



研究与开发

基于加权自学习散列的高维数据最近邻查询算法

彭聪, 钱江波, 陈华辉, 董一鸿

(宁波大学信息科学与工程学院, 浙江 宁波 315211)

摘要: 因为查询和存储具有高效性, 学习型散列逐渐被应用于解决最近邻查询问题。学习型散列将高维数据转化成二进制编码, 并使得原始高维空间中越相似的数据对应二进制编码的汉明距离越小。在实际应用中, 每次查询都会返回许多与查询点汉明距离相同而编码互不相同的数据。如何对这些数据进行排序是一个难题。提出了一种基于加权自学习散列的近邻查找算法。实验结果表明, 算法能够高效地对具有相同汉明距离的不同编码进行重排序, 加权排序后查询的 $F1$ 值约是原来的 2 倍并优于同系算法, 时间开销可比直接计算原始距离进行排序降低一个数量级。

关键词: 最近邻查询; 学习型散列; 加权自学习; 高维数据

中图分类号: TP391

文献标识码: A

doi: 10.11959/j.issn.1000-0801.2017100

Nearest neighbor search algorithm for high dimensional data based on weighted self-taught hashing

PENG Cong, QIAN Jiangbo, CHEN Huahui, DONG Yihong

College of Information Science and Engineering, Ningbo University, Ningbo 315211, China

Abstract: Because of efficiency in query and storage, learning hash is applied in solving the nearest neighbor search problem. The learning hash usually converts high-dimensional data into binary codes. In this way, the similarities between binary codes from two objects are conserved as they were in the original high-dimensional space. In practical applications, a lot of data which have the same distance from the query point but with different code will be returned. How to reorder these candidates is a problem. An algorithm named weighted self-taught hashing was proposed. Experimental results show that the proposed algorithm can reorder the different binary codes with the same Hamming distances efficiently. Compared to the naive algorithm, the $F1$ -score of the proposed algorithm is improved by about 2 times and it is better than the homologous algorithms, furthermore, the time cost is reduced by an order of magnitude.

Key words: nearest neighbor search, learning hash, weighted self-taught, high-dimensional data

收稿日期: 2017-02-07; 修回日期: 2017-03-31

通讯作者: 钱江波, qianjiangbo@nbu.edu.cn

基金项目: 国家自然科学基金资助项目 (No.61472194, No.61572266); 浙江省自然科学基金资助项目 (No.LY16F020003)

Foundations Items: The National Natural Science Foundation of China (No.61472194, No.61572266), Zhejiang Provincial Natural Science Foundation of China (No.LY16F020003)



1 引言

最近邻查询 (nearest neighbor search, NNS), 指在给定数据集中返回与查询对象 q 距离最近 (相似) 的数据对象的问题。最近邻查询在数据挖掘 (data mining, DM)^[1]、大数据背景下信息检索 (information retrieval, IR)^[2]、模式识别 (pattern recognition, PR)^[3] 以及推荐系统^[4,5] 等许多领域中是一个基础研究问题, 在实际应用中往往转化为求解近似最近邻 (approximate nearest neighbor, ANN) 问题。随着数据的爆炸式增长、数据类型的多样化以及高维化, 设计高效的索引结构和查询算法成为众多研究者关注的热点问题。

学习型散列^[6-13] 因其查询和存储的高效性, 被广泛应用于解决最近邻查询问题。学习型散列方法将高维数据转化为二进制编码, 基于汉明距离对数据进行查询。计算两个二进制串的汉明距离是高效的, 但是由于汉明距离的取值是离散的, 且只能取整数, 同时受限于数据二进制编码的长度, 返回结果中与查询对象具有相同汉明距离而编码不同的对象可能很多, 很难进一步区分。对于候选集中这些对象, 如何根据其与查询点的距离进行重排序是一个重要且亟待解决的问题。传统的方法是通过计算查询点候选集中数据的原始距离来解决, 由于数据维度高、数据量大, 导致计算开销过大。本文提出了一种基于加权自学习的散列算法, 面向高维数据最近邻查找, 解决了现有学习型散列方法面临的具有相同汉明距离但二进制编码不同的数据难以进一步区分距离的问题。

2 相关工作

在解决数据最近邻查找问题上, 主要通过构建高效的索引结构来加快查询的效率。现有的方法大致可以分为 KD-tree^[14]、R-tree^[15] 等基于空间划分的索引方法和基于局部敏感散列 (locality sensitive hashing, LSH)^[16-20] 的索引方法以及基于机器学习

的学习型散列方法。这些方法分别适用于不同应用场景, 基于空间划分的方法能够较好地适用于低维数据的近邻查找, 在数据维度较高时可能出现“维度灾难”问题。LSH 方法是将原始数据空间中的两个相邻数据对象通过相同的函数映射后, 在新的数据空间中相似性得到保持。基于 LSH 的方法较好地适用于高维数据, 其散列函数通常采用随机映射的方式, 独立于具体的数据, 所以又被称为数据独立 (data-independent) 散列。为了保证查询的查准率, 基于 LSH 的方法通常需要构造多个散列函数和散列表, 使得时空开销增大。

学习型散列将高维数据转化为紧凑的二进制编码, 数据在原始空间中的相似性得到保持, 且编码长度往往较短, 通过计算汉明距离来衡量数据之间的相似性是十分高效的。学习型散列所构造的散列函数依赖于具体的数据学习得到, 能更好地体现出数据的分布和语义特征, 所以学习型散列方法又被称为数据依赖 (data-dependent) 散列。近年来, 涌现出了许多优秀的学习型散列方法。根据学习模型是否利用样本的监督信息, 可以分为无监督模型、半监督模型和监督模型。无监督的模型有各向同性散列 (isotropic hashing, IsoH)^[6]、迭代量化 (iterative quantization, ITQ)^[7]、可扩展图散列 (scalable graph hashing, SDH)^[8] 等, 主要思想是通过类似降维将原始空间中的高维数据最终转化为二进制编码, 使得相似的数据转化为二进制编码后之间的汉明距离较小, 并以此为目标函数学习得到散列函数。在某些情况下, 人们能够获取到少量数据的类别信息, 又称为“标签”。标签信息的加入能够在一定程度上改善无监督模型的性能。这种在训练过程中同时使用带标签数据和不带标签数据来学习的模型被称为半监督模型^[9]。而在监督式学习下, 每组训练数据都有一个明确的标识或结果。比较经典的监督模型有最小损失散列 (minimal loss hashing, MLH)^[10]、核监督散列 (kernels supervised hashing, KSH)^[11]、深度监督散列 (deep

supervised hashing, DSH)^[12]、基于离散监督图的列采样散列(column sampling based discrete supervised hashing, CSDSH)^[13]等。监督学习模型在大部分情况下相比无监督或者半监督的学习模型学习到的结果更优,但是在实际应用中,标签的获取常常需要很大的人工成本。上述学习型散列方法都是通过汉明距离来衡量数据之间的相似性,每次查询都会得到许多与查询对象具有相同汉明距离而不同二进制编码的对象。如何进一步区分这些对象与查询点的距离,据笔者所知,仅有很少的解决方案。QsRank^[21]是基于PCA的散列方法,给每个比特分配权重并且定义了一个得分函数来衡量映射后所得邻居的置信度,返回的结果按置信度得分排序。WhRank^[22]对二值化之前的实数向量的每一维进行权重求解,然后再将数据转化为二进制编码求解加权汉明距离,从而对候选集进行重排序。QsRank只适用于基于PCA的散列方法,WhRank严重依赖于数据的分布假设,数据必须服从均匀分布,所以以上两种方法都存在局限性。目前,普遍采用的方法是通过计算它们与查询点的原始距离来对候选集进行重排序。

