文章编号: 1001-0920(2019)12-2567-09 DOI: 10.13195/j.kzyjc.2018.0329

基于学习型哈希的在线近邻查找算法

钱江波, 胡 伟†, 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: 基于哈希的近邻查找技术在图像检索、文本匹配、数据挖掘等信息检索领域均有广泛应用. 该技术将原始数据通过哈希函数压缩成低维的二进制编码,然后在海明距离下排序检索,具有快速高效且维度不敏感的优势. 但是,目前学术界针对流数据的实时在线哈希学习方法的研究很少,而且基本没有讨论哈希函数的更新频率和稳定性问题. 针对这一问题,通过增加置信区间来减少更换哈希函数的频率,并构造在线学习的目标函数,使得算法尽可能保持稳定,且快速收敛. 为了验证所提出算法的效率和有效性,在公开数据集上与同类的OSH、OKH在线哈希算法进行比较,比较结果表明,所提出的算法在平均准确率和训练时间上有一定优势.

关键词: 高维数据；数据流；信息检索；近邻查找；在线哈希学习；监督学习

中图分类号: TP18 **文献标志码:** A

Online learning to Hash for nearest neighbor search

QIAN Jiang-bo, HU Wei†, CHEN Hua-hui, DONG Yi-hong

(Faculty Electrical Engineering and Computer Science，Ningbo University，Ningbo 315211，China)

Abstract: Hash-based methods for nearest neighbor search has been widely used in information retrieval area, such as image retrieval, data mining and text match. The methods compress original data into low-dimensional binary codes by Hash functions, and then sort and search under Hamming distance. Therefore, the methods have the advantages of efficiency and dimension insensitivity for searching large-scale data. Currently, there is little literature that discuss on learning to online Hash-based methods for real-time dynamic streaming data. Furthermore, those methods do not discuss the update frequency of Hash functions and the stability. In order to solve the problem and improve the efficiency of learning to online Hash, the confidence interval is first designed to reduce the frequency of changing the Hash table, and the objective function is proposed to keep the Hash models as stable and convergent as possible. Compared with some related online Hash algorithms on several public large-scale datasets, the proposed method is competitive under the average accuracy and training time.

Keywords: high-dimensional data；stream data；information retrieval；nearest neighbor search；online learning to Hash；supervised learning

0 引 言

近邻查找(Nearest neighbor search)是信息检索领域的一个重要研究方向,在图像检索、数据挖掘等领域均有广泛应用. 随着信息技术的快速发展,数据呈现海量高维多样化特征,使得近邻查找越来越难以处理. 基于哈希的近邻查找方法的主要思想是将原始数据经过保距哈希函数映射到海明空间,利用计算机快速异或运算能力,以海明距离来取代计算原始数据的距离. 这里海明距离是两个字符串对应位置的不同字符的个数. 该方法具有空间占用小、计算速度快等优点,已经成为近邻查找领域的重要研究方向.

映射方法可以分为数据独立和数据依赖两类. 数据独立的方法一般基于随机空间划分的方法产生哈希函数,这类方法通常需要随着数据增长而增加编码长度,从而提高数据的搜索精度,这对于海量、高维、异构等数据,在时间效率和空间效率上会产生较大困难和挑战. 典型的方法有局部敏感哈希LSH[1]及其变种[2-4]. 数据依赖哈希方法主要通过判别数据结构及分布信息来自动学习哈希函数,并且根据数据标签属性进一步细分为: 1) 监督学习,如基于全局的最小损失哈希(Minimal loss hashing, MLH)[2]、监督的核哈希方法(Supervised hashing with kernels, SH)[5]、

收稿日期: 2018-03-20；**修回日期:** 2018-09-08.

基金项目: 国家自然科学基金项目(61472194, 61572266)；浙江省自然科学基金项目(LZ20F020001, LY20F020009, LY16F020003)；宁波市领军和拔尖人才培养工程择优科研项目(NBLJ201801003).

责任编委: 孙秋野.

†通讯作者. E-mail: huweiweis@foxmail.com.

基于决策树监督哈希(Fast supervised hashing with decision trees)[6]等; 2)半监督学习,如半监督图像检索[7]、半监督分类[8]、半监督降维[9]、半监督聚类[10]、半监督回归[11]等; 3)无监督学习,如谱哈希(Spectral hashing)[12]、迭代量化哈希(ITQ)[13]、深度哈希(Deep hashing)[14]、语义哈希(Semantic hashing)[15]等. 上述数据依赖的方法不需要较长的哈希编码也能得到较好的查询精度.

在线近邻查找处理和应用由于需要考虑增量计算、时延、存储消耗和准确性等多项因素[16],目前成果比较少. 针对连续空间强化学习问题,文献[17]提出了一种基于局部加权学习的增量最近邻时域差分学习框架;针对聚类精度不高、处理离群点能力较差以及不能实时检测数据流变化的缺陷,文献[18]提出了基于密度与近邻查找处理的数据流在线算法;文献[19]通过密度划分索引的方法逐步构建多棵k-d树,采用多近邻节点在线搜索方法,加快了近邻节点搜索速度;另外,文献[20-22]分别讨论了移动环境下、路网环境下、在线社交网络环境下如何快速准确获取在线近邻的方法和处理框架. 但是,随着数据维度不断增长,存储空间代价随之成倍增长.

在线哈希技术研究如何更新哈希函数来适应增量数据,且不需要重新学习历史数据. 如何根据之前的学习信息针对到来的新样本更新哈希函数是一个急需解决的问题. 文献[23]最早提出了一种基于在线哈希学习原型,在每次当前迭代中都使用新的一对数据样本,并且根据海明距离设计一对样本相似损失函数,测量哈希编码相似性,进一步用预测损失函数评估当前哈希投影向量是否适应当前数据,并且期望模型在更新过程中尽可能保存上一轮投影向量的历史信息;为了使原有在线哈希算法在损失函数理论上更完善,文献[24]针对哈希函数损失阈值提出了改进的在线哈希弱监督在线哈希学习模型,不需要数据的标签信息,根据相似性和非相似性数据设计一种新的损失函数来衡量一对数据海明距离的差异度,并严谨地分析了在线哈希理论的上限损失. 其次,由于在算法更新中学习的哈希函数依赖新数据,根据在线哈希算法适应新数据的特性很容易陷入局部偏差,于是产生了多模型策略以减少这类偏差.

