基于特征选择的局部敏感哈希位选择算法

周文柳，刘华文，李恩慧
浙江师范大学数学与计算机科学学院，浙江 金华 321001

摘要
作为主流的信息检索方法，局部敏感哈希往往需要生成较长的哈希码才能达到检索要求。然而，长哈希码需要消耗巨大的存储空间且携带大量的冗余哈希位。为了解决此问题，采用特征工程中10种简单高效的选择算法从长局部敏感哈希码中选择信息量丰富的哈希位，去除冗余、无效的哈希位。这10种选择算法使用不同的方式来刻画每一个哈希位的性能或两个哈希位之间的相关性，如方差、汉明距离等。通过去除长哈希码中性能较差或具有高相关性的哈希位进行哈希位的选择。将选择后的哈希码与原哈希码的性能进行比较。在4个常用数据集上的实验结果表明，去除冗余哈希位后的哈希码与原哈希码的性能几乎相同，且其哈希位的去除比率能达到30%~70%。

关键词
近似近邻搜索；哈希学习；哈希位选择；特征选择；降维

中图分类号：TP391 文献标识码：A doi: 10.11959/j.issn.2096-0271.2021061

Algorithm of locality sensitive hashing bit selection based on feature selection

ZHOU Wenhua, LIU Huawen, LI Enhui
College of Mathematics and Computer Science, Zhejiang Normal University, Jinhua 321001, China

Abstract
Locality sensitive hashing is one of the most popular information retrieval methods, which needs to generate long hashing bits to meet the retrieval requirement. However, a long hashing bits requires huge storage space, and contains plenty of redundant hashing bits. In order to solve this problem, ten simple and efficient selection algorithms in feature engineering were adopted to extract the hashing bits which carry the largest amount of information from the long hashing bits which were generated by locality sensitive hashing, and the redundant and useless hash bits were removed. Those ten algorithms tried to capture the performance of each hashing bit or the correlation among bits, such as variance and hamming distance. During selection process, the useless or high-correlated hashing bits were removed. Then the selected hashing bits were compared with the original long hashing bits. The experimental results on four common datasets show that the selected hashing bits works as well as the original hashing bits, and their reduction ratio can reach from 30% to 70%.

Key words
approximate nearest neighbor search, hashing learning, hashing bit selection, feature selection, dimensionality reduction

1 引言

随着互联网技术的高速发展，需要处理的数据的量爆炸式增长。在海量数据中检索出所需的数据变得越来越困难。最近邻搜索（nearest neighbor search，NNS）[1]在海量数据中寻找与查询数据最相似的近邻数据，在信息检索、数据挖掘、机器视觉等领域起到了至关重要的作用。若数据集中含有N个数据，则检索准确近邻数据的时间复杂度为$O(N)$。当数据库规模非常庞大时，计算成本迅速增加，因此通常使用近似最近邻（approximate nearest neighbor search，ANN）搜索作为替代方案来解决最近邻搜索问题[2]。因为在很多应用领域中，无须找到最近邻的数据，只要找到相似的数据即可。在过去的研究中，基于树结构（如KD tree[3]、K-means tree[4]）的算法在近邻问题上得到广泛应用。其主要思想是对数据空间进行划分，从而提高检索速度。但基于树结构的算法仅适用于低维数据，当遇到高维数据时，其性能快速下降。基于哈希的搜索算法在数据规模与数据维度很大时仍具有高效的检索性能，且其时间、空间复杂度较低，因此该算法成为主流的检索算法之一[5-6]。

在基于哈希的检索方法中，局部敏感哈希（locality-sensitive hashing，LSH）算法[6-8]是有代表性的算法之一。LSH会随机生成一组哈希函数，每一个哈希函数生成一个对应二值哈希位，将由多个哈希位组成的编码称为哈希码。LSH将原空间中的数据点映射成哈希码，使得相似度越高的数据具有相同哈希码的概率越高，而相似度越低的数据具有相同哈希码的概率越低。LSH的缺点是只有哈希码长度较长时，才能够达到理想的检索效果。但当哈希码的长度较长（如1 024位）时，计算的时间复杂度和数据所需的存储空间也随之增加。因此如何生成简短、性能优越的哈希码成为哈希学习中的主要问题[9]。

为了生成紧凑且信息量丰富的哈希码，近年来提出了各种类型的哈希算法，如无监督哈希学习[5]、有监督哈希学习[10-12]、半监督哈希学习[13]、深度哈希学习[14-15]等。上述哈希算法通过优化不同模型的目标函数来生成相应哈希码，如最小化排序损失、量化误差、重构误差等。但上述算法在处理不同的数据集和查询数据时，需要不断地调整模型结构和参数才能满足检索要求。

为了避免频繁地调整不同场景下的模型结构和参数，哈希位选择算法被提出[16-18]。该算法直接从现有的哈希位池中选取信息量最大的哈希码。在现有的研究工作中，很少有关于哈希位选择的研究。参考文献[17]将哈希位选择问题转化为图的二次规划问题，从而提取哈希码。然而，该图的二次规划为NP困难问题，只能得出其局部最优解；而且其时间复杂度较高，至少为$O(N^2)$，并不适用于处理大规模数据。

特征选择[19-20]也被称为特征子集选择，主要思想是从现有的M个特征中选取N个特征使得算法最优。特征选择能够有效减少数据的维度，降低存储成本，同时能够提高算法的效率。现有的特征选择算法主要分为3类：一是过滤法，根据特征的发散性或相关性对各个特征进行评分，通过设定阈值或排序方式选取特征；二是包裹法，每次选择若干特征并输入设定的目标函数，选出目标函数下的最优特征子集；三是嵌入法，使用与机器学习相关的算法对模型进行训练，得到各个特征的权值系数，根据系数从大到小选择特征。

