詹增荣, 程丹. 基于核化局部敏感哈希的快速文档检索方法[J]. 湖南科技大学学报(自然科学版),2019,34(3):75-83. doi:10.13582/j.cnki.1672-9102.2019.03.011

Zhan Z R, Cheng D. Kernelized Local-Sensitive Hashing for Fast Document Retrieval [J]. Journal of Hunan University of Science and Technology(Natural Science Edition), 2019,34(3):75-83. doi:10.13582/j.cnki.1672-9102.2019.03.011

基于核化局部敏感哈希的快速文档检索方法

詹增荣[1*],程丹[2]

(1.广州番禺职业技术学院 信息工程学院,广东 广州 510480;2.广州体育职业技术学院 体育产业系,广东 广州 510650)

摘　要:在大规模文档中进行快速的相似搜索对绝大多数信息检索应用程序是至关重要的.基于局部敏感哈希的检索方法将高维的空间数据映射到低维的二进制海明空间,从而实现了快速搜索.本文给出了一个基于核化局部敏感哈希的快速文档检索方法,可以在保证时间效率下允许算法使用不同的相似函数进行快速检索.实验结果表明该方法在大规模文档集合检索中具有较好的效率和准确率.

关键词:局部敏感哈希;相似搜索;文档检索;核函数

中图分类号:TP391　　**文献标志码:**A　　**文章编号:**1672-9102(2019)03-0075-09

Kernelized Local-Sensitive Hashing for Fast Document Retrieval

Zhan Zengrong[1], Cheng Dan[2]

(1. School of Information Engineering, Guangzhou Panyu Polytechnic, Guangzhou 511480, China;
2. Department of Sport Industry, Guangzhou Polytechnic of Sport Vocational and Technical College, Guangzhou 510650, China)

Abstract: Fast similarity search for large databases is critical to most Information Retrieval (IR) applications. Local sensitive hash function was proved to be efficient, as it embed high-dimensional features into a low-dimensional Hamming space where items was fast searched. A kernelized local sensitive hash method for document retrieval was presented. The method made use of arbitrary kernel functions to perform fast search for a wide class of useful similarity function without losing time efficiency. The experiment result shows that this method has higher efficiency and better accuracy for large scale document retrieval.

Keywords: local-sensitive hashing; similarity search; document retrieval; kernel function

随着互联网高速的发展,用户在享受着互联网带来的便利同时也产生了极其大量的数据.IDC 最近的一项研究表明近两年平均产生了近 3 ZB 的数据,按照这个增长速度到 2020 年平均每个人将拥有 5 TB 的数据[1].因此,在这样大数据环境下,用户时常需要对自己的电脑数据或网络数据进行搜索,为此高效、准确、易于扩展的信息检索技术是当前非常热点的研究内容.

为了能够实现快速地对大量的文档进行索引和搜索,许多不同的搜索技术和框架被陆续提出来,如文献[2-5]中提出的相似搜索技术是当前研究的重点.相似搜索技术的问题可以描述为给定一个数据样本如一篇文档,如何从大量的文档集中找出和该文档最相似的文档列表.它对很多信息检索(Information

收稿日期:2015-05-26　　**修改日期:**2018-11-14

基金项目:广州市教育科学规划(201811675);广东省科技发展专项资金项目(706049150203);2018 年度广东省普通高校特色创新类项目(2018GWTSCX057);广州市市属高校科研项目(1201620532)

*通信作者,E-mail: zhanzr@gzpyp.edu.cn

Retrieval, IR)[6]应用起着重要的作用, 如重复性检测[7]、剽窃分析[8]、协同过滤[9]、缓存[10]以及多媒体内容检索[11]等.在众多新的相似性搜索技术当中,局部敏感哈希函数(Locality Sensitive Hashing, LSH)[12-13]非常广泛有效被应用到高维数据的检索中,如基于欧式距离的 LSH 就是一个成功的例子,文献[14]通过调用很多哈希表来保证查询的准确性,而文献[15-16]则针对此类 LSH 提出了减少内存消耗的改进方法.文献[17]提出了一种基于同心超球分割的非对称 LSH 方案,用于高维近似最大内积搜索.类似的,很多其他文章如文献[18-21]等则在如何构造低维二进制嵌入方面进行了研究,其实验应用包括了不同类别的搜索任务如重复检测、样本目标识别、姿势估计以及特征匹配等.

现有的很多方法存在着搜索效率低下和准确率不高等问题.而基于 LSH 的搜索方法,由于效率高已经成功被应用到很多相似性搜索任务中.本文给出一种基于核化 LSH 文档检索技术来解决在大规模文档中进行快速检索的问题.这种方法借助中心极限原理[22]证明了可以使用任意的核函数来对哈希后的文档进行快速相似搜索.此外,该方法最终给出的结论表明了其可以非常容易地被应用到不同的相似搜索应用中,而且具有高效率和高准确率等特点.

1 相关工作

在信息检索领域,任何一个文档都被表示成一个向量,如基于语义搜索的代表 pSearch[23]利用向量空间模型(Vector Space Model,VSM)[24]和潜在语义索引技术(Latent Semantic Indexing, LSI)[25]去生成一个语义空间,并且应用在分布式环境中,可以对大规模数据进行快速检索.此外,文档对应的向量维度一般都非常大,多达几千上万维[6].但是往往对文档的检索不需要返回一个准确的结果,因此,为了保证在一定时间内返回结果,快速相似性搜索在高维数据应用更为适合.