3 加权自学习散列算法

3.1 算法目标

本文提出了一种加权自学习散列算法,算法

采用自学习散列(self-taught hashing, STH)^[23]框架将原始数据转化为二进制编码并学习得到散列函数。当查询对象到来时,首先通过学习得到的散列函数将其转化为二进制编码,然后再计算其与数据集中各数据的二进制编码的汉明距离,并返回满足特定距离要求的候选集。由于候选集中存在许多与查询对象具有相同的汉明距离而编码不同的数据,这些数据与查询对象的真实距离是有差异的,而通过汉明距离却无法区分。对于高维数据,如果去比较原始数据,计算的开销太大。本文算法的目标就是快速准确地对与查询点具有相同汉明距离、不同比特编码的数据进行排序。

3.2 加权自学习散列

如图1中左侧框图所示,自学习散列方法的学习阶段包含两个步骤,第一步为无监督的学习,目的是将原始空间中的高维数据 $\mathbf{x}$ 转化为一个较短的二进制编码 $\mathbf{y}$ 。假设 $\mathbf{X}$ 是大小为 m 的数据集合,数据维度为 n ,那么该数据集可表示为:

$$\mathbf{X} = \begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mn} \end{pmatrix} \quad (1)$$

自学习散列通过度量学习方式将 $\mathbf{x}_i$ 转化为长度为 l bit的二进制编码 $\mathbf{y}_i \in \{0,1\}^l$,则 $\mathbf{Y}$ 可以表示为 $m \times l$ 的矩阵:

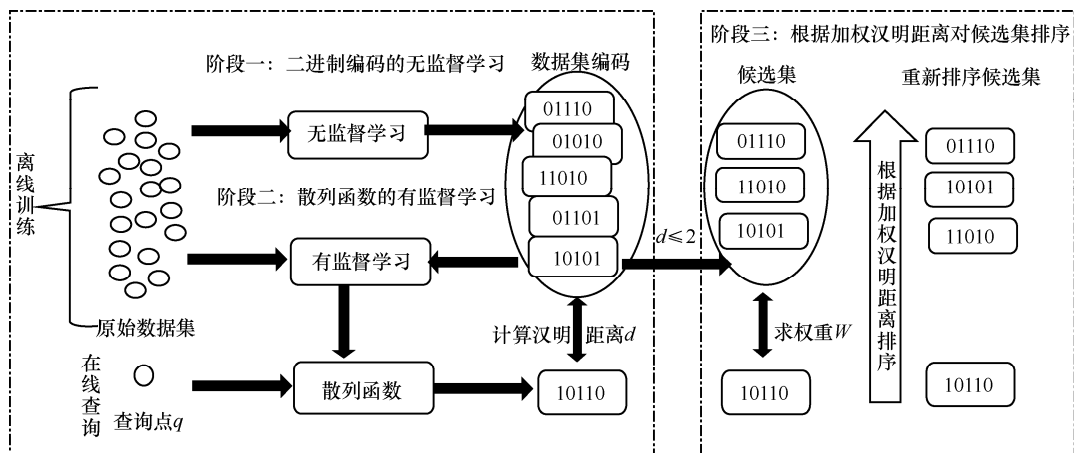


图1 加权自学习散列框架



$$\mathbf{Y} = \begin{pmatrix} y_{11} & \cdots & y_{1l} \\ \vdots & \ddots & \vdots \\ y_{m1} & \cdots & y_{ml} \end{pmatrix} \quad (2)$$

第二步是有监督的学习, 将第一步得到的二进制编码的每一位 $y_{i1}, \dots, y_{il} \in \{0, 1\}$ 作为类标签, 采用机器学习的方法训练出 l 个分类器并将其作为散列函数 $h_1, h_2, \dots, h_l$ 。查询数据 $\mathbf{x}$ 到来时, 通过散列 $H(\mathbf{x}) = (h_1(\mathbf{x}), h_2(\mathbf{x}), \dots, h_l(\mathbf{x}))$ 将其转化为二进制编码 $y_{i1}, \dots, y_{il} \in \{0, 1\}$ 。之所以称为自学习散列方法, 是因为自学习散列中数据的标签并非真实的标签, 只是将二进制编码人为地作为数据标签使用, 所以自学习散列整体上属于无监督的范畴。

3.3 查询自适应权重

自学习散列是一个开放的框架, 其学习过程的两个步骤, 可以根据具体的应用环境和需求采用不同的策略。例如参考文献[24]中在原始STH的无监督学习阶段改用LSH对高维数据进行二进制编码, 主要目的在于减小原始STH无监督学习阶段的度量学习^[22]计算开销。和大多数学习型散列方法一样, 自学习散列包括上述提到的对STH的改进方法, 依然面临与查询对象汉明距离相同而编码不同的候选对象难以进一步区分的问题。如图1中右侧框图所示, 假设查询对象 $\mathbf{q}$ 的二进制编码为“10110”, $\mathbf{q}$ 与 $\mathbf{o}_1(10101)$ 、 $\mathbf{o}_2(11010)$ 、 $\mathbf{o}_3(01110)$ 的汉明距离都是2, 通过汉明距离无法进一步明确地区分。假设数据的比特编码长度为 $H(\cdot) = (h_1, h_2, \dots, h_l)$, 那么返回的候选结果中与其汉明距离为 d 的不同编码有 C_l^d , 这些具有相同汉明距离但不同编码的数据很难进一步区分, 传统的方法需要计算数据的原始距离。在磁盘上保存原始数据需要额外的空间开销, 将高维数据加载到内存中计算原始距离开销也很大。为了解决这个问题, 本文对数据二进制编码的每一位赋予权重 w_i , 通过计算加权汉明距离来对候选集进行重排

序。原始数据的权重可以通过很多方法直接计算得到, 且与查询点无关。学习型散列将原始数据转化为二进制编码, 数据信息遭到损失且不同比特之间是相互独立的, 所以通过原始数据求权重的方法并不适用。在学习型散列中, 要想通过加权汉明距离对候选集进行重排序, 就需要根据每次查询得到的候选集计算当前查询的权重向量, 该权重与查询点相关, 故称为查询自适应权重。

3.4 权重求解原理

假设 $H(\cdot) = (h_1, h_2, \dots, h_l)$ 是由 l 个相互独立的散列函数组成的散列函数向量, $H(\cdot)$ 将高维数据 $\mathbf{q}$ 和 $\mathbf{o}$ 分别映射为长度为 l 的二进制编码 $H(\mathbf{o})$ 和 $H(\mathbf{q})$, 从左到右第 k bit 对应的二进制编码为 $h_k(\cdot)$, 则 $\mathbf{q}$ 与 $\mathbf{o}$ 的汉明距离^[23]表示为:

$$D(H(\mathbf{q}), H(\mathbf{o})) = \sum_{k=1}^l \|h_k(\mathbf{q}) - h_k(\mathbf{o})\| \quad (3)$$

假设查询对象 $H(\mathbf{q})$ 各比特的权重为 $\mathbf{W} = (w_1, w_2, \dots, w_l)$, 二者的加权汉明距离^[22]表示为:

$$D_w(H(\mathbf{q}), H(\mathbf{o})) = \sum_{k=1}^l w_k \|h_k(\mathbf{q}) - h_k(\mathbf{o})\| \quad (4)$$