自适应性哈希(Adaptive hashing for fast similarity search)[25]利用数据样本相似度和海明距离关系,解决了在线模型怎样适应当前数据的问题. 首先,利用当前样本对的相似度和海明距离关系结合最小损失方差函数构造目标函数,使用梯度下降算法求解哈希投影向量;其次,进一步泛化目标函数,使得相似(不相似)样本数据对的海明距离最小化(最大化),减少更新机制引起更新冗余;最后,使用铰链损失函数筛选误差最大的哈希映射,重新进入迭代计算,直致迭代数达到设置值. 在线监督哈希(Online supervised hashing)[26]适应数据变化且数据集标签种类未知. 使用随机的方法生成码本,使得数据根据哈希函数生成的编码与对应码本中类别匹配误差最小. 为了保证上轮次信息,把之前的哈希函数线性组合叠加,但是码本构造直接决定了编码的效率. 因此在后续文献中,针对这一问题提出了改进的监督哈希(Online supervised hashing)[27],根据在线监督哈希应用ECOC码本,提高了空间效率,并且针对原始海明损失公式求解的复杂性提出了一种基于上边界的高效求解方法,提高了算法时间效率.

在线概要哈希(Online sketch hashing)[28]是一种基于概要哈希的在线哈希函数学习方式,把整个数据集分成均匀的数据块,主要处理流数据下的小块数据并且基于每个数据块学习哈希函数. 根据谱哈希算法求解最优哈希编码的方法,把在线哈希的块数据处理转化为矩阵特征值问题,降低了复杂度,针对流数据模式不可访问历史数据导致偏置值的求解不确定问题,使用零均值块(Zero-mean sketch)方法,在一定程度上提高了数据处理效率.

在线自组织映射哈希(Online self-organizing hashing)[29]使用SOM算法更新哈希函数,结合神经网络训练数据准确性和哈希方法的简洁优点,利用高斯函数特性将训练过程数据的相似性很好地保留下来. 但是,对于高维数据,随着维度提高及输入量增多,计算量和时间也随之成指数上升.

多模式匹配哈希(Online hashing with mutual information, MIHash)[30]为了得到高质量的哈希编码,采用基于量化信息编码理论,利用数据集样本间互信息计算信息熵,在优化互信息目标时,利用可微分直方图合并技术推导出基于梯度的优化规则,最后用差异化规则需要把导出直方图合并,并将其应用于学习目标函数.

以上这些方法在一定程度上解决了在线学习哈希技术的难题,但是在学习过程中还存在哈希函数更新频率较快和哈希模型稳定性较弱等问题. 原因在于: 1)设计的损失函数在整个数据集上把相似和不相似样本设置成统一阈值,如果设置过大,将导致整个模型在局部趋于停止更新,反之则数据对于相似过于敏感,造成模型更新频繁,增大了计算开销; 2)仅根据相邻两次投影向量差别尽可能小来更新哈希函数,无法保证模型的稳定性.

在实际应用中,研究者更关心哈希模型在何时能快速迭代出最优哈希函数,以及是否能够达到稳定收敛的状态,而且在更新哈希模型过程中也需要更新频率尽可能少. 基于以下两点改进,本文提出在线选择哈希学习方法(OSELEH): 1)提出新的在线学习阈值损失函数,根据数据集样本相似(不相似)统计量动态设置一个置信区间,如果在置信度区间内,则维持原有哈希函数不变,否则进入更新模型; 2)提出新的在线模型更新的目标函数,使得在线哈希模型在相同规模的数据量下,该方法训练尽快达到稳定状态.

1　在线哈希问题描述

首先定义本文出现的基本数学符号及其表示含义. 整个数据集$\boldsymbol{X} \in \boldsymbol{R}^{d\times c}$, d为每个样本维数,数据集有c个样本.

在近邻查找中,通过哈希函数$\{h : \boldsymbol{R}^d \to \boldsymbol{Z}^r\}$将每个数据点$\boldsymbol{x}$映射为一个$r$维度的二值向量. 对于一个查询数据点$\boldsymbol{x} \in \boldsymbol{R}^d$,通过哈希函数$h$映射后得到的二值向量和数据库中已经二值化后的数据向量,进行逐个海明距离比较后排序,挑选海明距离近的点集返回作为查询点$\boldsymbol{x}$的近邻查找结果. 本文为了便于计算数据相似度与海明距离之间的关系,不需要把数据映射为标准的海明空间,定义映射后空间为$\boldsymbol{Z}^r \in \{-1,1\}^r$即可.

对于给定原始数据$\boldsymbol{X} \in \boldsymbol{R}^{d\times c}$, c个样本数据,通过哈希函数族映射到r位二进制编码向量的特征空间,第$k(1 \leqslant k \leqslant r)$个哈希函数定义为

$$f_k(\boldsymbol{w},\boldsymbol{x}) = \mathrm{sgn}(\boldsymbol{w}_k^{\mathrm{T}}\boldsymbol{x} + b_k) = \begin{cases} 1, \ \boldsymbol{w}_k^{\mathrm{T}}\boldsymbol{x} + b_k \geqslant 0; \\ -1, \ \text{other}. \end{cases} \tag{1}$$

其中$\boldsymbol{w}_k^{\mathrm{T}} \in \boldsymbol{R}^{d\times r}$是需要学习的哈希向量. 为了达到在二进制串$\{-1,1\}$的位数各占50%哈希编码, b_k取值为$\{-\boldsymbol{w}_k^{\mathrm{T}}\boldsymbol{x}_i\}_{i=1}^c$的均值,其中$c$为输入样本数. 把$b_k$代入式(1)可得

$$f_k(\boldsymbol{w},\boldsymbol{x}) = \mathrm{sgn}\Big(\boldsymbol{w}_k^{\mathrm{T}}\Big(\boldsymbol{x} - \frac{1}{c}\sum_{i=1}^{c}\boldsymbol{x}_i\Big)\Big). \tag{2}$$

由式(2)可以看出,如果数据样本数目发生变化,则b_k的值随之改变,对于流数据,样本数据始终不断处于动态更新中,若要学习适应更多数据的哈希函数,根据式(2)则需要每次重新计算变化后数据的所有样本的均值,计算量较大. 因此采取预先处理样本均值策略,即先将数据集分成连续的数据块,每个数据块$D_g(g \in \{1,2,\cdots\})$表示第$g$个数据块,在$D_g$中每个数据点需要减去当前数据块均值. 这样预处理不仅有助于提高数据处理性能,而且还简化了哈希函数学习过程. 由此在每个数据块内哈希函数为

$$F(\boldsymbol{W},\boldsymbol{X}_i) = \mathrm{sgn}(\boldsymbol{W}^{\mathrm{T}}\tilde{X}_i). \tag{3}$$

其中: $F = [f_1(\boldsymbol{w},\boldsymbol{x}), f_2(\boldsymbol{w},\boldsymbol{x}), \cdots, f_r(\boldsymbol{w},\boldsymbol{x})]$是$r$位的哈希编码, $\boldsymbol{W} = [\boldsymbol{w}_1,\boldsymbol{w}_2,\cdots,\boldsymbol{w}_r] \in \boldsymbol{R}^{r\times d}$是哈希投影向量, $\tilde{\boldsymbol{X}}_i$是第i个数据块除去均值后的结果.

