,图 11 为在 3 个数据集的候选集上的查询计算时间,图 12 为对应的查询质量.图中的横轴 $L\#R*$表示随机投影长度 $L$ 为#,半径 $R$ 为*.

在本实验中,$L$ 和 $R$ 的取值是查询性能和查询结果的质量折中时的取值组合.从实验结果图中可以看出,随着数据量的增加,查询处理的时间增加并不大,尤其是数据量在 50 000 和 100 000 时,表明了本文的方法具有良好的过滤效果.图 12 中的查询质量表明,与图 11 中对应的查询时间,其查询结果平均成功的概率基本都在 0.8 以上.综合图 11 和图 12 的结果可以看出,本文的方法对于大规模的数据也具有良好的查询性能和质量.

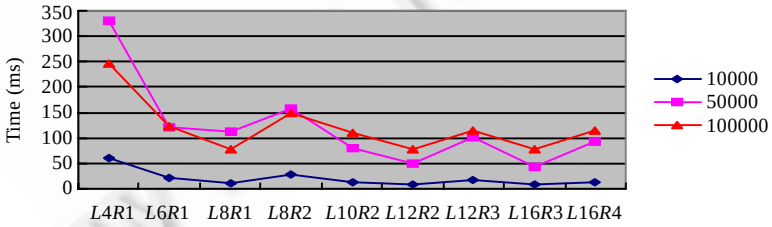



Fig.11 Performance scalability of the query

图 11 查询性能可扩展性

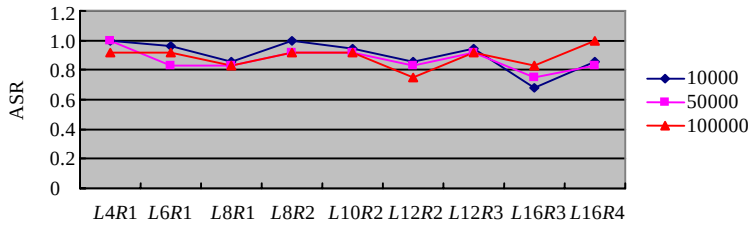



Fig.12 Quality scalability of the query

图 12 查询质量可扩展性

4.2.4 查询质量与性能对比实验

在查询的质量、性能和编码长度方面,本文与最相关的工作——Compact Projection[1],在两个真实数据集上做了实验对比.

Compact Projection 提出了数据$\Delta$-弱可分性性($\Delta$-weakly separability).对于满足$\Delta$-弱可分性性的数据集计算近似最近邻查询的过程是:对数据集和查询随机投影,在海明空间计算海明距离之后,选择与查询的海明距离 $T=an^l$ 个最小数据作为候选子集,进而在候选子集上计算欧氏距离最小作为最终结果.其中,$n$ 为数据库中记录的个数,$a$ 和 $l$ 为参数.在候选子集上,返回与查询欧氏距离最小的一个作为结果.在实验中选择参数 $a$=0.7,$l$=0.4,使得 Compact Projection 查询的质量是较优的情况下作对比.

图 13 是 Compact Projection 在数据集 WebKB 和 Reuters 上 $c$-近似最近邻查询的质量.横坐标为随机投影的次数,也就是二进制编码的长度 $L$;纵坐标为查询的平均成功概率 ASR.其中,$L$ 的取值范围从 4~1 024.

从图 13 可以看出,Compact Projection 近似查询的质量随着 $L$ 的增大而增加,ASR 从一个不超过 0.5 的值趋近于 1.对于数据集 Reuters,ASR 上升比较快,在 $L$ 超过 200 就达到了最大值.在数据集 WebKB 上 ASR 上升就比较平缓一些,$L$ 的值超过 600 时才达到最大.

综合图 8 和图 10 真实数据集上实验测试可以看出,与 Compact Projection 相比,本文的方法使用更少的二进制编码,获得了较好的近似查询质量,并且如果牺牲一些性能,那么本文的查询可以任意地提高查询成功的概率.可以看出,本文的方法在平衡查询时间开销和查询质量平衡方面比 Compact Projection 具有更好的灵活性.

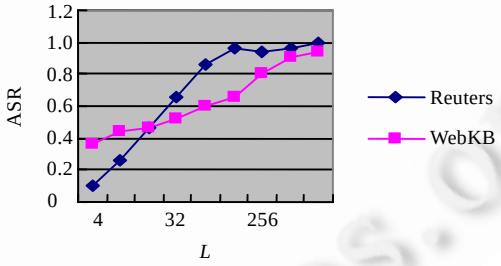



Fig.13 Query quality of the compact projection

图 13 Compact projection 的查询质量

图 14 是本文方法与 Compact Projection 在数据集 WebKB 和 Reuters 上性能方面的对比实验.对于 Compact Projection,取编码长度 $L$ 为 512 时的时间开销,因为此时该方法的查询质量才比较稳定.对于本文的方法,选取 3 组在查询平均成功概率 ASR 方面与 Compact Projection 相当时的编码长度 $L$ 和半径 $R$ 作为比较.

