



研究与开发

基于列表监督的 Hash 排序算法

杨安邦, 钱江波, 董一鸿, 陈华辉

(宁波大学信息科学与工程学院, 浙江 宁波 315211)

摘 要: Hash 学习技术目前被广泛应用于大规模数据的相似性查找中, 其通过将数据转化成二进制编码的形式, 同时提高查找速度和降低存储代价。目前, 大多数 Hash 排序算法通过比较数据在欧氏空间和海明空间的排序一致性来构造损失函数, 然而, 在海明空间的排序过程中, 因为海明距离是离散的整数值, 可能存在多个数据点共享相同的海明距离, 这样就无法准确地排序。针对这一问题, 将编码后的数据切分成几个长度相同的子空间, 并为每个子空间设置不同的权重, 比较时, 再根据不同的子空间权重来计算海明距离。实验结果表明, 与其他 Hash 学习算法相比, 本文算法能够有效地对海明空间中的数据进行排序, 并提高查询的准确性。

关键词: Hash 学习; 相似性查找; Hash 排序; 子空间权重

中图分类号: TP391

文献标识码: A

doi:10.11959/j.issn.1000-0801.2019072

A ranking hashing algorithm based on listwise supervision

YANG Anbang, QIAN Jiangbo, DONG Yihong, CHEN Huahui

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

Abstract: Recently, learning to hash technology has been used for the similarity search of large-scale data. It can simultaneously increase the search speed and reduce the storage cost through transforming the data into binary codes. At present, most ranking hashing algorithms compare the consistency of data in the Euclidean space and the Hamming space to construct the loss function. However, because the Hamming distance is a discrete integer value, there may be many data points sharing the same Hamming distance result in the exact ranking cannot be performed. To address this challenging issue, the encoded data was divided into several subspaces with the same length. Each subspace was set with different weights. The Hamming distance was calculated according to different subspace weights. The experimental results show that this algorithm can effectively sort the data in the Hamming space and improve the accuracy of the query compared with other learning to hash algorithms.

Key words: learning to hash, similarity search, ranking hashing, subspaces with different weights

收稿日期: 2018-10-31; 修回日期: 2019-04-12

通信作者: 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), The Natural Science Foundation of Zhejiang Province of China (No.LY16F020003)

1 引言

最近邻查询作为数据检索中使用最广泛的技术,一直以来都是人们研究的热点。一般来说,最近邻查询就是根据数据之间的相似性,从数据库中寻找与查询对象最相似的数据。其目前在信息检索、模式识别、数据挖掘等众多领域都有广泛的应用。然而,随着现代社会的不断发展、信息技术的日益更新,人们每天产生的数据不管从量上还是维度上都呈现爆炸式的增长,如何通过最近邻查询技术获取自己最想要的数据是一个亟需解决的问题。

目前,主要有两种方法来解决最近邻查询问题。一种是基于树的索引查找方法,其主要代表有红黑树、kd树、R树^[1]等。这种方法一般在低维空间中表现良好,然而,当维度急剧上升时,会产生“维度灾难”等问题,其搜索效率接近于线性搜索。第二种是目前比较热门的基于Hash的查找方法^[2-4],Hash查找方法具有存储代价低、搜索速度快的优点,但它存在的问题是将数据点转化为二进制编码后,在查找时,需要比较数据点与查询点的海明距离,而海明距离是离散的整数值,可能存在多个点与查询点的海明距离都相同,这样就无法获取最相似的数据。

本文针对这个问题,在特定的Hash排序算法基础框架上,提出了一种新的方案,解决了海明空间排序过程中具有相同海明距离的数据无法进一步区分的问题。

2 相关工作

Hash方法因为其较好的检索速度和较低的存储代价,近些年来受到越来越多研究者的关注,它的核心思想是将高维空间中的特征向量转化为海明空间中的低维二进制串,同时能保留原始数据之间的相似关系。

目前,Hash方法主要被分为两大类:数据独立和数据依赖的方式。数据独立的方法主要代表是局部敏感Hash(LSH)^[5]及其变体,通过随机映射的方式将数据点转化为编码,而数据依赖的方法就是Hash学习的方法,其通过机器学习机制将数据转化为编码。数据依赖的Hash方法分为3种:无监督Hash、有监督Hash和半监督Hash。无监督方式在生成Hash码的过程中没有用到已知的标签信息,比较经典的方法有谱Hash(SH)^[6]、自学习Hash(STH)^[7]、迭代量化(ITQ)^[8]、主成分分析Hash(PCAH)^[9]等。