2　在线哈希更新方法

2.1　置信区间

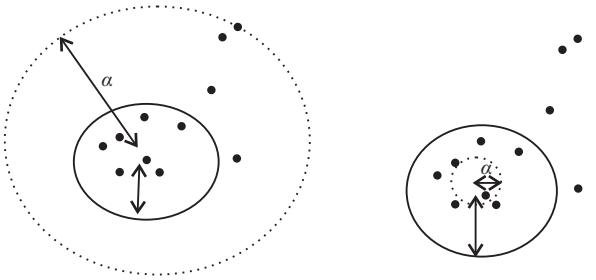
假设流数据序列成对,即将第t次两个新到来数据点记为$(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)$且相似度标记为$s_{ij}$,根据文献[25]可知,这对数据点的相似度标签定义为

$$s_{ij} = \begin{cases} 1, \ \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are similar}; \\ -1, \ \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are dissimilar}. \end{cases}$$

如果上轮次训练后得到的哈希投影向量为$\boldsymbol{W}^{(t-1)}$,则对于第t次到来的数据样本点对$(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)$,经映射后的哈希编码分别为$\boldsymbol{h}_i^t,\boldsymbol{h}_j^t$.

对于数据点$\boldsymbol{x}_i$的原始标记的标签信息,前一轮训练的哈希向量$\boldsymbol{W}^{(t-1)}$并不能总是准确地计算出对应的标签值. 当$\boldsymbol{W}^{(t-1)}$预测当前数据标签出现偏差,即当前数据映射后哈希编码不再具有保相似性时,需要更新哈希模型,选出更优的$\boldsymbol{W}^{(t)}$.

对于一对相似$s_{ij} = 1$数据点$(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)$,若直接根据经验给定数据样本相似阈值$\alpha$,代入模型计算误差损失,则可能会产生如图1所示情况,即α过大,导致损失值一直在较小的范围,假阳性提高;若α过小,则相似样本数据几乎为空集.



(a) α取值过大　　(b) α取值过小

图 1　相似数据阈值范围设置

在线哈希训练过程中,若相似性损失函数处理动态数据时使用确定的相似阈值α和不相似阈值β,则很可能由于数据更新使得固定阈值不适应哈希模型,导致频繁更新. 因此设计动态的α或β随数据实时调整取值范围,使得相似性损失函数置信值保持在适当的取值区间,不至于两级化. 在t次学习中,实时根据

数据分布和结构计算阈值,进一步由数据集的相似性(不相似性)分别调整损失函数阈值,对于将要到来的属于数据块D_g的数据$(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)$,若损失函数根据数据对样本的相似度标签$s_{ij}$不同,则在数据块中分别统计相似(不相似)数据样本的平均海明距离,划分不同的海明距离损失函数值域,因此设计相似性海明距离损失函数如下:

$$l(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t,s_{ij}^t)=\begin{cases}\max(0,d_h(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)-\alpha_g),\ s_{ij}^t=1;\\ \max(0,d_h(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)-\beta_g),\ s_{ij}^t=-1.\end{cases}\tag{4}$$

其中:$d_h(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)$是数据对$(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)$的海明距离;$\alpha_g$和$\beta_g$分别是在数据块$D_g$中相似和不相似数据的阈值统计量,根据上述原则需要使得在海明距离的阈值统计量上的相似数据小于等于不相似的数据,即$\alpha_g\leqslant\beta_g$;函数值$l(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t,s_{ij}^t)>0$表示$(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)$映射后失真误差较大,$l(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t,s_{ij}^t)=0$表示$(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)$经过哈希函数映射后相似性(不相似)得以保持.

在哈希模型中,训练过程采用动态的阈值,并且在一个阈值置信区间内调整取值范围,相比未设置置信区间的海明损失函数,新的损失函数对学习添加一个动态松弛,一定程度上减少了哈希函数更新次数. 因此加入了置信区间的哈希模型更能适应新数据且模型更新频率更合理.

2.2 更新策略

对于当前新输入的数据样本对,目标是使得更新后的哈希向量$\boldsymbol{W}^{(t)}$经过映射数据后得到的哈希编码对接近零损失. 为了使得更新哈希函数对新数据更适应,需要使得其哈希编码的损失值比$\boldsymbol{W}^{(t-1)}$映射后的损失值更小. 此外,在线哈希每次迭代时,数据样本对间海明距离损失可以用数值明确表示. 两步哈希(Two-step hash)[31]给出的哈希码内积与海明距离的关系如下:

$$\boldsymbol{h}_i\cdot\boldsymbol{h}_j=r-2d_h(\boldsymbol{x}_i,\boldsymbol{x}_j).\tag{5}$$

结合式(3)与数据对$(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)$的标签信息可重写式(5)并省去常系数,得到更新轮次数据哈希编码损失平方形式如下:

$$J(\boldsymbol{W}^{(t)},\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)=(F(\boldsymbol{W}^{(t)},\boldsymbol{x}_i)^{\mathrm{T}}F(W^{(t)},\boldsymbol{x}_j)-rs_{ij})^2,\tag{6}$$

其中$J(\boldsymbol{W}^{(t)},\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)$是数据样本在$t$轮次的$\boldsymbol{W}^{(t)}$映射后的海明距离损失值. 式(6)的值恒大于零,且值越小表明损失值越小、哈希编码越精确. 为了进一步使得在整个迭代计算过程中,随着样本数量增加,损失函数值逐渐稳定,需要使得t轮次的哈希函数$\boldsymbol{W}^{(t)}$满足$J(\boldsymbol{W}^{(t)},\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)\leqslant J(\boldsymbol{W}^{(t-1)},\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)$,为了达到这个目标定义,相邻两次更新哈希编码损失差值为

$$J(\boldsymbol{W}^{(t)},\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)-J(\boldsymbol{W}^{(t-1)},\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)\leqslant\eta.\tag{7}$$

其中:η是相邻两次模型更新差值上界,其值越小表示更新幅度越小,反之表示更新幅度越大.