快速相似性搜索对很多应用程序起着关键的作用,因此是当前研究的热点话题.在低维数据搜索中,基于树状的算法可以准确地返回需要的结果.如 KD 树[26]利用空间的分割技术提供了一个高效的方法来对低维的特征空间向量进行搜索,但是在实际高维数据应用中,KD 树效率完全不能满足应用,甚至不如传统的线性扫描方法.类似的方法还有 R 树[27],同样也面临着对高维数据搜索效率不高等限制.

在大量高维数据空间中,基于哈希函数的方法可以极大地提高搜索的速度,从而被广泛应用[28].这些基于哈希的方法可以被认为是尽可能保证数据相似结构的基础上,将高维数据的特征向量嵌入到低维的海明空间中,然后在海明空间中去快速计算相互间的距离或相似性.与标准的降维方法如潜在语义索引[29-30]和局部保留索引[31-32]技术不同的是,哈希技术将特征向量映射为二进制代码,从而可以通过海明距离来非常快速地进行索引.如文献[20]中将文档转成二进制,可以使用类似 LSI 技术对文档降维后,对低维向量进行二进制化.另外,最近还提出了一种改进方法叫 Co-Hashing (LCH)[34].

在众多哈希算法中,最广泛使用的且能保持相似信息的算法还是局部敏感哈希算法(LSH).它利用一个随机的线性投影将在欧式空间相近的数据点映射到相似的编码中.理论上,随着映射后的编码位数的增加,编码间的距离应该会越来越接近相应 2 个数据点的欧式距离.但是在为 LSH 设计哈希函数时,如果函数选择不恰当,在实际应用中往往会导致产生非常长且低效的编码[21,35].为了解决这个问题,需要研究通过机器学习的方法去寻找一个对数据更敏感的哈希函数,如[35]提出的利用受限波特曼机来压缩二进制编码从而加速文档的搜索,类似的研究还有文献[33,36]等.文献[21]中提出了一个新技术叫作谱哈希函数(SPH),它展示了在较少的二进制编码下能很好地找到相似的数据项.STH[37]则用了一个不同的谱方法,但两者利用了谱树分区.文献[38]中提出一种基于 p-稳定分布方法来优化 LSH 鲁棒性.针对 LSH 改进和优化的研究有很多如文献[39-42]等.此外,文献[43]提出了一种哈希距离的新方法,从而令相似搜索中距离测量与距离函数定义无关.

多数基于 LSH 方法通常都是假设需要哈希的数据是多维的且是已知并可计算的.例如 SPH 为了处理查询文档,它假设了数据在超矩形中是一致性分布的,在文献[21]中则假设输入向量是服从一个已知的概率分布.为此,本文研究提出一种基于核化 LSH 的方法,可以在不需要对输入数据向量的分布或可计算

等假设下对大规模文档中进行快速相似性检索.

2　背景介绍

2.1　向量空间模型(VSM)

在文档相似性搜索中,任何一个文档都可以利用向量空间模型(Vector Space Model, VSM)[24]表示成一系列的关键字,即利用多关键字来代表一个文档的特征,然后将这些关键字排列转化成向量,且向量中的每个元素的值代表着这个关键字在这篇文档的重要性.图 1 中给出了一个利用“computer”“network”和“peer-to-peer”3 个关键字代表的文档样例,其中向量 $\boldsymbol{v}=(2.5,5,0.5)$ 中每个元素为对应关键字在这篇文档的权重,而向量本身代表着这个文档的特征.在实际应用中,文档的关键字数量非常大,如果用矩阵来代表文档,则这个矩阵往往是几万维的稀疏矩阵.

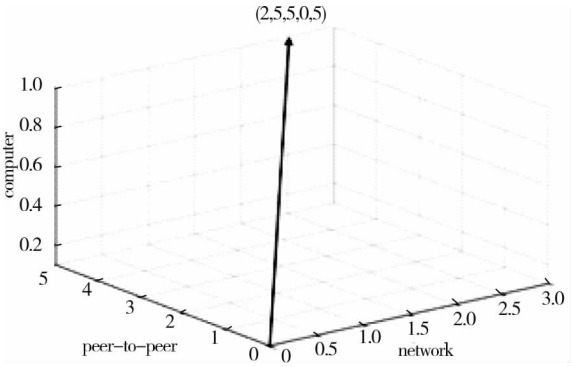

图 1　利用三维关键字空间来表示文档的模型案例

通过利用 VSM 技术,所有的文档和查询都可以被表示成向量的形式,向量的每一个元素代表着相应关键字在文档的权重系数,而这个系数是通过计算该关键字在文档中出现的频率得出的.实际应用中为了更好地表示关键字所代表文档的权重,一般都使用 TF-IDF[6]来得出这个权重系数,即权重系数值用词频(Term Frequency)和 IDF 逆向文件频率(Inverse Document Frequency)的乘积来表.TF-IDF 的主要思想是如果某个词或短语在一篇文章中出现的频率高,并且在其他文章中很少出现,则认为此词或者短语具有很好的类别区分能力,适合用来分类.