有监督和半监督的Hash方式都利用了监督信息(语义标签等)来生成Hash码,它们的区别在于半监督Hash只使用了部分的标签信息。有监督Hash目前比较经典的方法有最小损失Hash(MLH)^[10]、核监督Hash(KSH)^[11]、监督离散Hash(SDH)^[12]、线性判别分析Hash(LDAH)^[13]。而半监督的Hash方法目前较少,主要代表是半监督Hash(SSH)^[14],未来半监督方法是值得关注的研究点,因为随着数据收集和存储技术的飞速发展,收集大量未标记的示例已相当容易,而获取大量有标记的示例则相对较为困难,因为获得这些标记可能需要耗费大量的人力物力,而半监督的方式只用其中一部分标记就能达到很好的效果。

尽管目前上述Hash方法都取得了比较好的结果,但它们都很少考虑到在实际搜索任务中的排序信息。因为一般在通过搜索引擎搜索的过程中,返回来的查询结果都是根据与查询点的相关程度从上到下依次排列的。基于此,近些年来出现了一些基于排序的Hash方法^[15-19],它们通过保留三元组监督信息(比如列生成Hash(CG H)^[20]、海明距离度量学习(HDML)^[21]等)或者列表监督信息(列表监督Hash(RSH)^[22]、保排序Hash(RPH)^[23]、深度语义排序Hash(DSRH)^[24]、离散语义排序Hash(DSeRH)^[25]等)来学习生成



Hash 码。三元组监督方式是建立一个三元组 (x_q, x_i, x_j) , x_q 表示查询点, x_i 表示与查询点相似的数据, x_j 表示与查询点不相似的数据, 通过建立损失函数让编码后的 x_q 与 x_i 之间更相似, x_q 与 x_j 之间更不相似。列表监督方式是根据与查询点的相似度在欧氏空间与在海明空间中对数据库中的所有点进行排序, 让每个点的排名在两个空间中尽量保持一致。

虽然基于列表监督的 Hash 方法充分利用了排序信息, 但是某种特殊情况, 如图 1 所示: x_q 编码为 1101, x_1 编码为 0101, x_2 编码为 1111, x_3 编码为 1001, x_4 编码为 1100, 显然 x_1 、 x_2 、 x_3 、 x_4 与 x_q 的海明距离都为 1, 这样就无法对每个点更精确地排序, 造成排序误差。

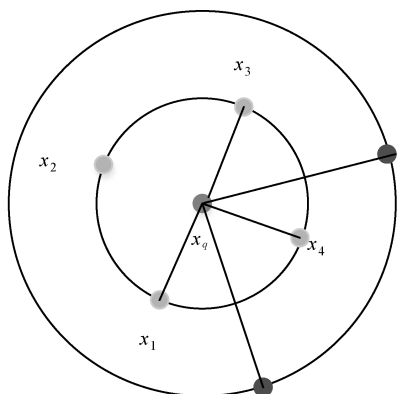


图 1 数据点与查询点海明距离分布示例

针对这个问题, 本文提出一种基于列表监督的子空间加权 Hash 方法, 通过将编码分割并赋予权重的方式, 对数据库中的所有点都能准确地排序, 并取得了比较好的搜索精确性。

3 基于列表监督的 Hash 排序算法

3.1 算法描述

(1) 计算查询点与数据集中所有数据点的欧氏距离, 并根据大小进行排序, 得到每个点的真实排名情况。

(2) 将查询点和数据集中所有数据点转化成二进制编码, 并将编码分成长度相同的子空间,

每个子空间赋予不同的权重, 计算海明距离, 得到在海明空间中的排名情况。

(3) 根据损失函数最小化数据点在不同空间中的排序不一致性。

如图 2 所示, x_1 、 x_2 、 x_3 、 x_4 表示数据库中的 4 个数据点, q 表示查询点, 在欧氏空间下计算 q 与 4 个点的距离, 根据距离大小进行排名, 得到排名列表 $R_1=(x_1, x_2, x_3, x_4)=(1, 2, 3, 4)$, 同时将所有数据点进行编码, 再计算 q 与其他 4 个数据点的海明距离, 同样得到排名列表 $R_2=(x_1, x_2, x_3, x_4)=(1, 2, 3, 4)$, 最后比较 R_1 、 R_2 , 构造损失函数, 让 R_1 、 R_2 尽量保持一致。