构造样本数据的海明距离损失来调整规划更新哈希模型,使得更新的哈希向量$\boldsymbol{W}^{(t)}$更好地预测新样本. 若将新数据生成的哈希码代入式(4)后损失值在置信区间内,则维持上一轮次学习的哈希函数,否则需要调整哈希函数以适应新数据. 文献[28]中规定先前观察到的数据点在当前轮次中不可访问,利用投影向量$\boldsymbol{W}^{(t)}$,尽可能使得新学习的$\boldsymbol{W}^{(t)}$靠近上一轮投影向量$\boldsymbol{W}^{(t-1)}$,以保留$t-1$轮学习的信息. 但仅仅考虑相邻的投影矩阵关系会导致迭代过程很缓慢,还需要保证更新过程中不同时刻的投影向量的差异性,才能使t轮训练后的$\boldsymbol{W}^{(t)}$准确性提高、趋于稳定,即相邻的$\boldsymbol{W}^{(t)}$与$\boldsymbol{W}^{(t-1)}$差值的F-范数尽可能小,因此更新$\boldsymbol{W}^{(t-1)}$到$\boldsymbol{W}^{(t)}$时,更新哈希投影向量的目标不等式关系为

$$\|\boldsymbol{W}^{(t)}-\boldsymbol{W}^{(t-1)}\|_F^2<\cdots<\|\boldsymbol{W}^{(t)}-\boldsymbol{W}^{(t-n)}\|_F^2,\tag{8}$$

其中$n\in[1,2,\cdots,m]$. 式(8)表示对于流数据,不能只依据当前样本决定哈希函数,因为随着数据变化,准确率会产生偏差,为了约束这种不确定误差,保证哈希函数更新在相对可接受速率范围,避免噪声数据影响,并根据当前数据点调整哈希函数,还需要避免更新过于频繁,因此更新需要考虑前m个哈希函数,对当前数据$(\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)$哈希编码的结果,需要确定范围,保证当前哈希函数组合损失越来越小,能近似代表整个数据模型信息. 结合式(7)和(8),最终的目标函数表示为

$$\begin{aligned}&\boldsymbol{W}^{(t)}=\arg\min_w\sum_{n=1}^{m}(\|\boldsymbol{W}^{(t)}-\boldsymbol{W}^{(t-n)}\|_F^2);\\&\text{s.t}\ J(\boldsymbol{W}^{(t)},\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)-J(\boldsymbol{W}^{(t-1)},\boldsymbol{x}_i^t,\boldsymbol{x}_j^t)\leqslant\eta.\end{aligned}\tag{9}$$

目标函数(9)通过约束$\boldsymbol{W}^{(t)}$与m个连续投影向量$\boldsymbol{W}^{(t-n)}$的关系保留了历史数据信息,同时也保证了$\boldsymbol{W}^{(t)}$对t轮次数据对的海明距离的保相似性. 两个条件相互作用使得哈希函数随着时间稳定更新.

按照数据顺序训练多个哈希函数模型,其中用来训练当前模型的误差依赖于前一个模型的效果,整个过程是逐步优化的过程. 在训练样本的局部区域选

择哈希函数组合,即一个区域内保留m个哈希函数(分类器). m个投影向量选择是连续迭代生成的,为了防止m个数过大产生过拟合,需要在哈希函数个数达到规定阈值时,总是选取当前合理范围的哈希函数,即错误率低的哈希函数组合. 从上述目标函数可以看出,需要通过控制当前样本数据损失最小和保留局部历史信息两方面来优化$\boldsymbol{W}^{(t)}$,从而确保整体迭代计算的方向.

3　目标函数优化

当式(4)哈希编码损失$l(\boldsymbol{x}_i^t, \boldsymbol{x}_j^t, s_{ij}^t) \leqslant 0$时,说明上一轮训练学习的$\boldsymbol{W}^{(t-1)}$映射新数据后即是最优的哈希编码,能正确预测新数据对的相似性标签,因此不需要更新模型,否则需要更新哈希函数,按照式(9)重新计算哈希函数得到最优哈希编码,由拉格朗日多项式方程式可知,式(9)可重写为

$$\begin{aligned}&\Gamma(\boldsymbol{W}^{(t)}, \lambda, \eta) = \\&\sum_{n=1}^{m}(\|\boldsymbol{W}^{(t)} - \boldsymbol{W}^{(t-n)}\|_F^2)+ \\&\lambda(J(\boldsymbol{W}^{(t)}, \boldsymbol{x}_i^t, \boldsymbol{x}_j^t) - J(\boldsymbol{W}^{(t-1)}, \boldsymbol{x}_i^t, \boldsymbol{x}_j^t) - \eta),\end{aligned} \tag{10}$$

其中$\lambda \geqslant$是多项式乘子. 式(10)的目标是求解一个$\boldsymbol{W}^{(*)}$使得该向量在局部区域内有极小值,该方式是将目标函数转化为凸优化问题,即在局部区域$\boldsymbol{W}^{(t)}$到$\boldsymbol{W}^{(t-1)}$范围内存在局部极小值,$\boldsymbol{W}^{(*)}$即为更新后的投影向量. 因为式(10)是$d \times r$的高元方程,直接求解方程太过复杂. 机器学习中的随机梯度下降算法(SGD)已被广泛用于高维数据最优化问题,采用该算法求解函数的梯度,在梯度下降的方向上,找到其导数近似为零的$\boldsymbol{W}^{(*)}$,即为极小值点. 每次迭代时随机选取一对数据$l(\boldsymbol{x}_i^t, \boldsymbol{x}_j^t)$,从初始$\boldsymbol{W}^{(0)}$开始不断迭代产生新的投影向量,对于每一个投影向量都要沿着梯度方向计算更新,如此往复直至代价函数足够小为止,即在$\boldsymbol{W}$维度空间内,不断向函数减小的方向逼近,直至局部最低点.

首先通过式(10)对$\boldsymbol{W}^{(t)}$求偏导,在式(3)中使用双曲正切函数近似不可微分的sgn函数,即

$$\sigma(x) = \frac{\mathrm{e}^x - \mathrm{e}^{-x}}{\mathrm{e}^x + \mathrm{e}^{-x}}. \tag{11}$$

把式(6)引入(3)可得

$$\begin{aligned}&J(\boldsymbol{W}^{(t)}, \boldsymbol{x}_i^t, \boldsymbol{x}_j^t) = \\&[\sigma(\boldsymbol{W}^{(t)^{\mathrm{T}}}\boldsymbol{x}_i^t)^{\mathrm{T}} \cdot \sigma(\boldsymbol{W}^{(t)^{\mathrm{T}}}\boldsymbol{x}_j^t) - s_{ij}r]^2.\end{aligned} \tag{12}$$

式(12)是$J(\boldsymbol{W}^{(t)}, \boldsymbol{x}_i^t, \boldsymbol{x}_j^t)$对$\boldsymbol{W}^{(t)}$求偏导,根据复合函数链式求导法则,先用中间函数替换变量记为

$$\psi(\boldsymbol{W}^{(t)}) = \boldsymbol{W}^{(t)}\boldsymbol{x}_i,\ \varphi(\boldsymbol{W}^{(t)}) = \boldsymbol{W}^{(t)}\boldsymbol{x}_j. \tag{13}$$