最后,在查询过程中,算法按照与查询文档的相似性进行排序返回相似性高的文档作为结果,其中常见的相似性测量方法有 Jaccard 散度、夹角余弦以及距离等.

2.2　局部敏感哈希(LSH)

根据 VSM 模型所有的文档都可以被表达为一个 d 维向量 $\boldsymbol{x}\in\boldsymbol{R}^d$,为此,一个相似搜索问题可以被表达为如何在给定的数据集 $\boldsymbol{X}=\{x_1,x_2,\cdots,x_n\}$ 中,根据给定的一个查询向量从数据集中查找到最相近的向量.LSH 的主要思想是通过应用多个哈希函数到数据集的各个向量中,使得相邻的元素经过多次哈希后的值相近的概率高,而不相邻的元素经过多次哈希后的值相近的概率低.例如,将数据投影到低维二进制海明空间,即任何一个数据被哈希成 b 个二进制位的向量,如果哈希函数选择恰当,在原始空间相邻的数据在低维空间中的海明距离就越近,相隔较远的数据在低维空间中的海明距离就越远.为此,低维空间的向量是通过应用 b 个二进制的哈希函数 $h_1,h_2,\cdots,h_n$ 到原始数据中得来的.LSH 的正式定义为

令 $M=(X,\Delta)$ 为任何距离空间,其中若 $\boldsymbol{q}\in X$,以 $\boldsymbol{q}$ 为中心 r 为半径的范围内区域为 $B(\boldsymbol{q},r)=\{\boldsymbol{x}\in\boldsymbol{X}\mid\Delta(\boldsymbol{x},\boldsymbol{q})\leqslant r\}$,则称 $H=\{h:\boldsymbol{S}\rightarrow U\}$ 为具有 (r_1,r_2,ρ_1,ρ_2) 敏感性的哈希函数集,如果对于所有的 $\boldsymbol{\eta},\boldsymbol{q}\in X$,满足

- 若 $\boldsymbol{\eta}\in B(\boldsymbol{q},r_1)$,则 $Pr_H[h(\boldsymbol{\eta})=h(\boldsymbol{q})]\geqslant\rho_1$;
- 若 $\boldsymbol{\eta}\notin B(\boldsymbol{q},r_2)$,则 $Pr_H[h(\boldsymbol{\eta})=h(\boldsymbol{q})]\geqslant\rho_2$.

式中:$\rho_1>\rho_2$ 且 $r_1>r_2$.

以上定义中,函数 Δ 用于表示 2 个数据样本的距离或相似度,它可以是 L_p 距离、马氏距离、Jaccard 相似度、夹角余弦等.简而言之,一个有效哈希函数 h 需要满足以下局部敏感的哈希属性:

$$Pr[h(\boldsymbol{\eta})=h(\boldsymbol{q})]=\Delta(\boldsymbol{\eta},\boldsymbol{q}).\tag{1}$$

局部敏感哈希的主要思想是将相邻的点尽可能高概率的映射到一样的哈希桶里面.Charikar[44]给出了一个基于随机投影的二值哈希编码方法,其定义为

$$h_w(\boldsymbol{x}) = \begin{cases} 1, \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} > 0; \\ 0, \text{其他}. \end{cases} \tag{2}$$

式中：$\boldsymbol{w}$ 是和 $\boldsymbol{x}$ 具有相同维度的，且服从零均值高斯分布 $N(0,I)$ 的随机超平面.这个超平面将整个空间分成了 2 个部分，并分别赋予 1 和 0 的哈希值.为此，只要从 $N(0,I)$ 中选择一个向量 $\boldsymbol{w}$，然后对每个数据点 $\boldsymbol{x}$ 计算 $\mathrm{sign}(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{w})$ 的值，就得到该数据样本的一个特征比特位.通过重复以上步骤，即随机选取 b 个向量，就构成了 b 个哈希函数.对所有数据样本应用这 b 个哈希函数便得到了一个 b 位的特征向量来代表这个数据样本.在文中还证明了这个方法具有保角的性质，即

$$\Pr[h_w(\boldsymbol{\eta}) = h_w(\boldsymbol{q})] = 1 - \frac{\theta(\boldsymbol{\eta},\boldsymbol{q})}{\pi}. \tag{3}$$

式中：θ 为 2 个向量的夹角，用内积表达为

$$\Pr[\mathrm{sign}(\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{w}) = \mathrm{sign}(\boldsymbol{q}^{\mathrm{T}}\boldsymbol{w})] = 1 - \frac{1}{\pi}\cos^{-1}\frac{\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{q}}{\|\boldsymbol{\eta}\|\|\boldsymbol{q}\|}. \tag{4}$$

3 核化 LSH 文档相似搜索

SVM 作为一种机器学习算法已经被广泛应用到各个领域，其特点是可以在不直接计算复杂的非线性变换，而是计算非线性变换的点积即核函数来得到运算结果.针对很多低维线性不可分的情况，应用核方法可以很好解决问题，常用的核函数有线性核，多项式核，RBF 核等.核化局部敏感哈希(KLSH)则是通过将局部敏感哈希泛化到核空间[45]，即通过将数据映射到高维甚至无限维核空间后，通过定义核矩阵来计算数据高维空间中的相似度.