本文的目的并不在于设计一个新的哈希算法，而是基于特征选择的思想，将每一个哈希位视为一个特征，从现有哈希算

法生成的哈希位池中高效地提取出信息量最大的哈希位。本文使用了10种简单且高效的基于特征选择的方法来进行哈希位选择。为了探索特征选择算法在哈希位选择上的作用，本文主要从以下两个角度进行探究：一是通过10种选择算法去除20%的冗余哈希位，观察精准率和召回率等性能指标的变化；二是在保持精准率和召回率等性能指标与原长度哈希位基本一致的前提下，探究每种选择算法能去除的最大冗余哈希位比率。

2 相关工作

2.1 局部敏感哈希

局部敏感哈希由于其原理简单、计算成本低而被广泛应用于各个领域，如大规模数据检索、异常检测、近邻问题[5,7,21]等。

局部敏感哈希将数据向量投影到随机超平面上，再进行二值化处理生成对应的二值码（哈希位），使数据在欧氏空间中的相似性在汉明空间中得以保存。设数据集 $\boldsymbol{X}=[x_1,x_2,\cdots,x_n]\in\mathbf{R}^{n\times d}$，$\boldsymbol{F}=\{f_1,f_2,\cdots,f_L\}\in\mathbf{R}^{d\times L}$为LSH中的函数族，$\boldsymbol{F}$中的每一项为随机生成。则数据的哈希位定义如下：

$$h(x)=\begin{cases}1, & x\cdot f\geqslant 0\\ 0, & \text{其他}\end{cases} \qquad (1)$$

数据点x与L个哈希函数f经过式（1）投影后生成长度为L的二值向量$\boldsymbol{H}(x)=\{h_1(x),h_2(x),\cdots,h_L(x)\}\in\{0,1\}^L$。整个数据集表示为二进制编码$\boldsymbol{B}$。

$$\boldsymbol{B}=\begin{pmatrix}\boldsymbol{H}(x_1)\\ \boldsymbol{H}(x_2)\\ \vdots\\ \boldsymbol{H}(x_n)\end{pmatrix}=\begin{pmatrix}h_1(x_1)\ h_2(x_1)\ \cdots\ h_L(x_1)\\ h_1(x_2)\ h_2(x_2)\cdots\ h_L(x_2)\\ \vdots\qquad\vdots\qquad\vdots\\ h_1(x_n)\ h_2(x_n)\ \cdots\ h_L(x_n)\end{pmatrix}=(\boldsymbol{h}_1,\boldsymbol{h}_2,\cdots,\boldsymbol{h}_L) \qquad (2)$$

其中，$\boldsymbol{h}_i\in\{0,1\}^{n\times 1}$表示编码$\boldsymbol{B}$的第$i$列，即整个数据集第$i$个哈希位组成的二值向量。

2.2 图模型哈希位选择

在现有的文献中，很少有关于哈希位选择的工作。仅有的基于图模型算法有参考文献[16-17]。在参考文献[17]中，图中节点权重表示每个哈希位保留原数据相似性的能力，边权重表示哈希位之间的独立性。一个好的哈希码能够保留数据在原空间中的相似性，且哈希码之间要互相独立，这使得哈希码包含的信息量最大。因此在进行哈希位选择时，应选取图中节点权重大且节点与节点之间的边权重也足够大的节点集合。此时，哈希位选择问题便转化为图的二次规划问题。然而该问题为NP困难问题。参考文献[17]采用模仿者动态理论求解，但是该解为局部最优解，而且需要调整节点权重与边权重之间的权值参数才能得到较优的哈希码。

在参考文献[18]中，使用马尔可夫过程求解上述图的二次规划问题。将节点权重（保留相似性的能力）转化为自我转移概率，将边权重（独立性）转化为节点之间的状态转移概率。通过马尔可夫过程，选取访问次数最多的节点来进行哈希位选择。然而使用马尔可夫过程求解的训练代价大、复杂度高。

3 哈希位选择算法

本节详细介绍10种哈希位选择算法，包括去除高相似性哈希位、低评分哈希位和随机选择3种类型。

3.1 去除高相似性哈希位

使用皮尔逊相关系数、余弦相似度、

Jaccard相似度等来描述哈希位之间的相似性程度。哈希位的相似性程度越高，其某种特定距离越小，如欧氏距离、汉明距离等。

设 $\boldsymbol{S}\in R^{L\times L}$ 表示 L 个哈希位之间的相似度矩阵，其中 $\boldsymbol{S}_{ij}=\mathrm{sim}(\boldsymbol{h}_i,\boldsymbol{h}_j)$，$\mathrm{sim}(\boldsymbol{h}_i,\boldsymbol{h}_j)$ 表示哈希位 $\boldsymbol{h}_i$ 与 $\boldsymbol{h}_j$ 之间的相似性大小。分别使用以下方式计算 $\mathrm{sim}(\boldsymbol{h}_i,\boldsymbol{h}_j)$。

（1）皮尔逊相关系数（高相关滤波）[22]。皮尔逊相关系数描述了两个向量之间变化趋势的相似性程度。

$$\mathrm{sim}(\boldsymbol{h}_i,\boldsymbol{h}_j)=\frac{\mathrm{cov}(\boldsymbol{h}_i,\boldsymbol{h}_j)}{\sqrt{D(\boldsymbol{h}_i)}\cdot\sqrt{D(\boldsymbol{h}_j)}} \tag{3}$$