将式(13)代入$J(\boldsymbol{W}^{(t)}, \boldsymbol{x}_i^t, \boldsymbol{x}_j^t)$化简,对$\boldsymbol{W}^{(t)}$求偏导后可得

$$\begin{aligned}&\frac{\partial J(\boldsymbol{W}^{(t)}, \boldsymbol{x}_i^t, \boldsymbol{x}_j^t)}{\partial \boldsymbol{W}^{(t)}} = \\&2[\sigma(\boldsymbol{W}^{(t)^{\mathrm{T}}}\boldsymbol{x}_i^t)^{\mathrm{T}} \cdot \sigma(\boldsymbol{W}^{(t)^{\mathrm{T}}}\boldsymbol{x}_j^t) - s_{ij}r]\times \\&\Big[\sigma(\psi(\boldsymbol{W}^{(t)}))^{\mathrm{T}} \cdot \frac{\partial \sigma(\varphi(\boldsymbol{W}^{(t)}))}{\partial \varphi} \cdot \frac{\partial \varphi(\boldsymbol{W}^{(t)})}{\partial \boldsymbol{W}^{(t)}} + \\&\sigma(\varphi(\boldsymbol{W}^{(t)})) \cdot \frac{\partial \sigma(\psi(\boldsymbol{W}^{(t)}))}{\partial \psi} \cdot \frac{\partial \psi(\boldsymbol{W}^{(t)})}{\partial \boldsymbol{W}^{(t)}}\Big].\end{aligned} \tag{14}$$

则式(10)对$\boldsymbol{W}^{(t)}$的偏导数为

$$\begin{aligned}&\frac{\partial \Gamma(\boldsymbol{W}^{(t)}, \lambda, \eta)}{\partial \boldsymbol{W}^{(t)}} = \\&2\Big(m\boldsymbol{W}^{(t)} - \sum_{i=1}^{m}\boldsymbol{W}^{(t-i)}\Big) + \frac{\partial J(\boldsymbol{W}^{(t)}, \boldsymbol{x}_i^t, \boldsymbol{x}_j^t)}{\partial \boldsymbol{W}^{(t)}}.\end{aligned} \tag{15}$$

采用梯度下降法,则$\boldsymbol{W}^{(t)}$的更新方式为

$$\boldsymbol{W}^{(t)} = \boldsymbol{W}^0 - \theta\frac{\partial \Gamma(\boldsymbol{W}^{(t)}, \lambda, \eta)}{\partial \boldsymbol{W}^{(t)}}. \tag{16}$$

其中: $\dfrac{\partial \Gamma(\boldsymbol{W}^{(t)}, \lambda, \eta)}{\partial \boldsymbol{W}^{(t)}}$为目标函数在$\boldsymbol{W}^{(t)}$处的梯度值,$\theta$为一个正的学习率参数.

关于在线哈希算法,随着轮次t的增加可以不断学习到新的哈希投影向量,当数据迭代增大时,整个算法系统已经学习到足够大的样本知识,需要计算在线哈希算法学习的上界问题. $\Gamma(\boldsymbol{W}^{(t)}, \lambda, \eta)$随着迭代进行,可能存在多个局部极小值点甚至最小值点. 在解得一个极小值哈希向量后,随着样本数量的增加,哈希模型在未迭代到新的更小极值点前不再频繁更新,逐渐达到一个稳定的状态.

在线哈希算法简单有效,运行过程只需要少量变量存储,记录从一次迭代到下一次迭代的哈希向量,输入样本的特征结构变化是比较敏感的,算法需要输入空间维数10倍的迭代次数才能达到稳定状态[25],当输入空间维数较高时收敛速度也会随之变得缓慢. 此外,输入向量特征值条件分布也会影响收敛速度,迭代过程的学习率也会对函数收敛速度产生影响. 当学习率θ初始值过大时,收敛速度快,在局部的极小值附近会有很大波动,甚至在迭代次数t时趋于无穷大;相反θ赋值过小,波动程度小,收敛速度慢. 因此学习率参数θ随着迭代次数而改变,根据文献[19]可得

$$\theta(t) = \frac{\theta_0}{(1 + (t/N))}. \tag{17}$$

其中: θ是初始阶段学习率,N是总迭代次数. 由式(17)可以看出,在t很小时,学习率近似等于$\theta(t) = \theta_0$,

这种可调的学习率在迭代过程中随机浮动,能使得迭代过程快速收敛.

以下是算法伪代码.

算法1 在线哈希学习算法.

输入: 流数据对$(\boldsymbol{x}_i^t, \boldsymbol{x}_j^t)$,数据集$X$,数据切分的块数batchCnt,拉格朗日乘子$\lambda$,学习率$\eta$,初始化哈希向量$\boldsymbol{W}^{(0)}$;

输出: t轮次学习后的哈希向量$\boldsymbol{W}^{(t)}$;

1) for $t = 1$ and g in batchCnt do

2) 统计第g个数据块数据中相似(不形似)阈值$\alpha_g(\beta_g)$;

3) for inc(t) and t_g in batchCnt / 2 do

4) 接受数据对$(\boldsymbol{x}_i^t, \boldsymbol{x}_j^t)$并计算数据的相似标签$s_{ij}$;

5) 根据式(3)计算数据对$(\boldsymbol{x}_i^t, \boldsymbol{x}_j^t)$的哈希编码$\boldsymbol{h}_i, \boldsymbol{h}_j$;

6) 根据式(4)计算一对数据的海明距离保相似损失值;

7) if $l(\boldsymbol{x}_i^t, \boldsymbol{x}_j^t, s_{ij}^t) \neq 0$ then

8) if $t \leqslant m$ then

9) 由式(15)计算梯度$\nabla_{\boldsymbol{w}}^{(t)} \Gamma(\boldsymbol{w}^{(t)}, \lambda, \eta)$;

10) 根据式(16)更新$\boldsymbol{W}^{(t)} \leftarrow \boldsymbol{W}^{(0)} - \theta \nabla_W \times \nabla_{\boldsymbol{w}}^{(t)} \Gamma(\boldsymbol{W}^{(t)}, \lambda, \eta)$;

11) else

12) 选择当前轮次前m向量作为候选集;

13) 计算m个哈希向量的范数$\|\boldsymbol{W}^{(t-n)}\|_F^2, 1 \leqslant n \leqslant t$并降序排序;

14) 通过式(16)更新哈希向量$\boldsymbol{W}^{(t)} \leftarrow \boldsymbol{W}^{(0)} - \theta \nabla_W \nabla_{\boldsymbol{w}}^{(t)} \Gamma(\boldsymbol{W}^{(t)}, \lambda, \eta)$;

15) else

16) $\boldsymbol{W}^{(t)} \leftarrow \boldsymbol{W}^{(t-1)}$;

17) end for

18) end for

19) return $\boldsymbol{W}^{(t)}$

4 实验结果与分析

为了评价本文提出的在线哈希学习方法的性能,分别在数据集GIST1M、CIFAR-10和MNIST上进行实验,这3个数据集近年来被广泛用于验证近邻查找方法的有效性.