在前文函数 d 可以表达为 2 个数据点的距离或相似性，根据上式在本文中将 d 定义如下：

$$d(\boldsymbol{\eta},\boldsymbol{q}) = \frac{\boldsymbol{\varphi}(\boldsymbol{\eta})^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{q})}{\|\boldsymbol{\varphi}(\boldsymbol{\eta})\|\|\boldsymbol{\varphi}(\boldsymbol{q})\|} = \frac{\kappa(\boldsymbol{\eta},\boldsymbol{q})}{\sqrt{\kappa(\boldsymbol{\eta},\boldsymbol{\eta})}\sqrt{\kappa(\boldsymbol{q},\boldsymbol{q})}}. \tag{5}$$

式中：$\boldsymbol{\varphi}$ 为映射函数用于将数据非线性的映射到核空间.

针对上式，应用 LSH 最大的难度在于如何从一个服从 $N(0,I)$ 分布中构造一个随机向量 $\boldsymbol{w}$ 使得 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$ 可以通过核函数计算出来.因此，核 LSH 的主要思想是将 $\boldsymbol{w}$ 作为数据集的一个子集的权重系数和，并通过近似构造方法来使得所选取的哈希函数可以通过核函数来计算，且需保证 $\boldsymbol{w}$ 近似于高斯分布.考虑核空间上任何一个点 $\boldsymbol{\varphi}(\boldsymbol{x})$ 为一个服从分布 D 且均值和方差分别为 μ 和 Σ 的向量.根据中心极限定理[15]，从中选取 n 个样本数据构成的集合 $\boldsymbol{Z}_n$，如果 n 足够大，则随机向量：

$$\tilde{\boldsymbol{s}} = \sqrt{n}\left(\frac{1}{n}\sum_{x_i \in \boldsymbol{Z}_n}\boldsymbol{\varphi}(x_i) - \mu\right). \tag{6}$$

同样服从 $N(0,\Sigma)$ 分布，通过变换，可知变量 $\sqrt{\Sigma}\,\tilde{\boldsymbol{s}}$ 将服从 $N(0,I)$ 分布，而这个分布刚好满足 Charikar 所提出的哈希函数中 $\boldsymbol{w}$ 的分布条件.因此，本文将从这个条件下构造随机变量，及令 $\boldsymbol{w} = \sqrt{\Sigma}\,\tilde{\boldsymbol{s}}$，从而 $h(\boldsymbol{\varphi}(\boldsymbol{x}))$ 可以定义为

$$h(\boldsymbol{\varphi}(\boldsymbol{x})) = \begin{cases} 1, \boldsymbol{\varphi}(\boldsymbol{x})^{\mathrm{T}}\sqrt{\Sigma}\,\tilde{\boldsymbol{s}} > 0; \\ 0, \text{其他}. \end{cases} \tag{7}$$

由于式(7)中，核空间的协方差矩阵和均值 μ 都是未知的，我们需要通过样本集来近似构造，如选择 p 个核空间样本 $\boldsymbol{\varphi}(x_1)$，$\boldsymbol{\varphi}(x_2)$，$\cdots$，$\boldsymbol{\varphi}(x_p)$，其均值为 $\mu = \sum_{i=1}^{p}\boldsymbol{\varphi}(x_i)$，同理其方差也可以通过这 p 个样本来近似.当然，由于映射未知，因此，这 2 个值也是无法计算出来，我们将在后面对这些值进行求解.利用核 PCA[23] 中所使用的特征值方法，令 $\Sigma = \boldsymbol{V}\Lambda\,\boldsymbol{V}^{\mathrm{T}}$，得到：

$$h(\boldsymbol{\varphi}(\boldsymbol{x})) = \mathrm{sign}(\boldsymbol{\varphi}(\boldsymbol{x})^{\mathrm{T}}\sqrt{\Sigma}\,\tilde{\boldsymbol{s}}) = \mathrm{sign}(\boldsymbol{\varphi}(\boldsymbol{x})^{\mathrm{T}}V\sqrt{\Lambda}\,V^{\mathrm{T}}\tilde{\boldsymbol{s}}). \tag{8}$$

另外,在所选的 p 样本中构造核矩阵并进行特征分解得 $K = \boldsymbol{U}\Theta\ \boldsymbol{U}^{\mathrm{T}}$, 根据核 PCA 的推导可以知道 Λ 和 Θ 的非零特征值是相等的,为此,选取协方差 Σ 和核矩阵 K 的第 m 个特征向量 $\boldsymbol{v}_m$ 和 $\boldsymbol{u}_m$, 可以得到 $\boldsymbol{\varphi}(\boldsymbol{x})$ 在 $\boldsymbol{v}_m$ 特征向量下的投影为

$$\boldsymbol{v}_m^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{x}) = \sum_{i=1}^{p} \frac{u_m(i)}{\sqrt{\theta_m}}\boldsymbol{\varphi}(x_i)^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{x}). \tag{9}$$