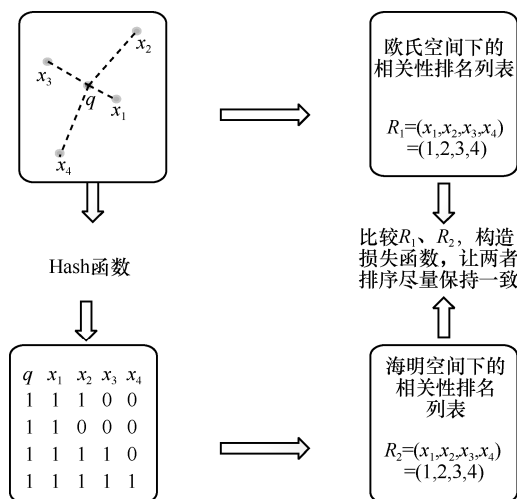


图 2 RLSH 算法

3.2 欧氏空间中的排序

假设数据集中有 n 个数据点, 表示为 $\mathbf{x} = \{x_1, x_2, \dots, x_N\} \in R^{d \times N}$, d 是数据的特征维度, 此外, 还有一个查询集 $\mathbf{Q} = \{q_1, q_2, \dots, q_M\} \in R^{d \times M}$ 。对于查询集中的任一查询点 q_j , 可以计算其与数据集 $\mathbf{x}$ 中每个点的欧氏距离, 得到相关性排序列表, 记作:

$$r(q_j, \mathbf{x}) = (r_1^j, r_2^j, \dots, r_N^j) \quad (1)$$

其中, $r_i^j \in [1, N]$, 表示数据样本 x_i^j 与查询点的相似性排名, 如果 $r_m^j < r_n^j$, 则表示 x_m 比 x_n 更相似于查询点 q_j 。

r_m^j 可以直接通过计算查询点与数据集中点的欧氏距离并进行排序得到。以此作为监督信息表示数据集中的每个数据与查询数据的真实相似度排名。

3.3 海明空间中的排序

本文的目标是得到 Hash 函数 $f(\cdot)$ 来生成二进制编码 $h: R \rightarrow \{-1, 1\}$, 定义 Hash 函数映射关系如下:

$$h_i = f(x_i) = \text{sgn}(Wx_i) \quad (2)$$

其中, $W \in R^{Bd}$, B 表示编码长度。

通常两个编码之间的海明距离计算定义为:

$$\begin{aligned} \text{Ham}(h_q, h_i) &= \frac{1}{4} \|h_q - h_i\|^2 \\ &= \frac{1}{2} (B - h_q^T h_i) \end{aligned} \quad (3)$$

但这样会存在一个问题, 因为海明距离是离散的整数值, 在计算查询点与数据集中点的海明距离时, 可能存在多个数据点共享相同的海明距离, 这样就无法准确地对每个点进行排名。为了解决这个问题, 重新定义海明距离为:

$$\begin{aligned} \text{Ham}(h_q, h_i) &= \sum_{s=0}^{\frac{B}{\sigma}-1} 2^s \cdot \frac{1}{4} \|h_{q_s} - h_{i_s}\|^2 \\ &= \sum_{s=0}^{\frac{B}{\sigma}-1} 2^s \cdot \frac{1}{2} (\sigma - h_{q_s}^T h_{i_s}) \\ &= \sum_{s=0}^{\frac{B}{\sigma}-1} 2^{s-1} (\sigma - h_{q_s}^T h_{i_s}) \end{aligned} \quad (4)$$

其中, 参数 σ 表示每个子空间长度, $\frac{B}{\sigma}$ 即表示分割的子空间个数, 2^s 表示每个子空间的加权值, 这样加权能给每个子空间赋予的权重有明显的区分, 排在前列的子空间一般权重较大, 能够明显地将海明距离区分, 但随着 s 值的减小, 权重也进一步减小, 进一步细分出不同的数据点。同时这样加权也能表示更多的海明距离, 然后再根据加权后的海明距离计算数据集中每个点的排序信