用 $p(H(\mathbf{q}) = H(\mathbf{o}))$ 表示数据对象 $\mathbf{q}$ 和 $\mathbf{o}$ 被映射为相同编码的概率, 由于散列函数之间是相互独立的, 所以两个数据二进制编码相等的概率就等价于它们每一位被映射为相同编码的概率的乘积^[22], 于是有:

$$p(H(\mathbf{q}) = H(\mathbf{o})) = \prod_k p(h_k(\mathbf{q}) = h_k(\mathbf{o})) \quad (5)$$

两个数据之间的距离或者相似度与其被映射相同编码的概率相关, 两个数据越相似, 其加权汉明距离越小, 其编码相同的概率就越大。为了量化求解各比特的权重, 本文提出可信度的概念来衡量一个散列函数将相似的数据对象映射为相同比特(0/1)的能力。实际上, 每个散列函数都可以看作一个二分类器, 二进制编码看作类的标号。对于一个散列函数, 如果大多数相似的数据

与查询点有相同的标签,则此散列函数可信度更高。可信度越高的散列函数,如果对应比特上两个数据的二进制编码不同,则它们相似的概率越小,即对应的加权汉明距离就越大,应该对此散列函数对应的二进制编码位上分配较大的权重。例如,假设 q 的二进制编码为101, o_1 、 o_2 的二进制编码为001、111。从左往右数, q 与 o_1 的二进制编码在第一比特位置不同, q 与 o_2 在第二比特不同。如果第一比特的可信度高于第二比特位,那么就认为 q 与 o_2 更相似,则有 $D_w(H(q), H(o_2)) < D_w(H(q), H(o_1))$,比特的权重应该满足 $w_1 > w_2$ 。

定理 1 一个散列函数将相似的数据对象映射为相同比特的概率越大,那么该散列函数的可信度就越高,对应该比特的权重越大。

证明:上述定理可以根据信息熵来证明。在信息论^[26]中,熵是不确定性的度量。一般来说,某个指标的信息熵越小,表明指标值的变异程度越大,提供的信息量越多,在综合评价中所能起到的作用也越大,其权重也就越大。相反,某个指标的信息熵越大,表明指标值的变异程度越小,提供的信息量也越少,在综合评价中所起到的作用也越小,其权重也就越小。

假设 X 是一个离散随机变量,即它的取值范围 $R=\{x_1, x_2, \dots, x_n\}$ 是有限可数的。设 $p_i=P\{X=x_i\}$,即 X 取 x_i 的概率。 X 的熵定义为^[26]:

$$E(X) = -\sum_{i=1}^n p_i \lg p_i \quad (6)$$

假设查询对象 q 二进制编码为 $H(q)$,候选集中的数据对象在第 k bit上编码与 q 编码相同的概率是 p_k ,则不相同的概率为 $1-p_k$,根据式(6)可知二进制编码第 k bit的信息熵为:

$$E_k(H(q)) = -p_k \lg p_k - (1-p_k) \lg (1-p_k) \quad (7)$$

计算出各个比特位的信息熵为 $E_1, E_2, \dots, E_k, \dots, E_l$,根据信息论中熵权理论^[27]可以通过信息熵计算各比特的权重:

$$w_k = 1 - E_k / (l - \sum_{k=1}^l E_k) \quad (8)$$

从式(8)可以看出, E_k 查询点与候选集数据对应比特编码的取值只有相同或不同两种情况,满足二元信源熵函数。如式(7)所示,对应二元信源熵函数的曲线如图2所示。

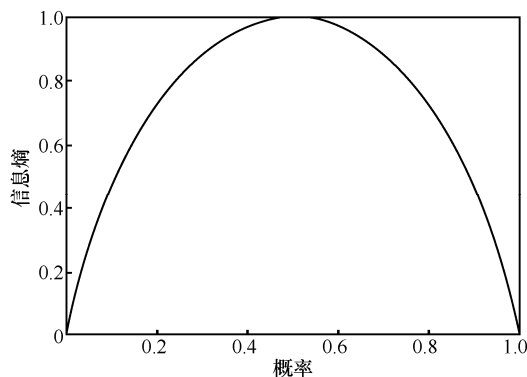


图2 二元信源熵函数曲线

因为候选集中大多数数据与查询对象在对应比特上的编码是相同的,假设相同的概率 $p_k > 1/2$ 。由图2可知,当概率 $p_k > 1/2$ 时,各比特位的信息熵 E_k 随着 p_k 的增大而减小。再由式(5)可知: E_k 越小, w_k 就越大。综上,所以可知 p_k 越大,对应 E_k 越小,则 w_k 就越大。即散列函数将相似的数据对象映射为相同比特的概率越大,那么该散列函数的可信度就越高,对应该比特的权重越大。

3.5 求解步骤

以 C_k 表示查询对象 q 与候选集中数据的比特编码在第 k bit上映射为相同编码的次数, $S(h_k(q)=h_k(o_i))$ 表示 $h_k(q)$ 与 $h_k(o_i)$ 是否相等。如果相等,则 $S(h_k(q)=h_k(o_i))=1$,否则为0。于是可以得到:

$$C_k = \sum_{i=1}^N S(h_k(q)=h_k(o_i)) \quad (9)$$

查询对象 q 与候选集中数据的二进制编码第 k 位相等的概率为 p_k , N 表示候选集的大小,则:

$$p_k = C_k / N \quad (10)$$

w_k 与 p_k 呈近似单调的线性关系,所有在实际



应用中, 为了简化计算, 可用 $(p_1, p_2, \dots, p_k)$ 来近似表示 $(w_1, w_2, \dots, w_k)$ 。例如, 给定一个查询对象 q , 假设自学习散列得到的二进制编码为 0110, 返回的与其汉明距离为 2 的对象为 o_1 、 o_2 、 o_3 、 o_4 、 o_5 , 二进制编码为 1010、1100、1010、0011、1100, 由定理 1 可知 p_k 越大, 则 w_k 越大, 它们之间的对应关系见表 1。

表 1 C_k 与 p_k 、 w_k 对应关系

bit	q	o_1	o_2	o_3	o_4	o_5	C_k	p_k	w_k
bit ₁	0	1	1	1	0	1	1	1/5	1/5
bit ₂	1	0	1	0	0	1	2	2/5	2/5
bit ₃	1	1	0	1	1	0	3	3/5	3/5
bit ₄	0	0	0	0	1	0	4	4/5	4/5

算法 1 求解权重向量

输入 q (查询数据的二进制编码)

$O(o_1, \dots, o_N)$

(根据自学习散列获得候选集 O , 然后依次计算各比特位权重)

输出 $w = (w_1, w_2, \dots, w_k)$

初始化 $w_1, w_2, \dots, w_k$ to 0

For $k = 1, \dots, \text{length}(q)$

$C_k = \text{count}(h_k(q) = h_k(O))$

If $\text{sum}(C_k) > 0$

$w_k = C_k / (\text{sum}(C_k) \times N)$

4 实验与分析

4.1 实验设置

本文的算法使用 MATLAB 编程实现, 实验运行机器配置 CPU 为 Intel Core i5-4950 2 Duo 3.30 GHz。原始 STH 在无监督学习阶段采用拉普拉斯特征映射 (Laplacian eigenmaps)^[28] 将高维数据映射为低维实数向量, 然后通过均值阈值法转化为二进制编码。拉普拉斯特征映射可以反映出数据内在的流形结构, 通过 KNN 来构建数据对象之间的相似性关系矩阵, 这样避免了计算全局相似性带来的