将式(8)代入式(9),得到:

$$h(\boldsymbol{\varphi}(\boldsymbol{x})) = \boldsymbol{\varphi}(\boldsymbol{x})^{\mathrm{T}}\boldsymbol{V}\sqrt{\Lambda}\,\boldsymbol{V}^{\mathrm{T}}\tilde{\boldsymbol{s}} = \sum_{k=1}^{p}\frac{1}{\sqrt{\theta_m}}v_k^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{x})\ v_k^{\mathrm{T}}\tilde{\boldsymbol{s}} = \sum_{m=1}^{p}\frac{1}{\sqrt{\theta_m}}\sum_{i=1}^{p}\frac{u_m(i)}{\sqrt{\theta_m}}\boldsymbol{\varphi}(x_i)^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{x})\ \sum_{j=1}^{p}\frac{u_m(j)}{\sqrt{\theta_m}}$$

$\boldsymbol{\varphi}(x_j)^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{x})\ \tilde{\boldsymbol{s}}$.

将其规范后得

$$h(\boldsymbol{\varphi}(x)) = \sum_{m=1}^{p}\sum_{i=1}^{p}\sum_{j=1}^{p}\frac{u_m(i)\ u_m(j)}{\theta_m^{\frac{3}{2}}}\boldsymbol{\varphi}(x_i)^{T}\boldsymbol{\varphi}(\boldsymbol{x})\ \boldsymbol{\varphi}(x_j)^{T}\tilde{s} = \sum_{i=1}^{p}\sum_{j=1}^{p}\boldsymbol{\varphi}(x_i)^{T}\boldsymbol{\varphi}(\boldsymbol{x})\ \boldsymbol{\varphi}(x_j)^{T}\tilde{s}\sum_{m=1}^{p}$$

$\dfrac{u_m(i)\ u_m(j)}{\theta_m^{\frac{3}{2}}}$.

由于 $K_{ij}^{-3/2} = \sum_{m=1}^{p}\dfrac{u_m(i)\ u_m(j)}{\theta_m^{\frac{3}{2}}}\boldsymbol{\varphi}(x_j)^{\mathrm{T}}\tilde{\boldsymbol{s}}$,故上式可以简化为

$$h(\boldsymbol{\varphi}(\boldsymbol{x})) = \sum_{i=1}^{p}\sum_{j=1}^{p}K_{ij}^{-3/2}(\boldsymbol{\varphi}(x_i)^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{x}))\ (\boldsymbol{\varphi}(x_j)^{\mathrm{T}}\tilde{\boldsymbol{s}}) = \sum_{i=1}^{p}\Big(\sum_{j=1}^{p}K_{ij}^{-3/2}\boldsymbol{\varphi}(x_j)^{\mathrm{T}}\tilde{\boldsymbol{s}}\Big)\ (\boldsymbol{\varphi}(x_i)^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{x}))\ =$$

$$\sum_{i=1}^{p}w(i)(\boldsymbol{\varphi}(x_i)^{\mathrm{T}}\boldsymbol{\varphi}(\boldsymbol{x}))\ . \tag{10}$$

式中: $w(i) = \sum_{j=1}^{p}K_{ij}^{-3/2}\boldsymbol{\varphi}(x_j)^{\mathrm{T}}\tilde{\boldsymbol{s}}$,因此,我们所寻找的随机向量可以表示成为 $\sum_{i=1}^{p}w(i)\boldsymbol{\varphi}(x_i)$,即可以看成从核样本数据中对数据加权求和.最后,将 $\tilde{\boldsymbol{s}} = \sqrt{n}\left(\dfrac{1}{n}\sum_{x_i\in\boldsymbol{Z}_n}\boldsymbol{\varphi}(x_i) - \mu\right)$ 代入到 $w(i)$,并令 μ 为 0 可以得到:

$$w(i) = \sum_{j=1}^{p}K_{ij}^{-3/2}\boldsymbol{\varphi}(x_j)^{\mathrm{T}}\sqrt{n}\left(\frac{1}{n}\sum_{x_i\in\boldsymbol{Z}_n}\boldsymbol{\varphi}(x_i) - \mu\right) = \sum_{j=1}^{p}\sum_{x_i\in\boldsymbol{Z}_n}\frac{1}{\sqrt{n}}K_{ij}^{-3/2}\boldsymbol{\varphi}(x_j)^{\mathrm{T}}\boldsymbol{\varphi}(x_i)\ . \tag{11}$$

最后, $w(i)$ 式子简化为

$$w(i) = \sum_{j=1}^{p}\sum_{l\in\boldsymbol{Z}_n}K_{ij}^{-3/2}K_{lj}. \tag{12}$$

此处省略了 $\dfrac{1}{\sqrt{n}}$,是因为它对 sign 函数运算结果没有影响.

最后,归纳一下整个 KLSH 的流程:首先,选择 p 个样本数据点并构造这些数据的核矩阵;其次,构建这些数据的哈希表,即从样本数据点中选取 n 个随机样本计算 $w(i)$ 的值;然后计算 $h(\boldsymbol{\varphi}(x))$ 的值得到二进制编码;最后,对任何一个查询,使用同样的方法得出相应哈希值,并用这个哈希值在哈希表中进行最近邻查询.