分别采用相关的哈希方法Online kernel hashing(OKH)[24]、Online sketch hashing(OSH)[28]、Locality sensitive hashing(LSH)[1]与本文的方法进行对比,其中LSH通过随机投影向量和符号函数方法直接将原始数据映射到二进制编码,从而简化计算.

4.1 实验数据集和环境

MNIST数据集是来自250个不同人的手写数字图片,包含60 000个训练样本,10 000个测试样本,每张图片由28×28个像素组成,每个像素点用一个灰度值表示,一张图片表示成一个784维度的行向量,数据标签为整数0～9.

CIFAR-10由10个类别的32×32彩色图像组成,有50 000条训练数据,10 000条测试数据. 该数据集中的每个图像都被分配了一个类标签,标签的范围是1～10,提取3 072维GIST描述符[32]来表示每个图像.

GIST-1M包含从随机图像提取的一百万个960维GIST描述符. 对于这个数据集,随机选择1 000个数据点进行查询,并将剩下的数据用作数据库和训练集.

表1 数据集规模及划分

数据集	MNIST	CIFAR-10	GIST-1M
维度	784	3 072	960
数据大小	7×10^4	6×10^4	6×10^5
训练集	6×10^4	5×10^4	5×10^5
测试集	10^4	10^4	10^3

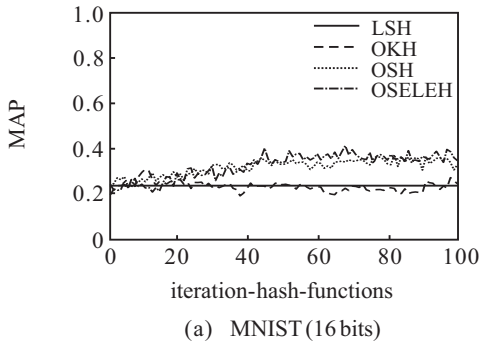
将上述每个数据样本集合分成独立的两部分,训练集、测试集都是从样本中随机抽取. 训练集用来控制模型参数,测试集合用来评估哈希模型的性能. 实验度量指标是平均准确率[33](MAP)和训练时间,其中平均准确率MAP是准确率-召回率曲线下的面积,能够评估一个哈希算法的总体性能. 本文提出的在线训练算法根据定义的数据对相似度确定标签,相当于间接的有监督在线哈希学习策略,在实验结果上根据原有标签判断本实验结果的准确性. 在GIST-1M数据集中没有确定的近邻标签,按照欧几里得距离将每块样本集合中距离最近的5 %作为近邻点[33],通过与OSH、OKH、LSH等方法在不同数据集上的平均准确率等对比来观测算法的性能,验证算法的正确性.

将所有实验结果独立运行多次以求达到数学期望. 实验的运行环境操作系统为Windows 8.1,处理器为Inter(R) Core(TM) i5-3 320 M CPU@2.60 GHz,内存为8 GB, Matlab 2013a.

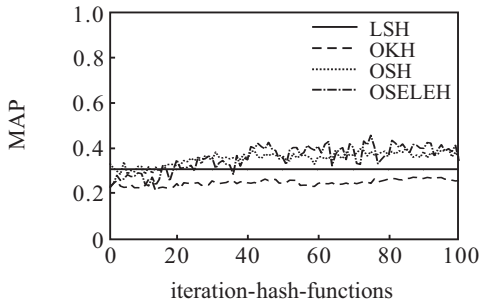
4.2 在线哈希方法实验对比

将MNIST、CIFAR-10、GIST-1M等数据分成若干块,模拟流数据形式. 选择公开源码的在线哈希方法(如OSH、OKH、LSH等)作为比较对象,并将LSH方法作为比较的基准线. 本文参数的设置基于文献[24-25],并结合本实验结果进行了调整. 在式(10)中,初始迭代的学习率是$0 < \lambda < 1$的一个随机数,η为

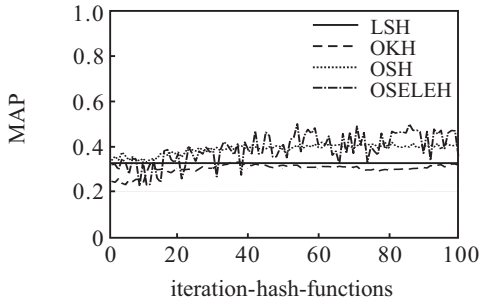
大于0的参数, $\boldsymbol{W}^{(0)}$ 是根据LSH随机生成的哈希向量, α_0 和 β_0 的初始都为当前哈希编码长度的一半. 图2～图7展示了不同的在线哈希算法在3个数据集上分别训练得到16位、32位、64位哈希编码后在对应测试数据集上的MAP值和平均MAP指标.



(a) MNIST (16 bits)



(b) MNIST (32 bits)



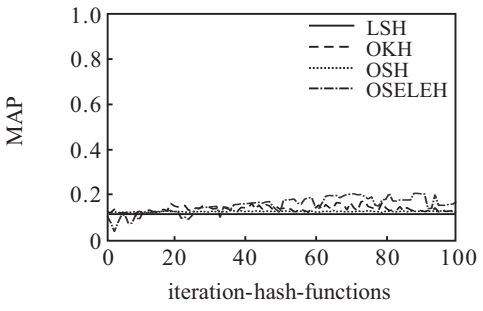
(c) MNIST (64 bits)

图 2　不同在线哈希算法在MNIST数据集及不同哈希编码位数上的MAP值对比

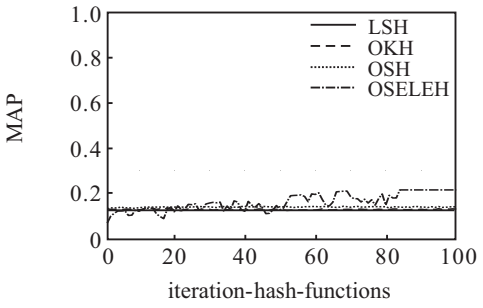
从图2可以看出, 在MNIST数据集上, 本文的OSELEH方法随着迭代的进行, 其MAP值呈现逐渐增长趋势. 由图5可见, 在16位、32位、64位哈希编码上, 本文方法比OSH方法的平均MAP值提高约0.03.

从图3可以看出, 在CIFAR-10数据集上, 本文的OSELEH方法随着迭代的进行, 准确率高于OSH和OKH, 且其MAP值整体呈现逐渐增长趋势. 由图6可见, 在16位哈希编码上, 本文方法比OKH方法的平均MAP值提高约0.15, 在32位哈希编码上比OSH方法的平均MAP值提高约0.16, 在64位哈希编码上比OSH方法的平均MAP提高约0.14.