时空开销。而在监督学习阶段, 则采用由 LIBLINEAR^[29] 提供的线性 SVM (support vector machine, 支持向量机) 来训练分类器, 得到 l 个散列函数。为了能够更好地适应高维数据和非线性可分问题, 本文对部分 SVM 参数进行了调整, 采用高斯核函数来替代线性核函数, 其他参数均采用默认设置。区别于基本的 STH, 把监督学习阶段采用核化的 SVM 来训练的自学习方法称为 STHs。在接下来的实验中, 本文将对 STHs 和 STH 的性能, 从而了解采用不同核函数的 SVM 对查询性能的影响。本文的主要贡献在于提出了一种加权的思想对候选集进行重排序, 对应 STHs 和 STH 基础上进行加权重排序的算法为 WSTHs 和 WSTH。

为了验证加权算法的先进性, 本文还与同系算法进行了对比。传统的 STH 首先对数据集构建 KNN 相似性矩阵, 并通过矩阵分解和特征值求解等操作将高维数据转化为二进制比编码, 时间和空间复杂度高。而参考文献[24]中所提出的改进算法是在 STH 的基础上, 在无监督学习阶段采用 LSH 将高维数据随机投影并通过阈值函数转化为二进制编码, 所以称为 LSTH。因为在查询得到候选集后, 同样面临着具有相同汉明距离而不同编码的数据无法进一步区分的问题。同时, 为了体现算法的可扩展性, 将提出的加权思想应用于 LSTH 上, 称为 WLSTH, 并对比二者的性能差异。

二进制编码的长度 (code length) 最小取值为 4 bit, 每次以 4 bit 的尺度增长, 最大编码长度取 128 bit。散列编码的增长使得查找的范围进一步缩小, 有利于提高查准率, 但是会降低查全率。所以, 为了提高查全率, 汉明距离取值 d 随编码长度的增大而增大, 编码长度每增加 4 bit, 汉明距离取值加 1。假设对于每个查询数据, 要求返回与其汉明距离不大于 d 的数据, 得到的候选集的大小为 N , 为了体现加权重排序后的效果, 所以将加权算法取候选集重排序后的前 $N/2$ 、 $N/3$ 的数据作为最终的返回结果, 对 LSTH、STH 和

STHs 得到的候选集, 为了体现随机性, 采用随机的方式抽取同样数量的数据作为最终的返回结果集。

4.2 实验数据集选取

本文分别在 Reuters21578、20Newsgroups、TDT2 这 3 个公开数据集上进行实验, 实验中对数据集进行停用词删除、词干提取以及计算 TF-IDF 等预处理, 最终用稀疏矩阵来表示。

Reuters21578 是 1987 年路透社新闻专线上的文章集合构成的语料库, 去除了出现在不同分类中的文章, 并且只保留文章数最多的 10 个分类, 最终选择出来的文章数为 7 285 篇, 训练集包含 5 225 篇(72%)文档, 测试集为 2 057 篇(28%), 数据的维度大小为 17 024。

TDT2 语料库收集自新闻快线、广播节目以及电视节目, 去除出现在不同类别中的文章, 将剩余数目最多的 30 个类别保留, 最终得到 9 394 篇, 通过随机选择的方法得到训练集大小为 5 644 篇(60%), 测试集为 3 750 篇(40%), 数据的维度大小为 33 878。

20Newsgroups 中总共有 18 846 篇来自 20 个类别的文章, 根据时间对数据集进行划分, 最终得到大小分别为 11 314 篇(60%)的训练集和 7 532 篇(40%)的测试集合, 数据的维度大小为 25 714。

4.3 数据集评估

4.3.1 查询结果的 F1 值

原始的数据集中包含样本的类别信息, 如果返回的结果与查询对象属于同一个类别, 则认为这个返回结果是正确的。最后, 通过查准率 P (precision) 和查全率 R (recall) 的调和平均数 $F1$ 值来评价不同算法的性能。 $F1$ 值被广泛应用在信息检索领域, 用来衡量检索和分类性能, 它同时兼顾了模型的查准率和查全率, $F1$ 值越大, 说明检索和分类效果越好。假定对一个大规模数据集进行检索或分类, 可根据真实类别与检索结果或预测类别的组合划分为 4 组: 检索到的相似数

据称为真正例 (true positive, TP), 检索到的不相似数据称为假正例 (false positive, FP), 相似但是没有检索到的数据称为假反例 (false negative, FN), 不相似且没有被检索到的数据称为真反例 (true negative, TN)。根据参考文献[30]则有:

$$P = TP / (TP + FP) \quad (11)$$

$$R = TP / (TP + FN) \quad (12)$$

$$F1 = 2P \times R / (P + R) \quad (13)$$

4.3.2 时间效率

本文所提出的加权自学习散列算法主要是在原始自学习散列的基础上引入了加权的概念。如, WTHs 在 STHs 的基础上对二进制数据不同比特赋予了不同的权重, 对候选集中难以区分的数据进行了重排序, 这一操作需要消耗额外的时间。对候选集进行重排序最直接、最准确的办法是通过计算查询数据与候选集中的数据的原始距离, 计算欧氏 (Euclidean) 距离是普遍采用的方法。两个 n 维向量 $a(x_{11}, x_{12}, \dots, x_{1n})$ 与 $b(x_{21}, x_{22}, \dots, x_{2n})$ 的欧氏距离计算式如下:

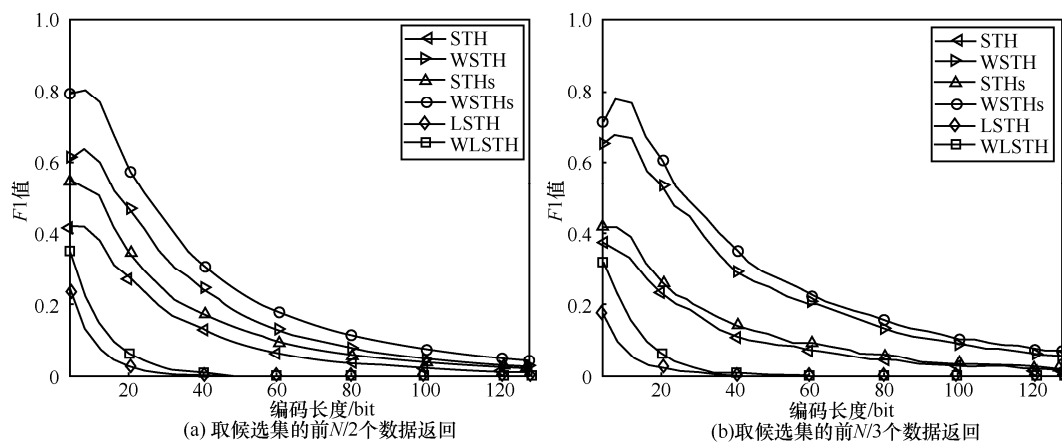
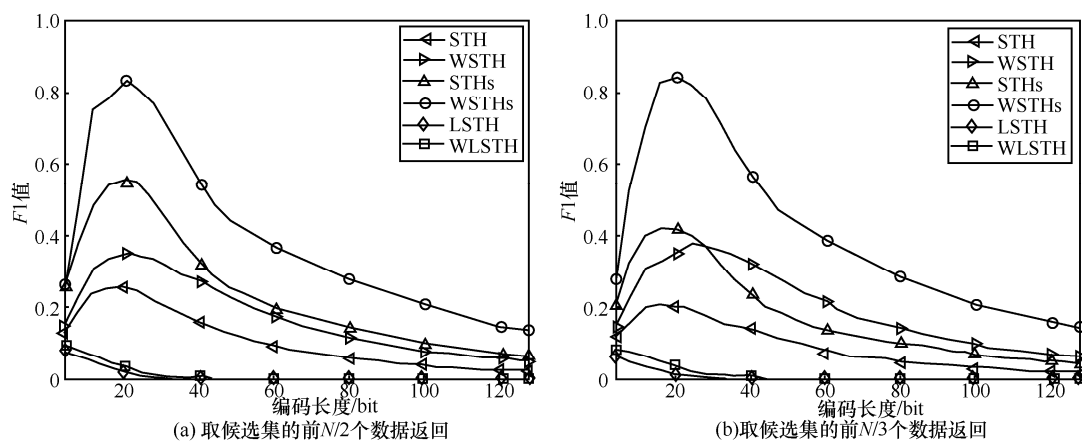
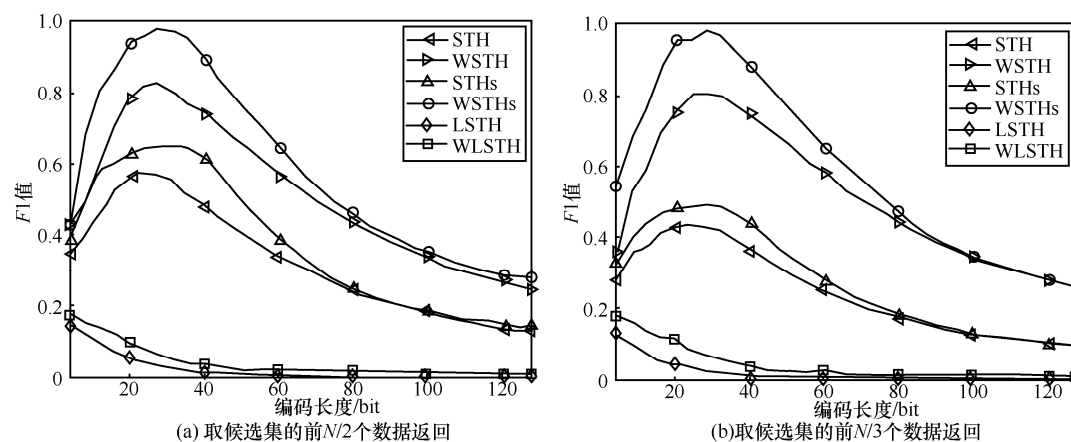
$$d_{ab} = \sqrt{\sum_{k=1}^n (x_{1k} - x_{2k})^2} \quad (14)$$

假设 N 为每次查询返回的满足距离要求的候选集的大小, l 为数据的二进制编码的长度, 从式 (14) 可以得出, 计算欧氏距离的时间复杂度为 $O(Nn)$ 。WTHs 算法中对候选集进行排序消耗的额外时间是根据候选集求权重计算加权汉明距离, 时间复杂度为 $O(Nl)$ 。二进制编码的长度却比原始数据小很多, 即 $l \ll n$, 那么, 在整体的时间消耗上, WTHs 要远小于直接计算欧氏距离。试验中所涉及的其他加权算法基于同样的原理, 所以后续实验只以 WTHs 为例。

4.4 实验结果及分析

4.4.1 查询的质量

3 个数据集上 $F1$ 值随编码长度的变化曲线如图 3~图 5 所示。可以看到, 相比原始的 STH, STHs

图3 Reuters21578数据集上 $F1$ 值随编码长度变化曲线图4 20Newsgroups数据集上 $F1$ 值随编码长度变化曲线图5 TDT2数据集上 $F1$ 值随编码长度变化曲线

采用高斯核函数重写 SVM 后 $F1$ 值有一定程度的提高,说明采用高斯核函数的 SVM 更适合于高维数据的分类,但是核函数相比线性函数计算需要消耗更多的时间。

相比 STH、STHs,改进后的加权自学习散列算法 WSTH 及 WSTHs 查询的 $F1$ 值提高了一倍,说明了加权算法的先进性。对比 3 个数据集上分别取前 $N/2$ 、 $N/3$ 的候选集作为返回结果时发现,

返回的结果越少时, $F1$ 值差距越大, 原因在于通过加权重排序后, 排序越靠前的数据与查询点越相似, 也证明了所提出的加权思想是正确有效的。同时可以看到, 不同的数据集映射为不同的编码长度时 $F1$ 值也是不一样的, 从图 3~图 5 中可以看出, 3 个数据集编码长度分别取 8 bit、24 bit、32 bit 时 $F1$ 最大, 这体现了学习型散列方法散列函数学习的数据依赖性。不同的数据集有不同的分布, 数据各个维度的重要性也不一样, 加权自学习散列算法能够对不同的数据集学习得到最优的编码长度, 并且通过对候选集进行排序返回更为精确的结果。从实验结果可以看出, 编码的长度并不是越长越好。因为编码长度越长, 查准率虽然有所提高, 但是具有相同汉明距离的不同编码种类增加了, 使得查全率下降。而编码的长度太短时, 大量的数据会被映射为相似的编码, 使得查准率降低。

另外, 从图 3~图 5 中可以看出, STH 和 STHs 查询的质量要优于 LSTH。主要原因在于 LSH 需要更多的散列函数来提高查准率, 这又使得落入同一个散列桶中的数据减少, 导致查全率降低, 所以整体的查询质量不如 STH 和 STHs。对应的加权算法 WSTH 及 WSTHs 也优于 WLSTH, 说明了本文所提出算法的先进性。值得注意的是, 应用本文所提出加权思想改进后的 LSTH 算法

(WLSTH 算法) 的 $F1$ 值相比原始的 LSTH 也有一定程度的提高, 说明所提加权算法具有一定的可扩展性。LSTH 算法性能的提高需要解决查全率随着散列函数的增加而降低和对散列后的实数进行阈值化所导致数据分布不均匀的问题。LSH 通常可以采用构建多个散列表来提高查全率, 但是应用于 LSTH 中时由于后续需要通过二进制编码来训练散列函数, 只能构建一张散列表。

4.4.2 STH、STHs 与 LSTH 查全率和查准率比较

3 个数据集上, STH、STHs 与 LSTH 查全率和查准率比较曲线如图 6~图 8 所示。可以看出, 要想达到相同的查准率, LSTH 比 STH、STHs 需要更多的散列函数。但是, LSTH 通过 SVM 来训练散列函数, 需求的散列函数多, 增加了时空开销。同时, 从图 6~图 8 查全率和查准率的实验结果可以看出, 随着编码长度的增加查准率是上升的, 而查全率却在降低, LSTH 的查全率很快接近 0, 所以 LSTH 只限于查找一个近似近邻却不能很好地适用于 KNN 查找。

4.4.3 查询的性能

以 WTHs 算法为例, 根据前面时间效率的理论分析可知, WTHs 对候选集重排序消耗的时间远小于直接计算原始数据的欧氏距离所消耗的时间, 下面通过实验来验证, 结果见表 2。