息 R_m^j 为:

$$\begin{aligned} R_m^j &= 1 + \sum_{k=1}^N I(\text{Ham}(q_j, x_m^j) > \text{Ham}(q_j, x_k^j)) \\ &= 1 + \sum_{k=1}^N I(\text{Ham}(q_j, x_m^j) - \text{Ham}(q_j, x_k^j) > 0) \\ &= 1 + \sum_{k=1}^N I\left(\sum_{s=0}^{\frac{B}{\sigma}-1} 2^{s-1} (\sigma - h_{q_{js}}^T h_{x_{ms}^j}) - \sum_{s=0}^{\frac{B}{\sigma}-1} 2^{s-1} (\sigma - h_{q_{js}}^T h_{x_{ks}^j}) > 0\right) \\ &= 1 + \sum_{k=1}^N I\left(\sum_{s=0}^{\frac{B}{\sigma}-1} 2^{s-1} \cdot h_{q_{js}}^T (h_{x_{ks}^j} - h_{x_{ms}^j}) > 0\right) \end{aligned} \quad (5)$$

其中, $I(\cdot)$ 表示决策函数, 当括号内条件成立时为 1, 否则为 0。类似于在 RPH、Top-RSBC 等算法中, 可以用 Sigmoid 函数 $g(x) = \frac{1}{1 + \exp(-x)}$ 来近似

替换 $I(\cdot)$, 如图 3 所示, 因此, 每个点的排序信息 R_m^j 可重新表示为:

$$\begin{aligned} R_m^j &\approx 1 + \sum_{k=1}^N \frac{1}{1 + \exp(-(\text{Ham}(q_j, x_m^j) - \text{Ham}(q_j, x_k^j)))} \\ &= 1 + \sum_{k=1}^N \frac{1}{1 + \exp\left(-\left(\sum_{s=0}^{\frac{B}{\sigma}-1} 2^{s-1} \cdot h_{q_{js}}^T (h_{x_{ks}^j} - h_{x_{ms}^j})\right)\right)} \\ &= 1 + \sum_{k=1}^N \frac{1}{1 + \exp\left(\sum_{s=0}^{\frac{B}{\sigma}-1} 2^{s-1} \cdot h_{q_{js}}^T (h_{x_{ms}^j} - h_{x_{ks}^j})\right)} = \\ &= 1 + \sum_{k=1}^N \frac{1}{1 + \exp\left(\sum_{s=0}^{\frac{B}{\sigma}-1} 2^{s-1} \cdot \text{sgn}(W_s q_j)^T (\text{sgn}(W_s x_m^j) - \text{sgn}(W_s x_k^j))\right)} \\ &\approx 1 + \sum_{k=1}^N \frac{1}{1 + \exp\left(\sum_{s=0}^{\frac{B}{\sigma}-1} 2^{s-1} \cdot q_j^T W_s^T W_s (x_m^j - x_k^j)\right)} \end{aligned} \quad (6)$$

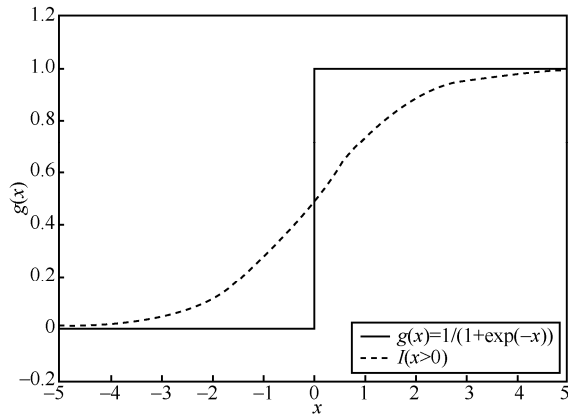


图3 用 Sigmoid 函数来近似代替决策函数 $I(\cdot)$

$$W = [W_0, W_1, W_2, \dots, W_s]^T \quad (7)$$

此外，平常通过搜索引擎查询数据时，返回的结果根据相关性从上到下依次排列，显然，最上面的数据与查询 q 最为相关，当其排错时，显然受到的惩罚更大，因此，可以定义根据单个查询点查询得到的排序损失为：

$$L(q, x_i) = \sum_{i=1}^N \frac{1}{2 \ln(1 + r_i)} (r_i - R_i)^2 \quad (8)$$

其中， $\frac{1}{2 \ln(1 + r_i)}$ 表示数据集中数据点 x_i 的排名权重，显然当 x_i 在欧氏空间中的真实排名越高（ r_i 的值越小）时， x_i 的权重越大（如图 4 所示）。