4　实验与分析

4.1　数据集

我们选择了 3 个真实环境的数据集 Reuters21578,20Newsgroups 和 TDT2 作为实验的数据,数据来源及预处理都来自文献[37].

Reuters21578 是路透社 1987 年中新闻专线的文档集合.它包含了 21 578 样本,分成 135 个类别.为了

实验方便我们剔除了属于多个类别的样本,并从中随机选取了 10 个类别集合作为实验数据.

20Newsgroups 文档集合包含了 18 846 个文档,平均的分布为 20 个类别,为了减少实验运算时间,我们随机选取了其中 6 个类别作为实验数据.

TDT2(NIST Topic Detection and Tracking)文档集包含了来自 6 个不同新闻源的文档集合,总共有 11 201个文档数据,并被分成 96 个语义类别.为了简化实验结果,我们剔除了属于多个类别的文档数据,最后只保留了 9 394 个文档,并从中随机选取了 6 个类别作为实验数据.

所有的数据集合都按照文献[37]的方法做了预处理,包括剔除无用词语、应用波特词干算法和 TF-IDF 权重计算.最后都按照 0.8 的比例划分训练集和测试集.

4.2 评估方法

给定一个数据集,实验将测试数据中的任何文档作为一个查询去从训练集合中检索相似的文档,并用海明距离来计算文档近邻的排序.其评价的标准主要为根据检索回来的文档的数量来评价它的准确率与召回率.其定义如下:

$$\text{准确率} = \frac{\text{检索回来与测试样本相关的文档数量}}{\text{检索回来所有的文档数量}};$$

$$\text{召回率} = \frac{\text{检索回来与测试样本相关的文档数量}}{\text{所有与测试样本相关的文档数量}}.$$

4.3 结果与分析

图 2 给出了本文方法在 3 个文档集合中的测试情况,实验采取了最简单的线性核函数 $\kappa(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{y}$. 可以看出此处随着要求返回文档数量的增加准确率稍微降低,并相对比较稳定.此外,用于近似高斯分布的样本数 n 的值的变化对准确率的影响并不明显.所提出的方法在 Reuters21578 和 TDT2 集合返回的前 100 个文档中,具有非常高的准确率.

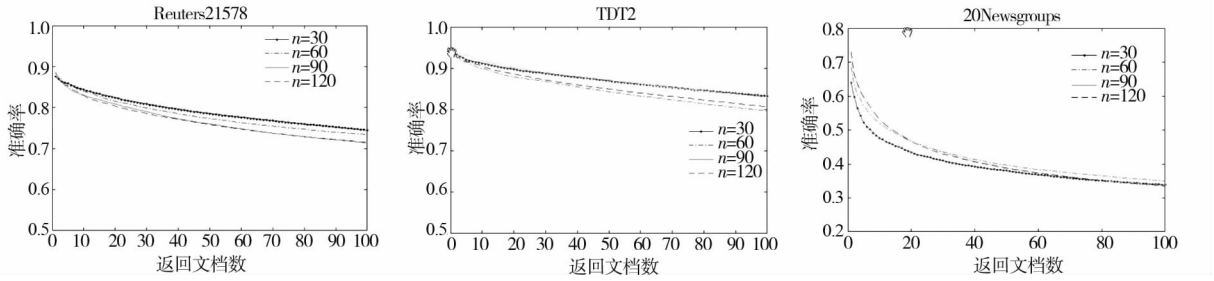

图 2 检索文档的准确率随着返回文档数的增加的变化情况

本文的方法允许 LSH 使用任意的核函数来对哈希后的文档进行快速相似搜索,为此,图 3 中给出了采用不同核函数对 Reuters21578 文档集合测试的结果.其中,Gaussian 核函数形式为 $\kappa(\boldsymbol{x},\boldsymbol{y}) = \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{y}\|^2}{2\sigma^2}\right)$,实验中 $2\sigma^2$ 的值设为文档类别数.Polynomial 核函数和 Sigmoid 核函数的形式分别为 $\kappa(\boldsymbol{x},\boldsymbol{y}) = (\alpha\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y} + c)^d$ 和 $\kappa(\boldsymbol{x},\boldsymbol{y}) = \tanh(\alpha\boldsymbol{x}^{\mathrm{T}}\boldsymbol{y} + c)$,实验中 α 设为文档类别数的倒数,c 取值为 0,而 d 取值为 2.从实验结果得出,选取多项式核函数(Polynomial Kernel)对 Reuters21578 文档集合进行实验得到比较高的准确率.

图 4 给出了 3 个数据集随着召回率增加而准确率逐渐减少的变化情况,实验中采用了线性核函数,n 取 30.

在算法时间复杂度方面,局部敏感哈希方法通过哈希函数映射变换操作将原始数据集合分成了多个子集合,而每个子集合中的数据间是相邻的且该子集合中的元素个数相对原始大集合少很多.因此,通过将一个超大集合的相邻函数的线性查找转化为在一个很小的集合内查找相邻元素的问题,大大降低了计

算量.核化局部敏感哈希所使用的哈希函数的参数如 $\boldsymbol{w}$ 都是通过离线计算得出,因此和其他哈希函数具有相同的运算效率.