其中，$\mathrm{cov}(\boldsymbol{h}_i,\boldsymbol{h}_j)$ 表示 $\boldsymbol{h}_i$ 与 $\boldsymbol{h}_j$ 之间的协方差，$D(\boldsymbol{h}_i)$ 表示 $\boldsymbol{h}_i$ 的标准差。

（2）余弦相似度[23]。特征之间的相似性用特征向量的夹角余弦来度量。

$$\mathrm{sim}(\boldsymbol{h}_i,\boldsymbol{h}_j)=\frac{\boldsymbol{h}_i\cdot\boldsymbol{h}_j}{\|\boldsymbol{h}_i\|\cdot\|\boldsymbol{h}_j\|} \tag{4}$$

（3）Jaccard相似度。Jaccard相似度通过两个向量集合的交集与并集之比来刻画向量之间的相似性。

$$\mathrm{sim}(\boldsymbol{h}_i,\boldsymbol{h}_j)=\frac{\boldsymbol{h}_i\cdot\boldsymbol{h}_j}{\|\boldsymbol{h}_i\|^2+\|\boldsymbol{h}_j\|^2+\boldsymbol{h}_i\cdot\boldsymbol{h}_j} \tag{5}$$

（4）基于欧氏距离的相似度。特征向量之间的欧氏距离是一种L_d范数，当d=2时，使用欧氏距离描述特征向量之间的相似性。

$$\mathrm{sim}(\boldsymbol{h}_i,\boldsymbol{h}_j)=\|\boldsymbol{h}_i-\boldsymbol{h}_j\|^2 \tag{6}$$

当d=1时，L_1表示曼哈顿距离。由于哈希码均为二值向量，哈希位之间的欧氏距离等于曼哈顿距离。

（5）基于汉明距离的相似度。汉明距离描述了两个集合之间的重合程度。重合程度越高，两个特征向量越相似。

$$\mathrm{sim}(\boldsymbol{h}_i,\boldsymbol{h}_j)=\sum_{k=1}^{n}(\boldsymbol{h}_{ik}\oplus\boldsymbol{h}_{ik}) \tag{7}$$

其中，$\oplus$ 表示异或运算，若 $\boldsymbol{h}_{ik}$ 与 $\boldsymbol{h}_{jk}$ 相同则结果为1，不同则为0。

（6）基于互信息的相似度[24]。互信息描述了两个变量之间包含的信息量大小。互信息越大，则两个向量之间包含的信息越大，两个向量越相似。

$$\mathrm{sim}(\boldsymbol{h}_i,\boldsymbol{h}_j)=\sum_{\boldsymbol{h}_i,\boldsymbol{h}_j}p(\boldsymbol{h}_i,\boldsymbol{h}_j)\log_2\frac{p(\boldsymbol{h}_i,\boldsymbol{h}_j)}{p(\boldsymbol{h}_i)p(\boldsymbol{h}_j)} \tag{8}$$

其中，$p(\boldsymbol{h}_i)$ 表示 $\boldsymbol{h}_i$ 的概率分布，$p(\boldsymbol{h}_i,\boldsymbol{h}_j)$ 表示 $\boldsymbol{h}_i$、$\boldsymbol{h}_j$ 的联合概率分布。

上述6种方式刻画了哈希位之间的相似性程度，通过去除高相似性哈希位选择出独立且信息量丰富的哈希位。具体算法RHSHB（remove high similarity hashing bit）如下。

算法1 RHSHB算法

输入：数据集$\boldsymbol{X}$，哈希码长度L，选择后的哈希码长度k。

输出：数据集哈希码 $\boldsymbol{B}'$。

① 使用式（1）得到数据集$\boldsymbol{X}$的哈希码$\boldsymbol{B}$。

② 分别使用式（3）~式（8）计算哈希位之间的相似度矩阵$\boldsymbol{S}$。

③ 将$\boldsymbol{S}$的上三角阵按从大到小排序，将前$L-k$个数值（具有高相似度）所在的列号作为需要去除的哈希位，记为集合$\boldsymbol{D}$。

④ 去除哈希码$\boldsymbol{B}$中集合$\boldsymbol{D}$记录的哈希位，得到去除冗余哈希位后的哈希码 $\boldsymbol{B}'$。

3.2 去除低评分哈希位

通过计算每个哈希位的方差、拉普拉斯分数、信息熵等属性来评定每个哈希位的好坏，每个哈希位给予相应的评分$\mathrm{score}(\boldsymbol{h}_i)$，去除其中评分低的哈希位。$\mathrm{score}(\boldsymbol{h}_i)$的计算方式如下。

（1）低方差滤波。数据取值变化小的哈希位所包含的信息量越少，该哈希位的方差

越低。将每个哈希位的方差作为评分。

$$\text{score}(\boldsymbol{h}_i)=\text{var}(\boldsymbol{h}_i) \tag{9}$$

其中，$\text{var}(\boldsymbol{h}_i)$表示$\boldsymbol{h}_i$的方差。

（2）拉普拉斯分数[25]。拉普拉斯分数描述了各个特征保留数据局部结构的能力。对于原始空间中的两个近邻点$\boldsymbol{X}_i$和$\boldsymbol{X}_j$，一个好的特征能够保持这种近邻关系，这在拉普拉斯分数上体现为数值变小。哈希位$\boldsymbol{h}_r$的拉普拉斯分数定义为：

$$L(\boldsymbol{h}_r)=\frac{\sum_{ij}(\boldsymbol{h}_{ri}-\boldsymbol{h}_{rj})^2\boldsymbol{S}_{ij}}{\text{var}(\boldsymbol{h}_r)} \tag{10}$$