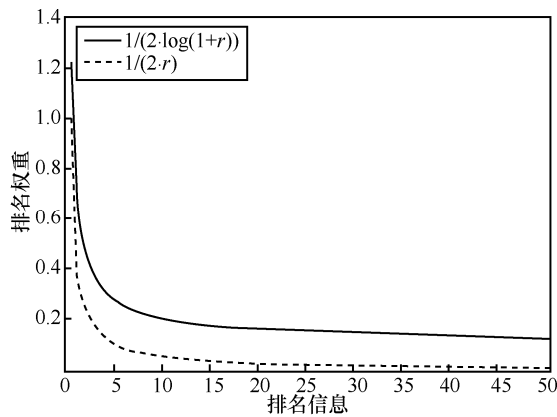


图4 两种不同的排名权重函数比较

则对于整个查询集，总的损失函数定义为：

$$L(Q, \chi) = \sum_{j=1}^M \sum_{i=1}^N \frac{1}{2 \log(1 + r_i^j)} (r_i^j - R_i^j)^2 + \frac{\lambda}{2} \|WW^T - I\|_F^2 \quad (9)$$

其中， $\frac{\lambda}{2} \|WW^T - I\|_F^2$ 表示正则项，参数 $\lambda > 0$ 来平衡排序损失和正则项， $WW^T - I$ 表示正交约束，使得学习到的每一位 Hash 函数不相关，最终通过对上述函数的优化，期望得到最优的 W 。

3.4 优化

通常由于 Hash 函数的离散性和非连续性，直接优化是比较困难的，这里通过前面的松弛函数 $\text{sgn}(x) = x$ 将目标函数替换成连续且可导函数，

得到梯度函数如式 (10)、式 (11)：

$$\frac{\partial L(Q, \chi)}{\partial W} = \sum_{j=1}^M \sum_{i=1}^N \frac{1}{\log(1 + r_i^j)} \cdot (R_i^j - r_i^j) \cdot \frac{\partial R_i^j}{\partial W} + 2\lambda W^T (WW^T - I) \quad (10)$$

$$\frac{\partial R_i^j}{\partial W} = \sum_{k=1}^N \frac{-\exp \left(\sum_{s=0}^{\frac{B-1}{\sigma}} 2^{s-1} \cdot q_j^T W_s^T W_s (x_i^j - x_k^j) \right)}{\left(1 + \exp \left(\sum_{s=0}^{\frac{B-1}{\sigma}} 2^{s-1} \cdot q_j^T W_s^T W_s (x_i^j - x_k^j) \right) \right)^2} \cdot \left(\sum_{s=0}^{\frac{B-1}{\sigma}} 2^{s-1} \cdot W_s \left((x_i^j - x_k^j) q_j^T + q_j (x_i^j - x_k^j)^T \right) \right) \quad (11)$$

基于得到的梯度函数，通过梯度下降法的更新规则：

$$W_{t+1} = W_t - \alpha \frac{\partial L(Q, \chi)}{\partial W} \quad (12)$$

整个过程如算法 1 所示。

算法 1 基于列表监督的 Hash 排序算法 (RLSH)

输入 训练数据集 $\chi = \{x_1, x_2, \dots, x_N\} \in R^{d \times N}$ ，

查询数据集 $Q = \{q_1, q_2, \dots, q_M\} \in R^{d \times M}$, 编码长度 B , 子空间长度 σ , 参数 λ 和 α 。