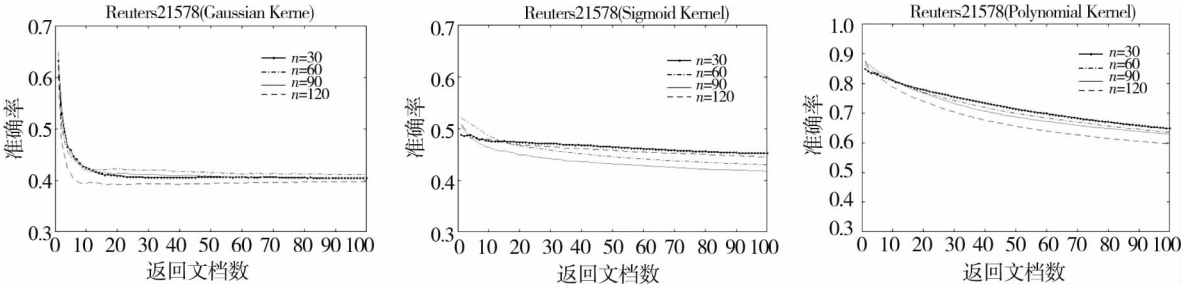

图 3　选用不同类型核函数对 Reuters21578 文档集的测试情况

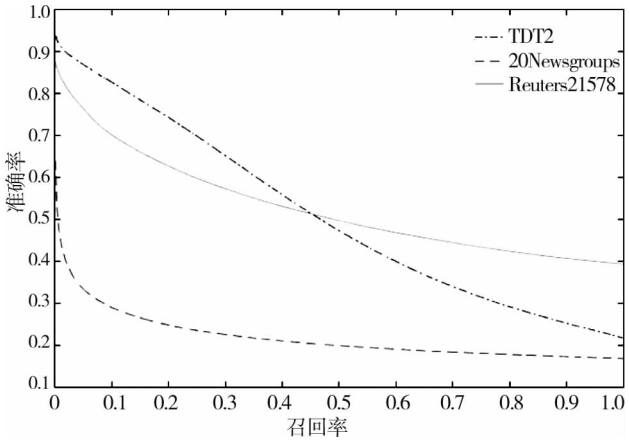

图 4　准确率随着召回率增加的变化情况

5　结论

本文提出了一种核化局部敏感哈希的文本搜索算法,该算法可以使用任意的核函数来表达哈希函数.

1)可以在不知道特征空间的情况下使用基于局部敏感哈希的搜索算法,从而拓宽了局部敏感哈希算法的应用.

2)可以在保证时间效率的情况下,利用任意的核函数对大规模的文档进行快速搜索.

3)本身对数据的分布是没有任何要求的,因此可以很直接的被推广到其他搜索类别的领域如图像搜索等.

因此,该算法对不同类型的大规模数据的特征索引具有实际的应用意义.

参考文献:

[1] Gantz J, Reinsel D. The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east[J]. IDC iView: IDC Analyze the future, 2012, 2007 (2012): 1-16.

[2] Gionis A, Indyk P, Motwani R, et al. Similarity search in high dimensions via hashing[C]//Vldb, 1999, 99(6): 518-529.

[3] Yu C, Ooi B C, Tan K L, et al. Indexing the distance: An efficient method to knn processing[C]//Vldb, 2001, 1: 421-430.

[4] Lv Q, Josephson W, Wang Z, et al. Intelligent probing for locality sensitive hashing: multi-probe lsh and beyond[J]. Proceedings of the VLDB Endowment, 2017, 10 (12): 2021-2024.

[5] Bhattacharya I, Kashyap S R, Parthasarathy S. Similarity searching in peer-to-peer databases[C]//25th IEEE International Conference on Distributed Computing Systems (ICDCS'05). IEEE, 2005: 329-338.

[6] Manning C, Raghavan P, Schütze H. Introduction to information retrieval[J]. Natural Language Engineering, 2010, 16 (1):

100-103.

[7] Henzinger M. Finding near-duplicate web pages: a large-scale evaluation of algorithms[C]//Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2006: 284-291.

[8] Stein B, zu Eissen S M, Potthast M. Strategies for retrieving plagiarized documents[C]//Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2007: 825-826.

[9] Koren Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model[C]//Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2008: 426-434.

[10] Pandey S, Broder A, Chierichetti F, et al. Nearest-neighbor caching for content-match applications[C]//Proceedings of the 18th international conference on World wide web. ACM, 2009: 441-450.

[11] Lew M S, Sebe N, Djeraba C, et al. Content-based multimedia information retrieval: State of the art and challenges[J]. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 2006, 2(1): 1-19.

[12] Indyk P, Motwani R. Approximate nearest neighbors: towards removing the curse of dimensionality[C]//Proceedings of the thirtieth annual ACM symposium on Theory of computing. ACM, 1998: 604-613.

[13] Andoni A, Indyk P. Near - optimal hashing algorithms for approximate nearest neighbor in high dimensions[J]. Communications of the ACM, 2008, 51 (1): 117.

[14] Datar M, Immorlica N, Indyk P, et al. Locality-sensitive hashing scheme based on p-stable distributions[C]//Proceedings of the twentieth annual symposium on Computational geometry. ACM, 2004: 253-262.