其中，T_{ij}表示样本i与样本j之间的权重，$T_{ij}=\mathrm{e}^{-\frac{\|x_i-x_j\|^2}{t}},t>0$。

将每个哈希位视为一个特征，则哈希位$\boldsymbol{h}_r$的评分为：

$$\text{score}(\boldsymbol{h}_r)=1-L(\boldsymbol{h}_r) \tag{11}$$

（3）信息熵[26]。哈希位的信息熵值越大，该哈希位的不确定性程度越高，包含的信息量越大。使用信息熵作为哈希位的评分：

$$\text{score}(\boldsymbol{h}_i)=-\sum_{i=1}^{m}p(\boldsymbol{h}_i)\log_2(p(\boldsymbol{h}_i)) \tag{12}$$

其中，$p(\boldsymbol{h}_i)$表示$\boldsymbol{h}_i$取值的概率分布，m表示$\boldsymbol{h}_i$取值的个数。在哈希位中，m=2，即$\boldsymbol{h}_i$中元素的取值只能为0或1。

通过上述3种方式计算每个哈希位的评分，选择评分高的哈希位。具体算法SHHBS（select high hashing bit score）如下。

算法2 SHHBS算法

输入：数据集$\boldsymbol{X}$，哈希码长度L，选择后的哈希码长度k。

输出：数据集哈希码$\boldsymbol{B}'$。

① 使用式(1)得到数据集$\boldsymbol{X}$的哈希码$\boldsymbol{B}$。

② 分别使用式（9）~式（12）计算每个哈希位的分数，记为$\text{score}\in\boldsymbol{R}^L$。

③ 将score从大到小排序，将前k个数值所在的列号作为选取的哈希位，记为集合$\boldsymbol{D}$。

④ 提取哈希码$\boldsymbol{B}$中集合$\boldsymbol{D}$记录的哈希位，得到去除冗余哈希位后的哈希码$\boldsymbol{B}'$。

3.3 随机选择

随机选择是一种直接的选择方式，即不考虑哈希位的属性或哈希位之间的关系，从现有的哈希位集合中随机选取哈希位子集。随机哈希位选择的具体算法如下。

算法3 随机选择算法

输入：数据集$\boldsymbol{X}$，哈希码长度L，选择后的哈希码长度k。

输出：数据集哈希码$\boldsymbol{B}'$。

① 使用式（1）得到数据集$\boldsymbol{X}$的哈希码$\boldsymbol{B}$。

② 从1至L中随机均匀生成k个随机数，记为集合$\boldsymbol{D}$。

③ 提取哈希码$\boldsymbol{B}$中集合$\boldsymbol{D}$记录的哈希位，得到去除冗余哈希位后的哈希码$\boldsymbol{B}'$。

4 实验与分析

4.1 数据集与实验设置

本文使用两个有标签数据集和两个无标签数据集进行实验验证。其中有标签数据集分别为CIFAR-10[27]和MNIST[28]，将具有相同标签的数据作为真实近邻点；无标签数据集分别为LabelMe[29]和Corel[30]，将其欧氏空间下的近邻点作为真实近邻点。下面简要描述上述4个常用数据集。

MNIST：MNIST数据集为整数0~9的手写数字图片，包含70 000张28×28像素的灰度图片。

CIFAR-10：CIFAR-10包含60 000张32×32像素的彩色图片。所有图片被分为10个种类，每类图片中含有6 000张图片。

LabelMe：LabelMe数据集包含

22 000张彩色图片，图片均为生活中的场景与实体。

Corel：Corel数据集包含10 000张192×128像素的彩色图片。其中多为风景类图片，如日落、山脉等。

对于MNIST和CIFAR-10两个数据集，分别从每个类别中随机抽取1 000张图片作为查询集（共计10 000张图片），剩余的所有图片作为数据库。对于LabelMe和Corel数据集，分别从中随机抽取3 000张图片作为查询集，余下的所有图片作为数据库。MNIST数据集直接使用图片的像素值作为特征向量（786=28×28），其他3个数据集则提取每张图片512维的GIST特征作为特征向量。

4.2 评价指标

本文采用文献中广泛使用的精准度（precision）、召回率（recall）、平均精度均值（mean average precision，MAP）3个性能指标来衡量实验结果。将测试数据的真实近邻点集合定义为$\boldsymbol{R}$，假设测试数据返回的数据集合为$\boldsymbol{R}'$，则定义精准度和召回率分别为：

$$\text{precision}=\frac{\left|\boldsymbol{R}\cap\boldsymbol{R}'\right|}{\left|\boldsymbol{R}'\right|} \tag{13}$$

$$\text{recall}=\frac{\left|\boldsymbol{R}\cap\boldsymbol{R}'\right|}{\left|\boldsymbol{R}\right|} \tag{14}$$

为了描述哈希位选择前后性能的变化，取返回不同数据点个数下的平均精准度（mean precision，MP）和平均召回率（mean recall，MR）进行对比，定义MP与MR为：

$$\text{MP}=\frac{1}{|\boldsymbol{Q}|}\sum_{i=1}^{|\boldsymbol{Q}|}\text{precision}_i \tag{15}$$

$$\text{MR}=\frac{1}{|\boldsymbol{Q}|}\sum_{i=1}^{|\boldsymbol{Q}|}\text{recall}_i \tag{16}$$

其中，$\boldsymbol{Q}=\{10,50,100,200,400,600,800,1\,000\}$表示返回数据点的个数。

根据平均精准度可以得到广泛使用的MAP：

$$\text{MAP}=\frac{1}{|\boldsymbol{M}|}\sum_{i=1}^{|\boldsymbol{M}|}\text{MP}_i \tag{17}$$