从图 14 可以看出,当参数对$\langle L,R\rangle$={$\langle$10,1$\rangle$,$\langle$16,3$\rangle$,$\langle$20,4$\rangle$}时[**],本文的方法在性能上超过了 Compact Projection,也即具有更好的过滤效果.同时,本文方法的编码长度远低于 Compact Projection 方法.

** 在数据集 Reuters 上,当编码 $L$ 为 10 时,半径 $R$ 的取值为 2,以保证与 Compact Projection 相当的查询质量.

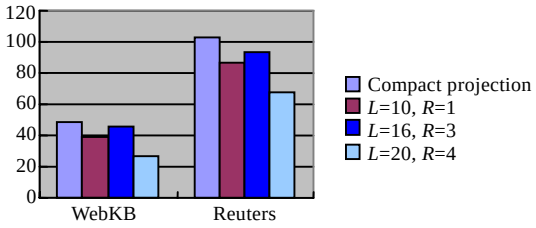



Fig.14 Performance comparison on two datasets

图 14 两个数据集上查询性能比较

本文方法与 Compact Projection 都是采用了过滤后验证的处理流程,即在候选集合上计算 $c$-最近邻查询问题.在性能方面优于 Compact Projection 的原因是过滤后的候选集的大小较少;在编码长度方面优于 Compact Projection,是因为本文在预处理步骤首先考虑了数据之间的关系,将数据之间的关系使用机器学习的技术体现在查询中,缩短了编码长度.

4.2.5 实验结论

虽然本文的方法需要考虑预处理步骤中训练分类器的开销,但是这个步骤可以采用离线学习的方式.由模拟数据集和真实数据集上的实验结果与分析可以看出,对于固定的随机投影编码长度 $L$,增加海明球半径 $R$ 可以增加过滤后候选集中候选对象的个数,可以提高查询的平均成功概率.但是候选子集越大,在验证步骤要花越多的时间.为此,要在查询的成功率和效率方面作一个权衡.因此,根据在模拟数据集和真实数据集上查询的性能和成功的概率,给出使用本文算法时随机投影的编码长度和海明球半径阈值的一组推荐参数值,用以指导随机投影二进制编码长度 $L$ 和海明球半径 $R$ 的选择,以便于平衡查询的性能和平均查询成功概率指标 ASR.参数 $L$ 和 $R$ 参考值的选择见表 3.

**Table 3** Recommendation of parameters $L$ and $R$

**表 3** 参数 $L$ 和 $R$ 推荐值

| $R$ | $L$ | ASR |
|---|---|---|
| 4 | [15,20] | ≥0.8 |
| 3 | [10,15] | 0.75~0.95 |
| 2 | [5,10] | ≥0.85 |
| 1 | [1,5] | ≥0.9 |

## 5 结 论

本文针对 $c$-近似最近邻查询,提出了一种基于学习的近似最近邻查询方法.该方法在位置敏感哈希的随机投影基础上,采用学习算法来处理近最近邻查询.与现有方法相比,该方法使用更少的二进制编码来表示高维空间的数据,具有较好的过滤效果和查询效率,且便于调控查询质量和查询效率.通过在模拟数据集和真实数据集上的实验,表明本文的方法具有良好的查询性能和扩展性,同时对正态分布数据集具有良好的查询性能.

**References**:

[1] Min K, Yang L, Wright J, Wu L, Hua X, Ma Y. Compact projection: Simple and efficient near neighbor search with practical memory requirements. In: Proc. of the IEEE Computer Society Conf. on Computer Vision and Pattern Recognition. San Francisco: IEEE Computer Society, 2010. 3477−3484.

[2] Indyk P, Motwani R. Approximate nearest neighbors: Towards removing the curse of dimensionality. In: Proc. of the 13th Annual ACM Symp. on the Theory of Computing. Dallas: Association for Computing Machinery, 1998. 604−613. [doi: 10.1145/276698.276876]

[3] Baluja S, Covell M. Learning to hash: Forgiving hash functions and applications. Data Mining and Knowledge Discovery, 2008, 17(3):402−430. [doi: 10.1007/s10618-008-0096-z]

[4] Weiss Y, Torralba A, Fergus R. Spectral hashing. In: Koller D, Schuurmans D, Bengio Y, Bottou L, eds. In: Koller D, Schuurmans D, Bengio Y, Bottou L, eds. Proc. of the Conf. on Neural Information Processing Systems (NIPS 2008). Vancouver: Curran Associates Inc., 2009. 1753−1760.