[15] Panigrahy R. Entropy based nearest neighbor search in high dimensions[C]//Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm. Society for Industrial and Applied Mathematics, 2006: 1186-1195.

[16] Joly A, Buisson O. A posteriori multi-probe locality sensitive hashing[C]//Proceedings of the 16th ACM international conference on Multimedia. ACM, 2008: 209-218.

[17] Huang Q, Ma G, Feng J, et al. Accurate and fast asymmetric locality-sensitive hashing scheme for maximum inner product search[C]//Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, 2018: 1561-1570.

[18] Athitsos V, Alon J, Sclaroff S, et al. Boostmap: A method for efficient approximate similarity rankings[C]//Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004. IEEE, 2004: II-II.

[19] Hu D, Nie F, Li X. Discrete spectral hashing for efficient similarity retrieval[J]. IEEE Transactions on Image Processing, 2018, 28 (3): 1080-1091.

[20] Salakhutdinov R, Hinton G. Semantic hashing[J]. RBM, 2007, 500 (3): 500.

[21] Weiss Y, Torralba A, Fergus R. Spectral hashing[C]//Advances in Neural Information Processing Systems, 2009: 1753-1760.

[22] Rice J A. Mathematical statistics and data analysis[M]. Cengage Learning, 2006.

[23] Tang C, Xu Z, Dwarkadas S. Peer-to-peer information retrieval using self-organizing semantic overlay networks[C]// Proceedings of the 2003 conference on Applications, Technologies, Architectures, and Protocols for Computer Communications. ACM, 2003: 175-186.

[24] Salton g, Wong A, Yang C S. A vector space model for automatic indexing[J]. Communications of the ACM, 1975, 18 (11): 613-620.

[25] Berry M W, Drmac Z, Jessup E R. Matrices, vector spaces, and information retrieval[J]. SIAM review, 1999, 41 (2): 335-362.

[26] Friedman J. An algorithm for finding best matches in logarithmic expected time[R]. SLAC National Accelerator Lab., Menlo Park, CA (United States), 2018.

[27] Leiserson C E, Rivest R L, Cormen T H, et al. Introduction to algorithms[M]. Cambridge, MA: MIT press, 2001.

[28] Satuluri V, Parthasarathy S. Bayesian locality sensitive hashing for fast similarity search[J]. Proceedings of the VLDB

Endowment, 2012, 5 (5): 430-441.

[29] Berry M W, Dumais S T, O'Brien G W. Using linear algebra for intelligent information retrieval[J]. SIAM Review, 1995, 37 (4): 573-595.

[30] Deerwester S, Dumais S T, Furnas G W, et al. Indexing by latent semantic analysis[J]. Journal of the American Society for Information Science, 1990, 41 (6): 391-407.

[31] He X, Cai D, Liu H, et al. Locality preserving indexing for document representation[C]//Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2004: 96-103.

[32] He X, Niyogi P. Locality preserving projections[C]//Advances in Neural Information Processing Systems, 2004: 153-160.

[33] Hinton G E, Salakhutdinov R R. Reducing the dimensionality of data with neural networks[J]. Science, 2006, 313 (5786): 504-507.

[34] Zhang D, Wang J, Cai D, et al. Laplacian co-hashing of terms and documents[C]//European Conference on Information Retrieval. Springer, 2010: 577-580.

[35] Hinton G E, Osindero S, Teh Y W. A fast learning algorithm for deep belief nets[J]. Neural Computation, 2006, 18 (7): 1527-1554.

[36] Shakhnarovich G, Viola P, Darrell T. Fast pose estimation with parameter sensitive hashing[J]. 2003.

[37] Zhang D, Wang J, Cai D, et al. Self-taught hashing for fast similarity search[C]//Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2010: 18-25.

[38] 李淼, 孙荣坤, 韩纪庆. 基于 p-稳定分布局部敏感哈希地址的鲁棒音频检索方法[J]. 信号处理, 2012 (3): 367-375.

[39] Indyk P. Near optimal hashing algorithms for approximate near (est) neighbor problem[C]//MMDS 2006. Workshop on Algorithms for Modern Massive Data Sets, Stanford, USA, 2006.

[40] Bawa M, Condie T, Ganesan P. Lsh forest: self-tuning indexes for similarity search[C]//Proceedings of the 14th International Conference on World Wide Web. ACM, 2005: 651-660.

[41] Lv Q, Josephson W, Wang Z, et al. Multi-probe lsh: efficient indexing for high-dimensional similarity search[C]//Proceedings of the 33rd International Conference on Very Large Data Bases. VLDB Endowment, 2007: 950-961.

[42] 曹玉东, 刘福英, 蔡希彪. 基于局部敏感哈希算法的图像高维数据索引技术的研究[J]. 辽宁工业大学学报:自然科学版, 2013, 33 (1): 1-3.

[43] Athitsos V, Potamias M, Papapetrou P, et al. Nearest neighbor retrieval using distance-based hashing[C]//2008 IEEE 24th International Conference on Data Engineering. IEEE, 2008: 327-336.

[44] Charikar M S. Similarity estimation techniques from rounding algorithms[C]//Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing. ACM, 2002: 380-388.

[45] Kulis B, Grauman K. Kernelized locality-sensitive hashing[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2011, 34 (6): 1092-1104.