其中，$\boldsymbol{M}$表示查询数据集。

4.3 实验结果

为了清晰地展示图片中的内容，将第2.2节中基于图模型的哈希位选择和本文使用的10种哈希位选择算法分别命名为：NDomSet（图模型）、HCF（高相关滤波）、Cosine（余弦相似度）、Hamming（汉明距离）、Euc（欧氏距离）、MI（互信息）、Jaccard（Jaccard相似度）、LCV（低方差滤波）、LS（拉普拉斯分数）、IE（信息熵）、Random（随机）。

在实验过程中，分别使用局部敏感哈希生成的128、256、512、1 024位哈希池进行哈希位选择。每个哈希码长度均约简（即去除冗余哈希位）20%，则约简后的哈希码长度为102、205、410、819位。

局部敏感哈希约简20%的哈希位后与原哈希码在MP和MR上的对比分别如**图1**、**图2**所示。在LabelMe和Corel数据集上，当原哈希码为128、256位时，约简后的哈希码与原码在平均精准度和平均召回率上的误差在1%~2%之间；当原哈希码为512、1 024位时，除了基于Cosine的选择算法，大部分选择算法误差在0~1%之间。这一现象表明，原哈希码越长，约简相同比例的哈希码对其性能影响越小。

表1给出了在有标签数据集CIFAR-10和MNIST上约简20%哈希位后，MAP的前后对比。在CIFAR-10数据集上，不同长度的哈希码约简后的MAP均与原码的MAP保持一致（误差小于2%）；在MNIST数据集上，当原哈希码为128位时，基于欧氏距