从表 2 可以看出, 通过 WTHs 算法对候选集进

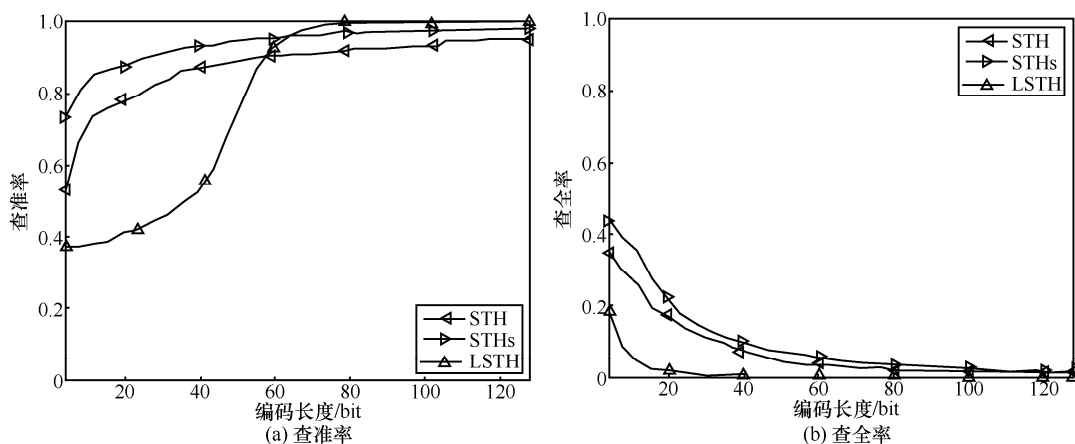


图 6 Reuters21578 数据集上 STH、STHs 与 LSTH 查询质量比较

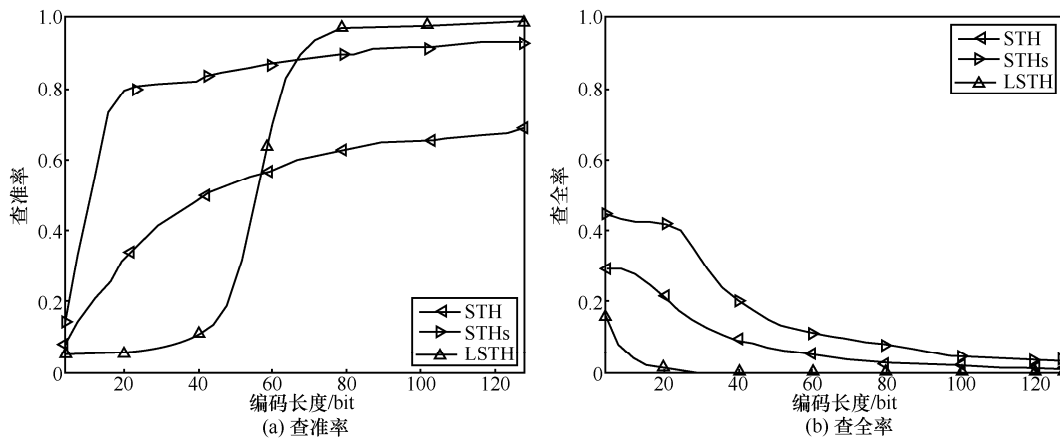


图7 20Newsgroups数据集上STH、STHs与LSTH查询质量比较

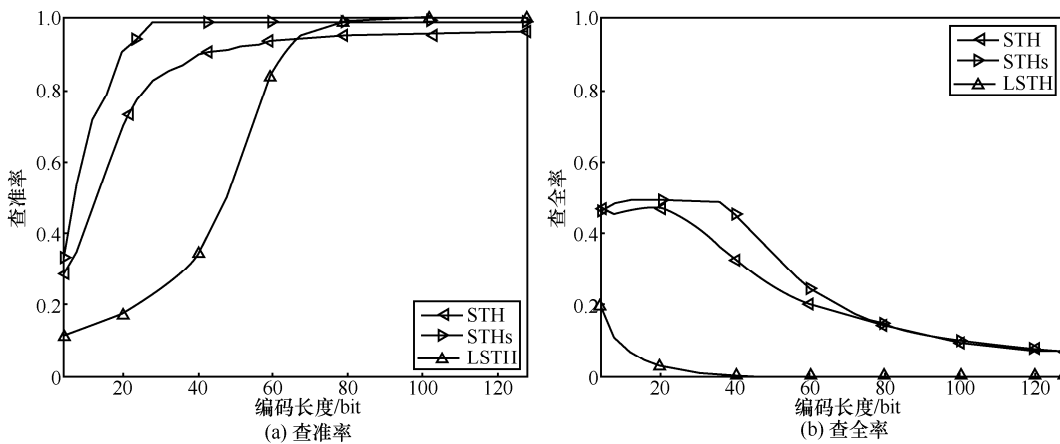


图8 TDT2数据集上STH、STHs与LSTH查询质量比较

表2 WTHs与直接计算欧氏距离时间对比(单位: s)

编码长度	Reuters21578 数据集		20Newsgroups 数据集		TDT2 数据集	
	WSTHs	欧氏	WTHs	欧氏	WTHs	欧氏
4	35.428 6	739.105 4	73.172 1	1 353.835 6	260.610 7	4 665.874 5
8	32.559 2	570.788 6	29.594 3	600.568 3	108.156 5	1 947.040 1
12	21.704 7	484.824 2	26.362 6	478.037 5	67.570 3	1 225.862 0
16	21.347 3	354.663 8	21.305 6	442.278 6	50.856 9	920.451 3
20	14.465 1	286.251 0	26.604 4	433.643 9	45.930 0	767.526 7
24	12.248 7	222.789 1	21.476 8	404.780 9	49.711 9	691.038 2
28	12.171 8	205.063 9	23.263 4	414.414 6	38.512 1	617.036 0
32	10.007 8	172.408 3	19.495 6	391.412 2	35.790 2	504.167 3
36	10.338 0	167.010 9	21.275 7	375.782 5	29.896 0	412.498 6
40	8.725 2	135.351 7	20.138 6	333.844 0	25.688 7	346.459 0
44	8.076 7	126.817 7	18.878 6	292.501 3	22.997 9	295.564 6
48	7.850 7	122.781 0	15.489 9	245.141 9	23.666 2	265.888 2
52	6.751 4	96.049 3	14.120 7	207.877 7	21.322 1	235.231 9
56	5.910 4	99.728 5	11.549 9	179.772 8	18.795 4	218.865 2
60	6.908 4	97.791 1	10.841 5	161.983 0	17.524 7	192.378 7
64	5.677 4	81.916 7	10.758 2	141.673 4	17.307 4	189.965 1

行重排序所消耗的时间远远小于通过计算欧氏距离所消耗的时间, 后者的时间开销是 WTHs 算法的 10~20 倍。对于需要快速时间响应的应用来说, WTHs 无疑是更优的选择。

4.4.4 其他性能比较

空间上, 由于 WTHs 算法将数据转化为一串较短的二进制编码, 计算权重并不消耗太多额外的空间, 而要计算数据的原始距离则必须保留原始数据, 空间消耗大。直接计算原始距离的方法是计算相似性和距离最为直接有效的方法。为了衡量 WTHs 和直接计算欧氏距离在性能上的表现, 除了对比 WTHs 和直接计算欧氏距离对数据进行排序后的查准率, 本文还对比了两种方法进行 KNN 分类的准确性。即在查询点类别标签未知