输出 二进制编码 h 和 Hash 函数 $f(\cdot)$

步骤 1 从训练集和查询集中, 根据式 (1) 计算得到真实排序列表。

步骤 2 随机初始化 W , 得到一个 $\{-1, +1\}^{Bd}$ 的矩阵。

步骤 3 重复步骤 4 和步骤 5。

步骤 4 根据式 (10) 和式 (11) 计算梯度。

步骤 5 根据式 (12) 进行梯度下降。

步骤 6 直到算法收敛或者达到最大迭代次数。

步骤 7 返回得到二进制编码 $h = C(f(x)), \forall x$ 。

3.5 时间复杂度分析

基于列表监督的 Hash 排序算法主要分为数据集的线下训练和线上测试两部分, 训练阶段中, 在应用梯度下降法的每次迭代过程中, 需要根据式 (10) 计算梯度, 其中式 (10) 中前半部分的时间复杂度为 $O(MN^2sdB)$, 后半部分的时间复杂度为 $O(d^2B)$, 总的时间复杂度为 $O(MN^2sdB + d^2B)$ 。

4 实验分析

实验运行机器配置 CPU 为 Intel Core i5-7500, 16 GB DDR3 内存, 在真实数据集上进行测试。本文用 MATLAB 语言实现了算法, 并与其他经典 Hash 学习方法如 RSH、RPH、SKLSH 等算法进行了对比。

4.1 实验数据集

本文分别在 NUS-WIDE、ImageNet、Uci-usps 这 3 个公开数据集上进行实验, 实验中先通过对数据进行零均值化等预处理, 然后通过特征提取和转化将图像数据集转化为矩阵形式以便于运算。3 个数据集介绍如下。

(1) NUS-WIDE 数据集

该数据集是由新加坡国立大学的媒体搜索实验室创建的网络图像数据集, 包括来自 Flickr

269 648 个图像数据以及 5 018 个唯一的关联标签、从数据集图像中提取的 6 种低级特征、可用于评估的 81 个概念的校验。基于这个数据集, 可以确定几个关于网络图像注释和检索的研究问题。基准测试结果表明, 可以从该数据集中学习模型, 以完成一般的图像检索算法研究。

(2) ImageNet 数据集

该数据集是一个计算机视觉系统识别项目, 是目前世界上图像识别最大的数据库。其中含有大约 1 500 万张图片、2.2 万种类别。

(3) Uci-usps 数据集

美国邮政手写数字数据集, 由 0~9 共 10 个类组成, 每个类有 1 100 个样本。数据的维度大小为 256。

4.2 评价指标

为了评价算法的性能, 使用了 NDCG (normalized discounted cumulative gain) 作为评价指标。NDCG 表示归一化折损累积增益, 常用于对排序的评价指标, 为了计算 NDCG, 首先要计算 DCG:

$$DCG_p = rel_1 + \sum_{i=2}^p \frac{rel_i}{1bi} \quad (13)$$

其中, p 表示根据查询点得到的排序列表中的数据个数, rel_i 表示返回的第 i 个数据点的相关程度, 与此同时, 将真实排序的 DCG 值记作 IDCg, 因此可以得到 NDCG 的值为:

$$NDCG_p = \frac{DCG_p}{IDCG_p} \quad (14)$$

实验中计算返回的前 K 个数据的 NDCG 值, 记为 $NDCG@K$, 这里的 K 取 50。

4.3 实验结果

不同数据集下返回的前 50 个数据 NDCG 值, 如图 5 所示。实验结果表明, 在 NUSWIDE 数据集上, RLSH 算法一般比其他的经典 Hash 学习方法好, 而在 Uci-usps 数据集上, 在编码



长度比较低时本文的算法有比较好的效果,但编码位数超过 28 时,效果被其他方法超越,这是由于长编码能表示更多的数据点,数据点之间的区分度更大,因此, NDCG 值可能没有其他的排序 Hash 方法高。同时,在 ImageNet 数据集上,除了 SKLSH 方法外,其他几种 Hash 排序方法效果基本一致。