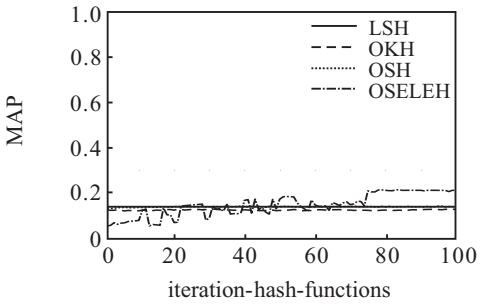
从图4可以看到, 在GIST-1M数据结果上, 本文方法在哈希编码位为32位时, 平均准确率要高于其



(a) CIFAR-10 (16 bits)



(b) CIFAR-10 (32 bits)



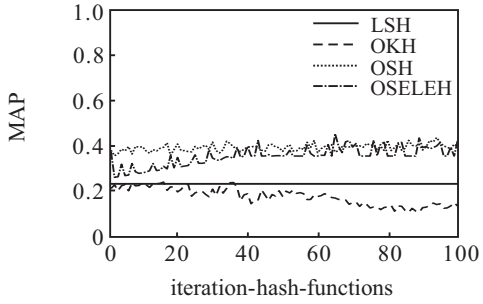
(c) CIFAR-10 (64 bits)

图 3　不同在线哈希算法在CIFAR-10数据集及不同哈希编码位数上的MAP值对比

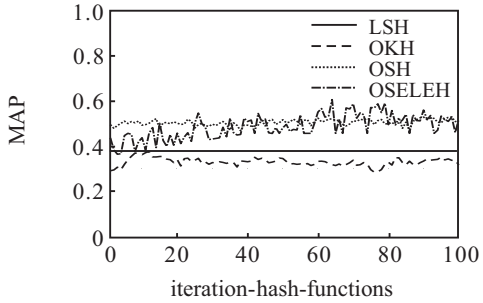
他方法. 由图7可知, 在32位哈希编码上, 本文方法比OSH方法的平均MAP值高0.02. 由于GIST-1M数据集没有确定的近邻标签, 方法OSELEH效果一般.

从图5～图7中还可以观察出哈希编码的长度与哈希算法性能关系, 每种方法在3种编码长度上的平均MAP表现在整体上会随着编码位数的提升有一定提高, 但提升并不显著, 因此简单提高哈希编码长度并不一定能提升算法本身性能. 由于本文的OSELEH方法在相似性损失函数上采用松弛间隔, 哈希模型更新频率相对另外两种方法较小, 这也造成每次更新准确率波动相对较大. 随着数据量的增多, 哈希函数学习到的准确性会逐渐增加, 不需要训练完整的数据集就能反应整个数据集的特征, 达到同类方法接近的平均准确率.

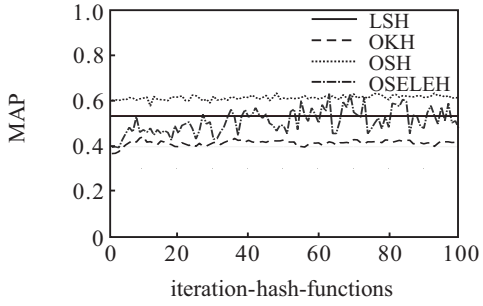
因为LSH是数据独立方法, 所以不需要训练, 表2比较了数据依赖方法OKH和OSK在CIFAR-10和GIST-1M数据集上的平均训练时间, 可见本文方法在一定程度上减少了训练时间.



(a) GIST-1M (16 bits)



(b) GIST-1M (32 bits)



(c) GIST-1M (64 bits)

图 4 不同在线哈希算法在GIST-1数据集及不同哈希编码位数上的MAP值对比

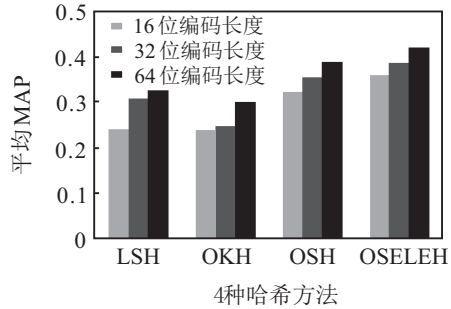


图 5 4种方法的不同长度哈希编码的平均MAP(MNIST)

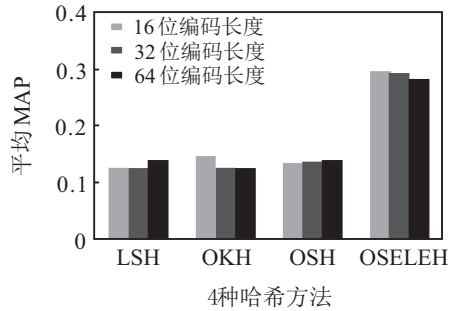


图 6 4种方法的不同长度哈希编码的平均MAP(CIFAR)

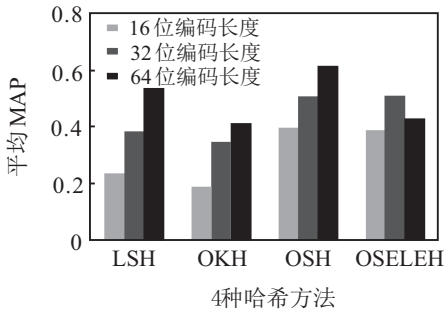


图 7 4种方法的不同长度哈希编码的平均MAP(GIST)

表2 训练时间对比 s

数据集	CIFAR-10	GIST-1M
OKH	31.96	0.069 252
OSK	8.23	0.036 331
OSELEH	8.10	0.027 239

5 结 论

本文提出了一种新的在线学习哈希算法,设计了分别根据样本相似(不相似性)的预测损失函数并且扩宽了损失函数的范围,进一步设置了相似(不相似)范围. 结合哈希模型需要保持历史信息与需要当前数据对损失最小的原则,提出新的目标函数,并对在线哈希算法的结果进行了分析,在理论上初步验证了可行性. 实验结果表明,本文方法在训练时间和MAP上具有一定优势.

参考文献(References)

[1] Datar M, Immorlica N, Indyk P, et al. Locality-sensitive hashing scheme based on p-stable distributions[C]. The 20th Symposium on Computational Geometry. New York: ACM, 2004: 253-262.

[2] Norouzi M, Fleet D J. Minimal loss hashing for compact binary codes[C]. Int Conf on Machine Learning. Washington: Omnipress, 2011: 353-360.

[3] Qian J, Zhu Q, Wang Y. Bloom filter based associative deletion[J]. IEEE Trans on Parallel & Distributed Systems, 2014, 25(8): 1986-1998.

[4] Qian J, Zhu Q, Chen H. Multi-granularity locality-sensitive bloom filter[J]. IEEE Trans on Computers, 2015, 64(12): 3500-3514.

[5] Liu W, Wang J, Ji R, et al. Supervised hashing with kernels[C]. Computer Vision and Pattern Recognition. Providence: IEEE, 2012: 2074-2081.