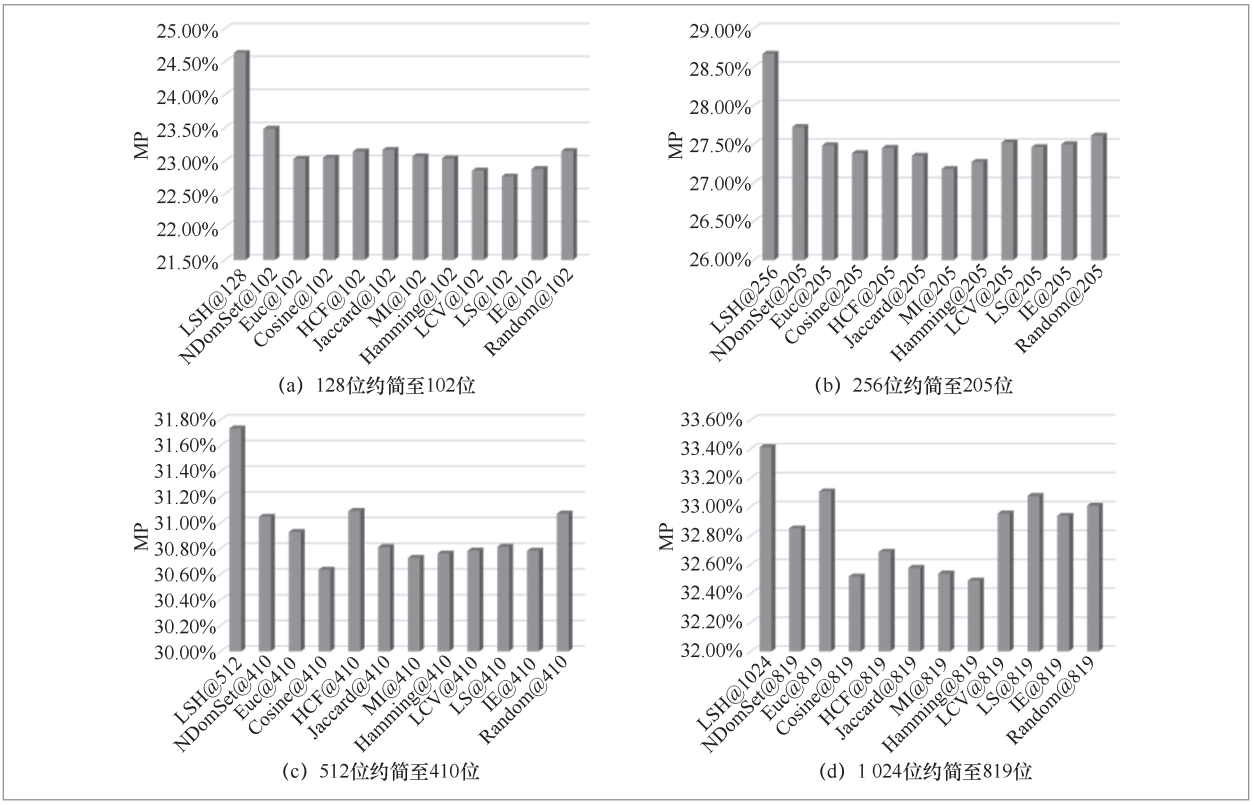

图 1 数据集 LabelMe 上不同编码长度下的 MP

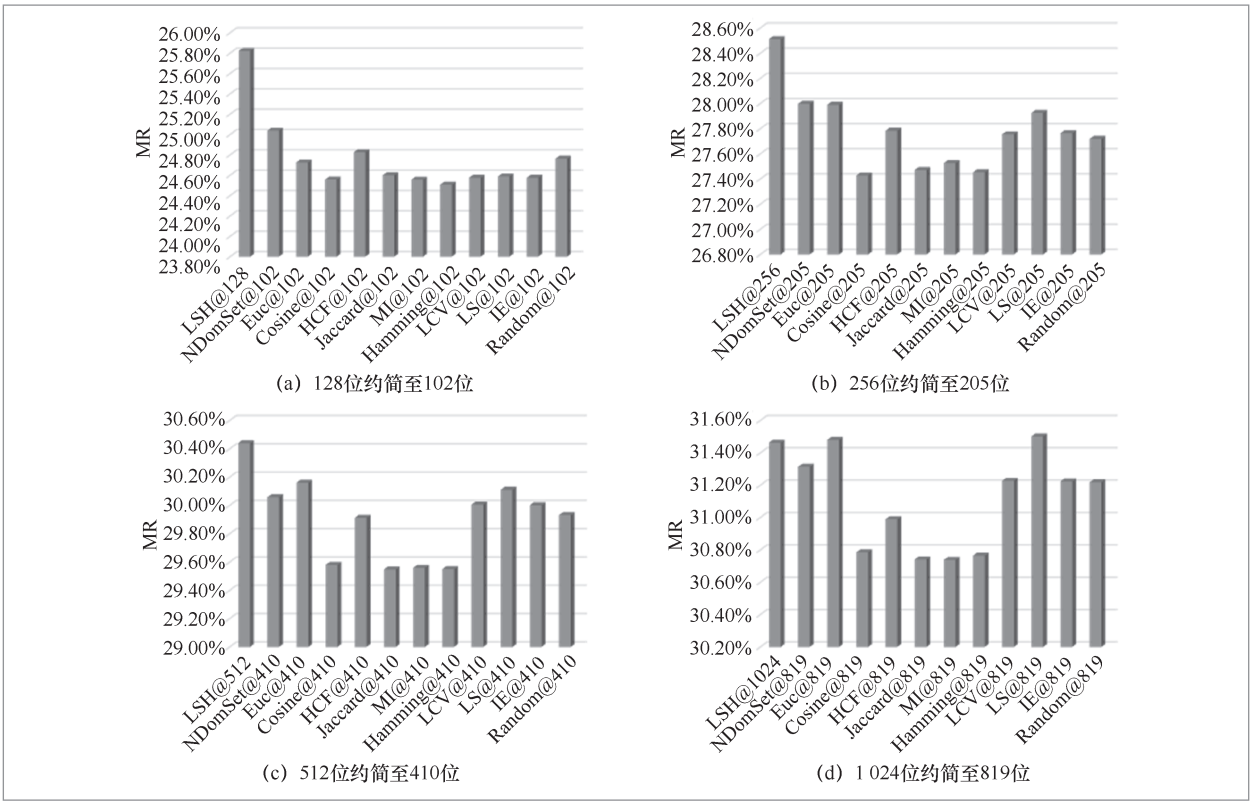

图 2 数据集 Corel 上不同编码长度下的 MR

表1 MAP在CIFAR-10和MNIST数据集上不同编码长度下的MAP

算法	CIFAR-10				MNIST			
	128/102	256/205	512/410	1 024/819	128/102	256/205	512/410	1 024/819
LSH	0.169 9	0.179 0	0.183 9	0.188 0	0.390 4	0.410 2	0.428 1	0.451 6
NDomSet	0.167 5	0.178 9	0.184 1	0.190 4	0.389 8	0.412 5	0.434 5	0.460 4
Euc	0.161 0	0.178 7	0.178 6	0.184 9	**0.363 8**	0.391 9	0.412 7	0.439 2
Cosine	0.168 1	0.178 6	0.183 6	0.187 8	0.380 0	0.406 4	0.429 8	0.453 7
HCF	0.166 3	0.178 5	0.182 4	0.187 8	0.378 2	0.402 7	0.423 9	0.448 6
Jaccard	0.168 2	0.178 3	0.183 1	0.187 8	0.377 9	0.404 5	0.428 3	0.453 9
MI	0.168 5	0.178 2	0.183 5	0.187 6	0.380 6	0.406 4	0.428 1	0.453 6
Hamming	0.168 4	0.178 1	0.183 3	0.187 5	0.379 2	0.406 8	0.428 5	0.455 2
LCV	0.167 8	0.178 0	0.184 1	0.190 0	**0.365 0**	0.392 3	0.417 8	0.440 9
LS	0.159 9	0.177 8	0.178 0	0.184 5	**0.361 3**	**0.385 6**	**0.404 5**	0.432 8
IE	0.165 9	0.177 7	0.182 2	0.187 6	**0.365 1**	0.392 3	0.417 8	0.441 1
Random	0.166 3	0.177 6	0.182 5	0.187 4	**0.375 1**	0.403 6	0.424 0	0.450 5

离(Euc)、低方差滤波(LCV)、拉普拉斯分数(LS)、信息熵(IE)的哈希位选择算法与原哈希码的性能误差在2%~3%之间。其他长度的哈希码基本与原码保持一致(误差小于2%)。

在MP、MR和MAP均与原哈希码基本保持一致的前提下(误差小于2%),探究128、256、512、1 024位局部敏感哈希在11种哈希位选择算法下能约简的最大比率。从**图3**和**图4**中可以发现,随着原哈希码长度的增加,使用不同哈希位选择算法能约简的哈希位比率也在增加。该现象说明虽然随着哈希码长度的增加,原局部敏感哈希的检索性能有所提升,但其中冗余的哈希位也相应增多。

在MNIST数据集上,基于欧氏距离、低方差滤波、拉普拉斯分数、信息熵的哈希位选择算法能约简的哈希位比率较少。而其他哈希位选择算法均能约简20%以上。当原哈希码为1 024位时,使用基于图模型、余弦相似度、高相关滤波等选择算法的约简比率高达60%以上。在CIFAR-10数据集上,所有哈希位选择算法均能约简20%以上的哈希码,且哈希码长度较长(如512、1 024)时,约简比率为30%~70%。