[5] Zhang D, Wang J, Cai D, Lu J. Self-Taught hashing for fast similarity search. In: Crestani F, Marchand-Maillet S, Chen HH, Efthimiadis EN, Savoy J, eds. Proc. of the 33rd Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval (SIGIR 2010). Geneva: Association for Computing Machinery, 2010. 18−25.

[6] Guttman A. R-Trees: A dynamic index structure for spatial searching. In: Yormark B, ed. Proc. of the Annual Meeting ACM Special Interest Group on Management of Data (SIGMOD'84). Boston: Association for Computing Machinery, 1984. 47−57. [doi: 10.1145/971697.602266]

[7] Bentley J. Multidimensional binary search trees used for associative searching. Communications of the ACM, 1975,18(9):509−517. [doi: 10.1145/361002.361007]

[8] Weber R, Schek H, Blott S. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In: Gupta A, Shmueli O, Widom J, eds. Proc. of the Int'l Conf. on Very Large Data Bases. New York: Morgan Kaufmann Publishers, 1998. 194−205.

[9] Broder AZ. On the resemblance and containment of documents. In: Proc. of the '97 Int'l Conf. on Compression and Complexity of Sequences. Positano: IEEE, 1997. 21−29.

[10] Datar M, Immorlica N, Indyk P, Mirrokni V. Locality-Sensitive hashing scheme based on p-stable distributions. In: Snoeyink J, Boissonnat JD, eds. Proc. of the 20th ACM Symp. on Computational Geometry. Brooklyn: Association for Computing Machinery, 2004. 253−262. [doi: 10.1145/997817.997857]

[11] Ramsak F, Markl V, Fenk R, Zirkel M, Elhardt K, Bayer R. Integrating the UB-tree into a database system kernel. In: Abbadi AE, Brodie ML, Chakravarthy S, Dayal U, Kamel N, Schlageter G, Whang KY, eds. Proc. of the 26th Int'l Conf. on Very Large Data Bases. Cairo: Morgan Kaufmann Publishers, 2000. 263−272.

[12] Tao Y, Yi K, Sheng C, Kalnis P. Quality and efficiency in high dimensional nearest neighbor search. In: Çetintemel U, Zdonik SB, Kossmann D, Tatbul N, eds. Proc. of the ACM SIGMOD Int'l Conf. on Management of Data (SIGMOD 2009). Providence: Association for Computing Machinery, 2009. 563−576. [doi: 10.1145/1559845.1559905]

[13] Salakhutdinov R, Hinton G. Semantic hashing. Int'l Journal of Approximate Reasoning, 2009,50(7):969−978. [doi: 10.1016/j.ijar.2008.11.006]

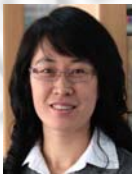
[14] Charikar M. Similarity estimation techniques from rounding algorithms. In: Agarwal P, Borodin A, Buss S, eds. Proc. of the 34th Annual ACM Symp. on Theory of Computing. Montréal: Association for Computing Machinery, 2002. 380−388. [doi: 10.1145/509907.509965]

[15] Tan PN, Steinbach M, Kumar V. Introduction to Data Mining. Boston: Addison Wesley, 2005.

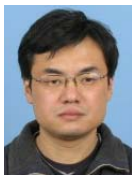
[16] Joachims T. Training linear SVMs in linear time. In: Proc. of the 12th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. Philadelphia: Association for Computing Machinery, 2006. 217−226. [doi: 10.1145/1150402.1150429]

[17] Chang CC, Lin CJ. LIBSVM: A library for support vector machines. 2001. http://www.csie.ntu.edu.tw/~cjlin/libsvm [doi: 10.1145/1961189.1961199]

**袁培森**(1980－),男,河南淮阳人,博士,主要研究领域为 Web 数据管理和数据挖掘,大规模数据并行计算.



**王晓玲**(1975－),女,博士,教授,CCF 会员,主要研究领域为 XML 数据管理,Web 服务技术.



**沙朝锋**(1976－),男,博士,讲师,CCF 会员,主要研究领域为数据管理,数据挖掘,机器学习.

**周傲英**(1965－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为数据密集型计算的数据管理,数据管理服务,WEB 搜索和挖掘,数据流,数据挖掘.