[6] Lin G, Shen C, Shi Q, et al. Fast supervised hashing with decision trees for high-dimensional data[C]. IEEE Conf on Computer Vision and Pattern Recognition. Columbus: IEEE, 2014: 1971-1978.

[7] Wu C, Zhu J, Cai D, et al. Semi-supervised nonlinear hashing using bootstrap sequential projection learning[J]. IEEE Trans on Knowledge & Data Engineering, 2013, 25(6): 1380-1393.

[8] Zhang T, Ando R K. Analysis of spectral kernel design based semi-supervised learning[C]. Proc of the 20th Annual Conf on Neural Information Processing Systems. Cambridge: MIT Press, 2006: 1601-1608.

[9] Brefeld U, Scheffer T, Wrobel S. Efficient co-regularised least squares regression[C]. Int Conf on Machine Learning. Pittsburgh: ACM, 2006: 137-144.

[10] Wagstaff K, Cardie C, Rogers S. Constrained k-means clustering with background knowledge[C]. The 18th Int Conf on Machine Learning. Williams College: Morgan Kaufmann Publishers, 2001: 577-584.

[11] Zhang D, Zhou Z H, Chen S. Semi-supervised dimensionality reduction[C]. SIAM Int Conf Data Mining. Minneapolis: SIAM, 2007: 629-634.

[12] Weiss Y, Torralba A, Fergus R. Spectral hashing[C]. Inte Conf on Neural Information Processing Systems. Vancouver: Curran Associates Inc, 2008: 1753-1760.

[13] Gong Y, Lazebnik S. Iterative quantization: A procrustean approach to learning binary codes[C]. IEEE Conf on Computer Vision and Pattern Recognition. Colorado Springs: IEEE, 2011: 817-824.

[14] Liong V E, Lu J, Wang G, et al. Deep hashing for compact binary codes learning[C]. Computer Vision and Pattern Recognition. Boston: IEEE, 2015: 2475-2483.

[15] Salakhutdinov R, Hinton G. Semantic hashing[J]. Int J of Approximate Reasoning, 2009, 50(7): 969-978.

[16] Xiaohui Jiang, Peng Hu, Yanchao Li, et al. A surveyof real-time approximate nearest neighbor query over streaming data for fog computing[J]. J of Parallel and Distributed Computing, 2018, 116(6): 50-62.

[17] 张春元, 朱清新, 钟声. 连续空间增量最近邻时域差分学习 [J]. 控制与决策, 2014, 29(12): 2121-2128.
(Zhang C Y, Zhu Q X, Zhong S. Temporal difference learning with incremental nearest neighbors in continuous spaces[J]. Control and Decision, 2014, 29(12): 2121-2128.)

[18] 张建朋, 陈福才, 李邵梅, 等. 基于密度与近邻传播的数据流聚类算法 [J]. 自动化学报, 2014, 40(2): 277-288.
(Zhang J P, Chen F C, Li S M, et al. Data stream clustering algorithm based on density and affinity propagation techniques[J]. Acta Automatica Sinica, 2014, 40(2): 277-288.)

[19] 温乃峰, 苏小红, 马培军, 等. 低空复杂环境下基于采样空间约减的无人机在线航迹规划算法 [J]. 自动化学报, 2014, 40(7): 1376-1390.
(Wen N F, Su X H, Ma P J, et al. Sampling space reduction-based UAV online path planning algorithm in complex low altitude environments[J]. Acta Automatica Sinica, 2014, 40(7): 1376-1390.)

[20] Jin Li, Xuguang Lan, Xiangwei Li, et al. Online variable coding length product quantization for fast nearest neighbor search in mobile retrieval[J]. IEEE Trans on Multimedia, 2017, 19(3): 559−570.

[21] Yinan Jing, Ling Hu, Wei-Shinn Ku, et al. Authentication of k nearest neighbor query on road networks[J]. IEEE Trans on Knowledge and Data Engineering, 2014, 26(6): 1494−1506.

[22] Xiwang Yang, Chao Liang, Miao Zhao, et al. Collaborative filtering-based recommendation of online social voting[J]. IEEE Trans on Computational Social Systems, 2017, 4(1): 1-13.

[23] Huang L K, Yang Q, Zheng W S. Online hashing[C]. Int Joint Conf on Artificial Intelligence. Beijing: AAAI Press, 2013: 1422-1428.

[24] Huang L K, Yang Q, Zheng W S. Online hashing[J]. IEEE Trans on Neural Networks & Learning Systems, 2018, 29(6): 2309-2322.

[25] Cakir F, Sclaroff S. Adaptive hashing for fast similarity search[C]. IEEE Int Conf on Computer Vision. Santiago: IEEE, 2015: 1044-1052.

[26] Cakir F, Bargal S A, Sclaroff S. Online supervised hashing for ever-growing datasets[EB/OL]. (2015-11-10)[2018-3-20]. https://arxiv.org/abs/1511.03257.

[27] Cakir F, Sclaroff S. Online supervised hashing[C]. Int Conf on Image Processing. Quebec City: IEEE, 2015: 2606-2610.

[28] Leng C, Wu J, Cheng J, et al. Online sketching hashing[C]. IEEE Conf on Computer Vision and Pattern Recognition. Boston: IEEE, 2015: 2503-2511.

[29] Chen J, Li Y, Lu H. Online self-organizing hashing[C]. Int Conf on Multimedia and Expo. Seattle: IEEE, 2016: 1-6.

[30] Çakir Fatih, Sclaroff S. Online supervised hashing[J]. Computer Vision & Image Understanding, 2017, 156(3): 162-173.

[31] Lin G, Shen C, Suter D, et al. A general two-step approach to learning-based hashing[C]. Int Conf on Computer Vision. Sydney: IEEE, 2014: 2552-2559.

[32] Zhang T, Ando R K. Analysis of spectral kernel design based semi-supervised learning[C]. Proc of the 20th Annual Conf on Neural Information Processing Systems. Vancouver: MIT Press. 2005: 1601-1608.

[33] Li K, Huang Z, Cheng Y C, et al. A maximal figure-of-merit learning approach to maximizing mean average precision with deep neural network based classifiers[C]. Int Conf on Acoustics, Speech and Signal Processing. Florence: IEEE, 2014: 4503-4507.

作者简介

钱江波 (1974−), 男, 教授, 博士生导师, 从事机器学习、模式识别与智能系统等研究, E-mail: qianjiangbo@nbu.edu.cn;

胡伟 (1990−), 男, 硕士生, 从事机器学习的研究, E-mail: huweiweis@foxmail.com;

陈华辉 (1964−), 男, 教授, 博士生导师, 从事数据库技术、流数据处理的研究, E-mail: chenhuahui@nbu.edu.cn;

董一鸿 (1969−), 男, 教授, 博士, 从事大数据处理、数据挖掘和人工智能等研究, E-mail: dongyihong@nbu.edu.cn.

(责任编辑: 闫　妍)