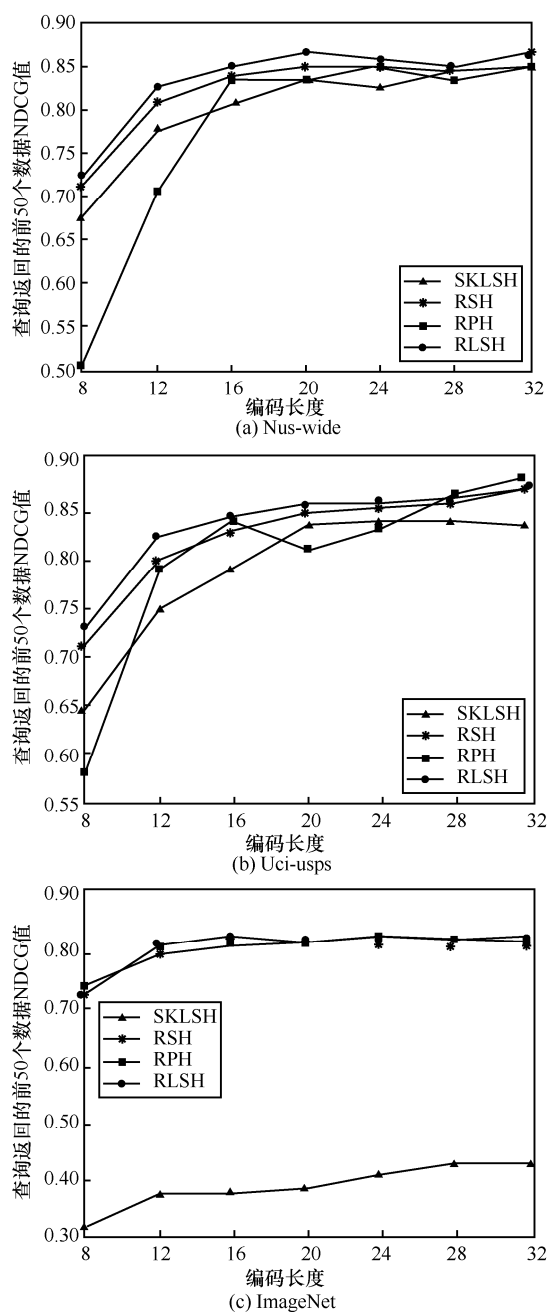


图5 不同数据集下返回的前 50 个数据 NDCG 值

5 结束语

本文提出了一种新的基于列表监督的加权 Hash 排序算法,针对海明距离比较过程中存在的缺点,通过将长编码分割成短编码,再对每段短编码进行加权的方式,有效地对数据集中的每个数据点进行准确的排序。另外,在面对非凸非连续的损失函数优化问题时,通过线性转化和函数的近似替换,最后通过梯度下降法,得到最优的 Hash 函数。通过在 3 个大规模数据集上的实验,证明了算法的有效性和优越性。

RLSH 算法虽然实现了较好的排序精确性,但目前还存在的问题是算法时间复杂度较高,需要的训练时间较长,未来可进一步考虑改进或替换海明空间中的排序函数,从而提高训练速度。

参考文献:

- [1] GUTTMAN A. R-trees: adynamic index structure for spatial searching[J]. ACM Sigmod Record, 1984, 14(2): 47-57.
- [2] SONGD, LIU W, JI R. Top rank supervised binary coding for visual search[C]//2015 IEEE International Conference on Computer Vision, Dec 13-16, 2015, Santiago, Chile. Piscataway: IEEE Press, 2015.
- [3] PARK Y, CAFARELLA M, MOZAFARI B. Neighbor-sensitive Hashing[M]. New York: ACM Press, 2016.
- [4] WANG J, ZHANG T, SONG J, et al. A survey on learning to hash[J]. IEEE Transaction on Pattern Analysis and Machine Intelligence, 2016(99): 1.
- [5] DATAR M, IMMORLICA N, INDYK P, et al. Locality-sensitive hashing scheme based on p-stable distributions[C]//Twentieth Symposium on Computational Gemetry, Jun 8-11, 2004, New York, USA. New York: ACM Press, 2004: 253.
- [6] WEISS Y, TORRALBA A, FERGUS R. Spectral Hashing[C]//Neural Information Processing System, Dec 8-11, 2008, British Columbia, Canada. New York: ACM Press, 2008.
- [7] ZHANGD, WANG J, CAI D, et al. Self-taught hashing for fast similarity search[C]//33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, July 19-23, 2010, IGO, Geneva Switzerland. New York: ACM Press, 2010.
- [8] GONG Y, LAZEBINK 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 and Machine Intelligence, 2013, 135(12): 2916-2919.
- [9] LIN R S, ROSS D A, YAGNIK J. Spec hashing: similarity preserving algorithm for entropy-based coding[C]//Computer Vision and Pattern Recognition, Jun 13-18, 2010, San Francisco, CA, USA. [S.l.: s.n], 2010.
- [10] NOROUZI M E, FLEET D J. Minimal loss hashing for compact binary codes[C]//The 28th International Conference on Machine Learning, June 28-July 2, 2011, Washington, USA. Piscataway: IEEE Press, 2011.
- [11] CHANG S F, JIANG Y G, JI R, et al. Supervised hashing with kernels[C]//Computer Vision and Pattern Recognition, June 16-21, 2012, Providence, RI, USA. Piscataway: IEEE Press, 2012.
- [12] SHEN F, SHEN C, LIU W. Supervised discrete hashing[C]//Computer Vision and Pattern Recognition, June 8-10, 2015, Boston, USA. Piscataway: IEEE Press, 2015.
- [13] STRECHA C, BRONSTEIN A, BRONSTEIN M, et al. LDAHash: improved matching with smaller descriptors[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2011, 34(1): 66-78.
- [14] WANG J, KUMA R S, CHANG S F. Semi-supervised hashing for large-scale search[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012, 34(12): 2393-2406.
- [15] LIN G, SHEN C, WU J. Optimizing ranking measures for compact binary code learning[Z]. 2014.
- [16] SONG D, LIU W, MEYER D, et al. Rank preserving hashing for rapid image search[C]//2015 Data Compression Conference, April 7-9, 2015, Snowbird, UT, USA. Piscataway: IEEE Press, 2015.
- [17] JIANG Y, WANG J, XUE X. Query-adaptive image search with hash codes[J]. IEEE Transactions on Multimedia, 2013, 15(2): 442-453.
- [18] SHUM H Y. QsRank: query-sensitive hash code ranking for efficient ϵ -neighbor search[C]//IEEE Conference on Pattern Recognition, June 16-21, 2012, Providence, RI, USA. Piscataway: IEEE Press, 2012.
- [19] JI T, LIU X, DENG C, et al. Query-adaptive hash code ranking for fast nearest neighbor search[C]//ACM International Conference on Multimedia, Dec 3-7, 2014, Orlando, FL, USA. New York: ACM Press, 2014.
- [20] LI X, LIN G, SHEN C, et al. Learning hash functions using column generation[J]. Computer Science, 2013: 142-150.
- [21] NOROUZI M, FLEET D J, SALAKHUTDINOV R. Hamming distance metric learning[C]//International Conference on Neural Information Processing System, Dec 3-6, 2012, Lake Tahoe, Nevada. New York: ACM Press, 2012: 1061-1069.
- [22] WANG J, LIU W, SUN A X, et al. Learning hash codes with

listwise supervision[C]//The 2013 IEEE International Conference on Computer Vision, Dec 1-8, 2013, Sydney, Australia. Piscataway: IEEE Press, 2013: 3032-3039.

- [23] WANG Q, ZHANG Z, SI L. Ranking preserving hashing for fast similarity search[C]//IJCAI, July 25-31, 2015, Buenos Aires, Argentina. [S.l.: s.n], 2015.
- [24] YAO T, LONG F, MEI T, et al. Deep semantic preserving and ranking-based hashing for image retrieval[C]//IJCAI, July 9-15, 2016, New York, USA. [S.l.: s.n], 2016.
- [25] LIU L, SHAO L, SHEN F, et al. Discretely coding semantic rank orders for supervised image hashing[C]//2017 IEEE Conference on Computer Vision and Pattern Recognition, July 21-26, 2017, Honolulu, HI, USA. Piscataway: IEEE Press, 2017.

[作者简介]



杨安邦 (1992-), 男, 宁波大学信息科学与工程学院硕士生, 主要研究方向为大数据、数据挖掘。



钱江波 (1974-), 男, 博士, 宁波大学信息科学与工程学院教授, 主要研究方向为数据处理与挖掘、多维索引与查询优化、机器学习。



董一鸿 (1969-), 男, 博士, 宁波大学信息科学与工程学院教授, 主要研究方向为大数据、数据挖掘和人工智能。



陈华辉 (1964-), 男, 博士, 宁波大学信息科学与工程学院教授, 主要研究方向为数据处理与挖掘、云计算。