表2给出了不同哈希码长度下,对于给定的查询数据,检索3 000个近邻数据所需时间。从**表2**可以看出,检索所需时间随着哈希码长度的增加而增加。例如,哈希码长度从256位增加至512位时,检索时间增加近一倍。结合**图3**与**图4**可以看出,本文使用的哈希位选择算法能够将原哈希码约简30%~70%,使用约简后的哈希码进行信息检索,不仅能够充分减少检索所需时间,还可以降低数据(图片、文本等)转换后的哈希码所需存储空间。

表3给出了11种哈希位选择算法的时间复杂度和将512位哈希码约简20%后的MAP和实际运行时间。其中,n表示数据个数,d表示数据维度,k表示哈希码长度($k \ll n$)。从**表3**可以看出,虽然基于NDomSet的哈希位选择算法的MAP最高,但是其时间复杂度也最大。基于NDomSet的哈希位选择算法的MAP高于基于Cosine、HCF、Jaccard、Hamming、LCV、IE、Random的哈希位选择算法0~0.002,然而其运行时间为这几种算法的20~100倍(除了基于IE的哈希位选择算法)。因此,在处理小规模数据集和追求高精度的场景下可以使用基于NDomSet

的哈希位选择算法，但当处理大规模数据时，基于特征选择的哈希位选择算法更加高效，同时不会严重损失哈希码的精度。

5 结束语

本文首次将特征工程中的10种降维算法应用于哈希位选择中。在保证约简后的哈希码与原码性能基本一致的前提下，尽可能约简较多的哈希码，使得约简后的哈希码更加紧凑、高效，包含的冗余信息更少。约简后的哈希码不仅提高了检索效率，且减少了基于哈希码表示的数据集所需的存储空间。

参考文献:

[1] KALANTIDIS Y, AVRITHIS Y. Locally optimized product quantization for approximate nearest neighbor search[C]// Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE Press, 2014: 2329-2336.

[2] ANDONI A, INDYK P. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions[C]// Proceedings of the 2006 47th Annual IEEE Symposium on Foundations of Computer Science. Piscataway: IEEE Press, 2006: 459-468.

[3] BENTLEY J L. Multidimensional binary search trees used for associative searching[J]. Communications of the ACM, 1975, 18(9): 509-517.

[4] SILPA-ANAN C, HARTLEY R. Optimised KD-trees for fast image descriptor matching[C]//Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE Press, 2008: 1-8.

[5] WEISS Y, ANTONIO T, ROB F. Spectral hashing[C]//Advances in Neural Information

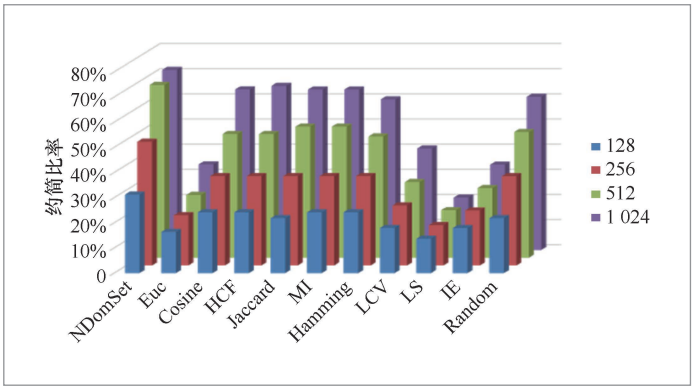

图 3 数据集 MNIST 上 11 种算法的约简比率对比

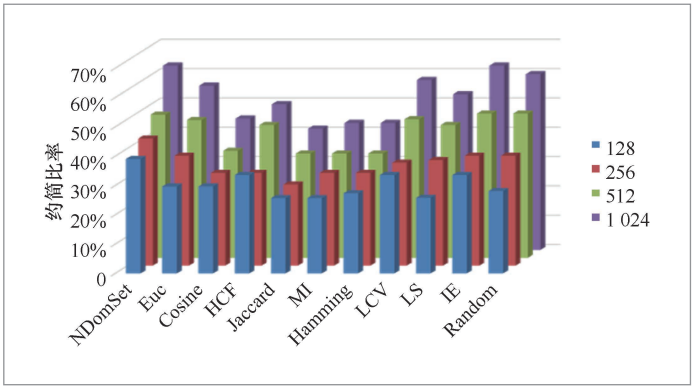

图 4 数据集 CIFAR-10 上 11 种算法约简的比率对比

表 2 不同哈希码长度下检索 3 000 个近邻数据所需时间

哈希码长度/bit	256	307	358	409	512
运行时间/s	7.74	9.22	10.42	11.92	14.47

表 3 数据集 CIFAR-10 上 11 种算法的时间复杂度、MAP 与运行时间

选择算法	时间复杂度	MAP	运行时间/s
NDomSet	$O(n^2d)$	0.184 1	10.680 9
Euc	$O(k^2)$	0.178 6	0.102 9
Cosine	$O(k^2)$	0.183 6	0.195 5
HCF	$O(k^2)$	0.182 4	0.143 9
Jaccard	$O(k^2)$	0.183 1	0.581 3
MI	$O(nk^2)$	0.183 5	10.143 9
Hamming	$O(k^2)$	0.183 3	0.370 8
LCV	$O(nk)$	0.184 1	0.121 2
LS	$O(n^2)$	0.178 0	0.403 4
IE	$O(k)$	0.182 2	0.005 7
Random	$O(1)$	0.182 5	0.000 7

Processing Systems 21. Cambridge: The MIT Press, 2008: 1753–1760

[6] GIONIS A, INDYK P, MOTWANI R. Similarity search in high dimensions via hashing[C]//Proceedings of the 25th International Conference on Very Large Data Bases. California: Morgan Kaufmann Publishers, 1999: 518–529.