的情况下, 通过查询返回的 KNN 结果可以推断查询对象所属的类别, 查准率越高, 说明所取得的 K 近邻越准确。KNN 分类的准确率 (accuracy) 定义为: 查询得到正确分类的查询点数量与总的查询点数量的比值。前者是通过判断查找到的候选集的准确性来衡量算法的性能, 后者则是通过判断查询点的类别来衡量。假设 N 为每次查询返回的满足距离要求的候选集的大小, K 取 $N/2$, WTHs 与直接计算欧氏距离查询的查准率和 KNN 分类的准确率如图 9~图 11 所示。

可以看出, 当数据的二进制编码长度较小时, WTHs 算法相比直接计算欧氏距离对数据进行排序查询的查准率高。原因在于欧氏距离对数据的各个维度同等对待, 而 WTHs 则更能体现出数据

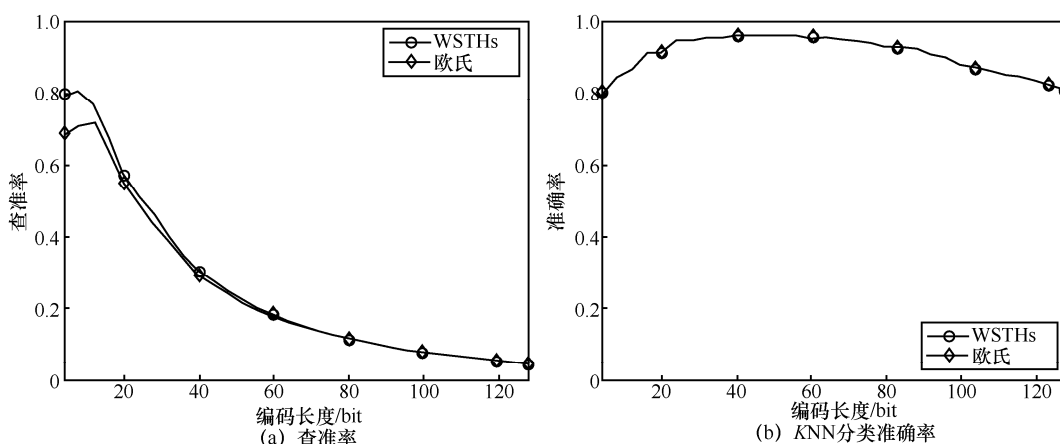


图9 Reuters21578数据集上 WTHs 与计算欧氏距离查询质量比较

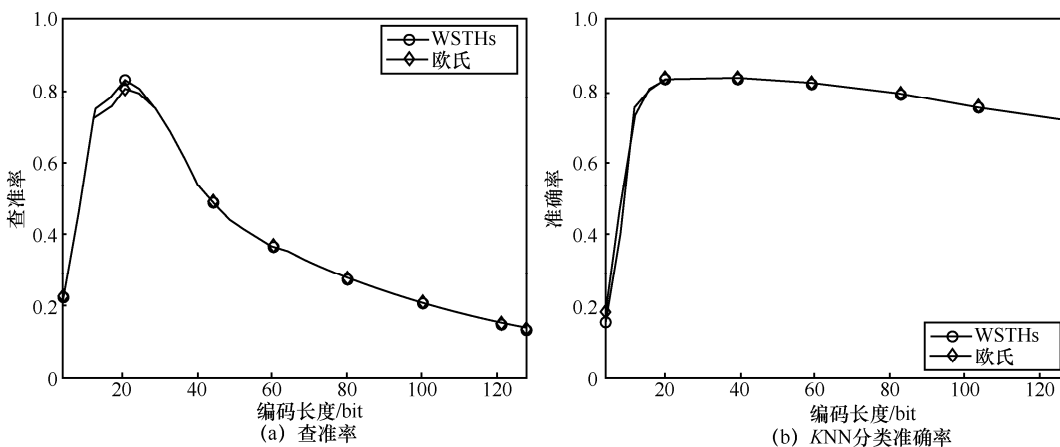


图10 20Newsgroups数据集上 WTHs 与计算欧氏距离查询质量比较

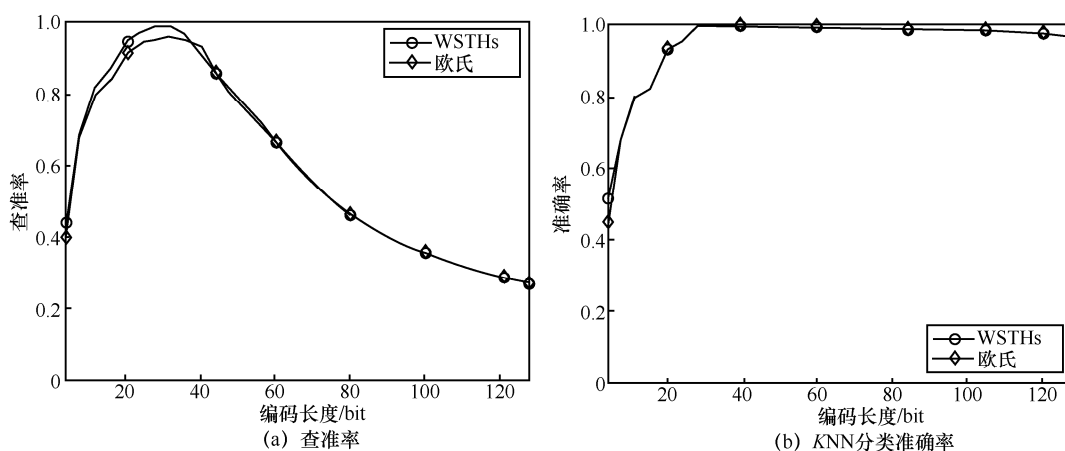


图 11 TDT2 数据集上 WTHs 与计算欧氏距离查询质量比较

不同维度重要性差异。通过 WTHs 所得结果集来进行 KNN 分类与直接计算欧氏距离的准确性相差不大。WTHs 相比传统的计算欧氏距离这类线性扫描方法,不仅在时间上表现出了优异的性能,而且准确性也十分接近。

从上面的几组实验可以看出,加权自学习散列算法在解决最近邻等相关问题上取得了优越的性能。相比原始的算法,查准率和查全率都有很大的提高,查询的质量与线性扫描方法十分接近,效率更高。与同系算法的比较体现了其先进性和可扩展性。

5 结束语

大多数学习型散列方法将高维数据从原始空间映射到低维的汉明空间之后,由于降维和量化过程带来的误差损失,使得具有相同汉明距离的数据对象与查询对象的相似性难以进一步区分。而数据的不同维度携带的信息量是不一样的,对应转化后的二进制编码的各位的重要性也是不一样的。为得到更为精确的返回结果,在基于自学习散列框架的基础上,根据每次查询得到的候选集对应位置上比特编码相同的概率来衡量散列函数的可信度,从而对比特位赋予不同的权重,最终返回加权排序后的结果。本文从理论方面进行了分析并通过实验验证了加权自学习散列方法的正确性和高效性。

参考文献:

- [1] HE J, LIU W, CHANG S F. Scalable similarity search with optimized kernel hashing[C]//The 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), July 25-28, 2010, Washington, DC, USA. New York: ACM Press, 2010: 1129-1138.
- [2] 吴军. 大数据和机器智能对未来社会的影响[J]. 电信科学, 2015, 31(2): 7-16.
WU J. Big data, machine intelligence and their impacts to the future World [J]. Telecommunications Science, 2015, 31(2): 7-16.
- [3] ZHANG D, WANG F, SI L. Composite hashing with multiple information sources[C]//The International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, July 24-28, 2011, Beijing, China. New York: ACM Press, 2011: 225-234.
- [4] 尤海浪, 钱锋, 黄祥为, 等. 基于大数据挖掘构建游戏平台个性化推荐系统的研究与实践[J]. 电信科学, 2014, 30(10): 27-32.
YOU H L, QIAN F, HUANG X W, et al. Research and practice of building a personalized recommendation system for mobile game platform based on big data mining [J]. Telecommunications Science, 2014, 30(10): 27-32.
- [5] 徐雅斌, 刘超, 武装, 等. 基于用户兴趣和推荐信任域的微博推荐[J]. 电信科学, 2015, 31(1): 13-20.
XU Y B, LIU C, WU Z, et al. Micro-blog recommendation based on user interests and recommendation trust domain [J]. Telecommunications Science, 2015, 31(1): 13-20.
- [6] KONG W, LI W J. Isotropic hashing[J]. Advances in Neural Information Processing Systems, 2012(2): 1646-1654.
- [7] GONG Y, LAZEBNIK S, GORDO A, et al. Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval[J]. IEEE Transactions on Pattern Analysis Machine Intelligence, 2013(35): 2916-2929.
- [8] JIANG Q Y, LI W J. Scalable graph hashing with feature transformation[C]//International Conference on Artificial Intelli-

- gence, July 25-31, 2015, Buenos Aires, Argentina. New York: ACM Press, 2015: 2248-2254.
- [9] WANG J, KUMAR S, CHANG S F. Semi-supervised hashing for large-scale search[J]. IEEE Transactions on Pattern Analysis Machine Intelligence, 2012, 34(12): 2393-2406.
- [10] NOROUZI M E, FLEET D J. Minimal loss hashing for compact binary codes[C]//International Conference on Machine Learning, June 28-July 1, 2011, Bellevue, Washington, USA. [S.l.:s.n.], 2011: 353-360.
- [11] LIU W, WANG J, JI R, et al. Supervised hashing with kernels[C]//2012 IEEE Conference on Computer Vision and Pattern Recognition(CVPR), June 16-21, 2012, Kingston, USA. New Jersey: IEEE Press, 2012: 2074-2081.
- [12] LI W J, WANG S, KANG W C. Feature learning based deep supervised hashing with pairwise labels[J]. arXiv preprint arXiv:1511.03855, 2015.
- [13] KANG W C, LI W J, ZHOU Z H. Column sampling based discrete supervised hashing[C]//AAAI, February 12-17, 2016, Phoenix, Arizona, USA. [S.l.:s.n.], 2016: 1230-1236.
- [14] BENTLEY J L. K-d trees for semi dynamic point sets[C]//Symposium on Computational Geometry, June 7-9, 1990, Berkley, California, USA. New York: ACM Press, 1990: 187-197.
- [15] AHMED M, MAHAR K, ABDELKADER H, et al. Combining R-Tree and B-Tree to enhance spatial queries[C]//International Conference on Computer Theory and Applications, Oct 8-11, 2013, Alexandria, Egypt. [S.l.:s.n.], 2013.
- [16] SHRIVASTAVA A, LI P. Asymmetric LSH (ALSH) for sublinear time maximum inner product search (MIPS)[J]. Advances in Neural Information Processing Systems, 2014(3): 2321-2329.
- [17] ANDONI A, INDYK P, LAARHOVEN T, et al. Practical and optimal LSH for angular distance[J]. Computer Science, 2015.
- [18] QIAN J, ZHU Q, CHEN H. Multi-granularity locality-sensitive bloom filter[J]. IEEE Transactions on Computers, 2015, 64(12): 3500-3514.
- [19] HUANG Q, FENG J, ZHANG Y, et al. Query-aware locality-sensitive hashing for approximate nearest neighbor search[J]. Proceedings of the VLDB Endowment, 2015, 9(1): 1-12.
- [20] LIU Y, CUI J, HUANG Z, et al. SK-LSH: an efficient index structure for approximate nearest neighbor search[J]. Proceedings of the VLDB Endowment, 2014, 7(9): 745-756.
- [21] SHUM H Y, ZHANG L, ZHANG X. QsRank: query-sensitive hash code ranking for efficient ϵ -neighbor search[C]//2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 16-21, 2012, Kingston, Rhode Island, USA. New York: ACM Press, 2012: 2058-2065.
- [22] ZHANG L, ZHANG Y, TANG J, et al. Binary code ranking with weighted hamming distance[C]//2013 IEEE Conference on Computer Vision and Pattern Recognition(CVPR), June 23-28, 2013, Portland, Oregon, USA. New York: ACM Press, 2013: 1586-1593.
- [23] ZHANG D, WANG J, CAI D, et al. Self-taught hashing for fast similarity search[C]//International ACM SIGIR Conference on Research and Development in Information Retrieval, July 19-23, 2010, Geneva, Switzerland. New York: ACM Press, 2010: 18-25.
- [24] 袁培森, 沙朝锋, 王晓玲, 等. 一种基于学习的高维数据 c-近似最近邻查询算法[J]. 软件学报, 2012, 23(8): 2018-2031.
- YUAN P S, SHA Z F, WANG X L, et al. A high dimensional data c-approximate nearest neighbor query algorithm based on learning[J]. Journal of Software, 2012, 23(8): 2018-2031.
- [25] 殷良鹰. 非线性度量学习算法研究[D]. 北京: 北京理工大学, 2016.
- YIN L Y. Research of nonlinear metric learning algorithm[D]. Beijing: Beijing Institute of Technology, 2016.
- [26] 姜丹. 信息论与编码[M]. 北京: 中国科学技术大学出版社, 2004.
- JIANG D. Information theory and coding[M]. Beijing: University of Science & Technology China Press, 2004.
- [27] 魏书堤, 姜小奇. 一种利用信息熵确定属性权重的模糊单因素评价方法[J]. 计算机工程与科学, 2010, 32(7): 93-94.
- WEI S D, JIANG X Q. A fuzzy single factor evaluation method based on entropy to determine the weight of attributes [J]. Computer Engineering & Science, 2010, 32 (7): 93-94.
- [28] BELKIN M, NIYOGI P. Laplacian eigenmaps for dimensionality reduction and data representation[J]. Neural Computation, 2003, 15(6): 1373-1396.
- [29] FAN R E, CHANG K W, HSIEH C J, et al. Liblinear: a library for large linear classification[J]. Journal of Machine Learning Research, 2008, 9(9): 1871-1874.
- [30] 周志华. 机器学习[M]. 北京: 清华大学出版社, 2016.
- ZHOU Z H. Machine learning[M]. Beijing: Tsinghua University Press, 2016.

[作者简介]



彭聪 (1990-), 男, 宁波大学硕士生, 主要研究方向为大数据、数据挖掘。



钱江波 (1974-), 男, 博士, 宁波大学教授, 主要研究方向为数据处理与挖掘、逻辑电路设计、多维索引与查询优化。

陈华辉 (1964-), 男, 博士, 宁波大学教授, 主要研究方向为数据处理与挖掘、云计算。

董一鸿 (1969-), 男, 博士, 宁波大学教授, 主要研究方向为大数据、数据挖掘和人工智能。