[7] YAN C G, GONG B, WEI Y X, et al. Deep multi-view enhancement hashing for image retrieval[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021, 43(4):1445–1451.

[8] WANG J D, ZHANG T, SONG J K, et al. A survey on learning to hash[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2018, 40(4): 769–790.

[9] WANG Z, DUAN L Y, YUAN J S, et al. To project more or to quantize more: minimizing reconstruction bias for learning compact binary codes[C]// Proceedings of the 25th International Joint Conference on Artificial Intelligence. California: AAAI Press, 2016: 2181–2188.

[10] GONG Y C, 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 and Machine Intelligence, 2013, 35(12):2916–2929.

[11] NOROUZI M, FLEET D J. Minimal Loss Hashing for Compact Binary Codes[C]// Proceedings of the 28th International Conference on Machine Learning. Madison: Omnipress, 2011: 353–360.

[12] LIU W, WANG J, JI R R, et al. Supervised hashing with kernels[C]//Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE Press, 2012: 2074–2081.

[13] WANG J, KUMAR 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.

[14] ZHAO F, HUANG Y Z, WANG L, et al. Deep semantic ranking based hashing for multi-label image retrieval[C]// Proceedings of the 2015 IEEE Conference on Computer Vision and Patten Recognition. Piscataway: IEEE Press, 2015: 1556–1564.

[15] WANG J, LIU W, SUN A X, et al. Learning hash codes with listwise supervision[C]// Proceedings of the 2013 IEEE International Conference on Computer Vision. Piscataway: IEEE Press, 2013: 3032–3039.

[16] LIU X L, HE J F, CHANG S F. Hash bit selection for nearest neighbor search[J]. IEEE Transactions on Image Processing, 2017, 26(11): 5367–5380.

[17] LIU X L, HE J F, LANG B, et al. Hash bit selection: a unified solution for selection problems in hashing[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE Press, 2013: 1570–1577.

[18] ZHANG D C, LIU X L, LANG B. Hash bit selection using Markov process for approximate nearest neighbor search[C]// Proceedings of the International Conference on Advances in Mobile Computing. New York: ACM Press, 2013: 205–208.

[19] AL-TASHI Q, ABDULKADIR S J, RAIS H M, et al. Approaches to multi-objective feature selection: a systematic literature review[J]. IEEE Access, 2020, 8: 125076–125096.

[20] CAI J, LUO J W, WANG S L, et al. Feature selection in machine learning: a new perspective[J]. Neurocomputing, 2018, 300: 70–79.

[21] HEO J P, LEE Y, HE J F, et al. Spherical hashing[C]//Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE Press, 2012: 2957–2964.

[22] SENLIOL B, GULGEZEN G, YU L, et al. Fast correlation based filter (FCBF) with a different search strategy[C]//Proceedings of the 2008 23rd International Symposium on Computer and Information Sciences. Piscataway: IEEE Press, 2008: 1–4.

[23] NGUYEN H V, BAI L. Cosine similarity metric learning for face verification[C]// Proceedings of the Asian Conference on Computer Vision. Heidelberg: Springer,

2010: 709–720.
[24] GIERLICHS B, BATINA L, TUYLS P, et al. Mutual information analysis[C]// Proceedings of the International Workshop on Cryptographic Hardware and Embedded Systems. Heidelberg: Springer, 2011: 426–442.
[25] HE X F, CAI D, NIYOGI P. Laplacian score for feature selection[C]//Advances in Neural Information Processing System. [S.l.:s.n], 2005: 507–514.
[26] ZHENG K F, WANG X J. Feature selection method with joint maximal information entropy between features and class[J]. Patten Recognition, 2018, 77: 20–29.
[27] ABOUELNAGA Y, ALI O S, RADY H, et al. CIFAR-10: KNN-based ensemble of classifiers[C]//Proceedings of the 2016 International Conference on Computational Science and Computational Intelligence. Piscataway: IEEE Press, 2016: 1192–1195.
[28] COHEN G, AFSHAR S, TAPSON J, et al. EMNIST: extending MNIST to handwritten letters[C]//Proceedings of the 2017 International Joint Conference on Neural Networks. Piscataway: IEEE Press, 2017: 2921–2926.
[29] RUSSELL B C, TORRALBA A, MURPHY K P, et al. LabelMe: a database and web-based tool for image annotation[J]. International Journal of Computer Vision, 2008, 77: 157–173.
[30] TANG J Y, LEWIS P H. A study of quality issues for image-annotation with the Corel dataset[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2007, 17(3): 384–389.

作者简介

周文桦（1997– ），男，浙江师范大学数学与计算机科学学院硕士生，主要研究方向为信息检索、哈希学习。

刘华文（1977– ），男，博士，浙江师范大学数学与计算机科学学院教授，主要研究方向为数据挖掘。

李恩慧（1996– ），女，浙江师范大学数学与计算机科学学院硕士生，主要研究方向为聚类、异常点分析。

收稿日期： 2020-12-31
通信作者： 刘华文，hwliu@zjnu.edu.cn
基金项目： 国家自然科学基金资助项目（No.61976195）
Foundation Item: The National Natural Science Foundation of China (No.61976195)