基于哈希技术和MapReduce的大数据集K-近邻算法

翟俊海 张明阳 王婷婷 郝 璞
(河北大学数学与信息科学学院 保定 071002)

摘 要 K-近邻是一种著名的分类算法。由于简单且易于实现,因此其被广泛应用于许多领域,如人脸识别、基因分类、决策支持等。然而,在大数据环境中,K-近邻算法变得非常低效,甚至不可行。针对这一问题,提出了一种基于哈希技术和MapReduce的大数据集K-近邻分类算法。为了验证算法的有效性,在4个大数据集上进行了实验,结果显示,在保持分类能力的前提下,所提算法可以大幅度地提高K-近邻算法的效率。

关键词 K-近邻,哈希技术,分类算法,大数据集
中图法分类号 TP181 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.07.037

K-Nearest Neighbor Algorithm Based on Hash Technology and MapRecuce

ZHAI Jun-hai ZHANG Ming-yang WANG Ting-ting HAO Pu
(College of Mathematics and Information Science, Hebei University, Baoding 071002, China)

Abstract K-nearest neighbor(K-NN) is a famous classification algorithm. Because the idea of K-NN is simple and it is easy to implement, K-NN has been widely applied to many fields, such as face recognition, gene classification and decision making, etc. However, in the big data environment, the efficiency of K-NN is very low, even it is not workable. In order to deal with this problem, based on hash technology and MapRecuce, this paper proposed an improved K-nearest neighbor algorithm. In order to verify the effectiveness of the proposed algorithm, some experiments were conducted on 4 big data sets. The experimental results show that the proposed algorithm is effective and efficient.

Keywords K-nearest neighbor, Hash technology, Classification algorithms, Big data sets

1 引言

K-近邻(K-Nearest Neighbor, K-NN)[1]是求解分类问题的著名算法,它是十大数据挖掘算法之一。K-NN算法的基本思想非常简单:对于给定的测试样例x,在训练集中寻找它的K个最近邻样例,并根据这K个最近邻来确定x的类别。应用K-NN算法时,需要注意3个基本问题:1)针对具体的分类问题,需要选择一个合适的距离度量或相似性度量来衡量待测样例与测试集中样例之间的距离;2)参数K的确定;3)用来确定测试样例类别的方法。一般地,在实际应用中,最常用的距离度量是欧氏距离。常用来确定测试样例x类别的方法是统计其K个近邻类别的方法,得票数最多的类别即为测试样例x的类别。关于参数K的最优取值,理论上没有统一的结论。如果K太小,分类结果将会对噪声非常敏感;如果K太大,测试样例x的近邻可能包含许多其他类的样例点。对于给定的问题,可用交叉验证的方法确定合适的K值。

当$K=1$时,K-NN算法被称为最近邻算法。关于最近邻算法,Cover和Hart证明了以下结论成立[2-3]:最近邻算法的渐近错误率最坏不会超过两倍的贝叶斯错误率,而最好则有可能接近或达到贝叶斯错误率。

K-NN算法的计算时间复杂度和空间复杂度均为$O(n)$,其中n为训练集中包含的样例数。对于中小型数据集分类问题,K-NN算法的效率是很高的;但若要在大数据环境中找到测试样例x的K个近邻,则需计算它到训练集中每一个样例之间的距离,此时K-NN算法会变得非常低效,甚至不可行。针对这一问题,人们进行了深入的研究,并提出了许多方法来提高K-NN算法的效率,例如采用分支限界法[4-6]将训练集分级划分成多个子集,形成一个树状结构,每个节点代表一个子集,每个子集只用几个较少的样例来代表,通过把测试样例按顺序与各个节点进行比较来排除不可能包含最近邻的子集,只在最后的节点上才需要与每一个样例进行比较,详细的算法描述可参见文献[3]。基于样例选择的思想,Hart提出了压缩近邻选择算法[7],它的基本思想是把靠近分类边界即容易被错误分类的样例从训练集中选择出来,用以代替原训练集分类测试样例。沿着这一技术路线,研究人员提出了许多能提高K-NN算法效率的样例选择算法,文献[8-10]对样例

到稿日期:2016-06-16 返修日期:2016-10-05 本文受国家自然科学基金项目(71371063),河北省自然科学基金项目(F2017201026),河北省高等学校科学技术研究重点项目(ZD20131028),河北大学研究生创新资助项目(X2016059)资助。

翟俊海(1964-),男,博士,教授,CCF会员,主要研究方向为机器学习、数据挖掘;张明阳(1991-),男,硕士生,主要研究方向为云计算与大数据处理;王婷婷(1991-),女,硕士生,主要研究方向为云计算与大数据处理;郝 璞(1992-),男,硕士生,主要研究方向为云计算与大数据处理。

选择算法进行了很好的综述。

对于许多实际应用问题，寻找测试样例的近似近邻集合就已足够。过去十多年，基于树的搜索方法（如 KD-树[11]、Ball-树[12]、Metric-树[13]等）是寻找近似近邻的主流方法。然而，基于树的方法需要的辅助存储空间非常大；另外，当处理高维数据时，这些方法的性能会急剧下降。最近几年，基于哈希技术的方法受到了研究人员的广泛关注。哈希技术[14]是一种映射技术，它通过一个哈希函数将样例集合从原空间映射到 Hamming 空间，要求在原空间相似的样例在 Hamming 空间中依然相似，即哈希函数要满足保持相似性的性质。由于在 Hamming 空间中每一个样例被一个 0-1 串表示，因此计算两个样例点之间的距离（Hamming 距离）时只需做简单的异或操作即可，这样，经过合适的哈希变换，不仅可以大幅度地提高计算的效率，而且可以大幅度地降低存储所需的空间。根据哈希函数是否从数据中学习得到，哈希方法可分为独立于数据的哈希方法[15]和基于数据的哈希方法[16-17]，后者又称为哈希学习方法。文献[18-19]对哈希方法进行了较好的综述，非常具有参考价值。

本文针对 K-NN 算法在大数据集分类中遇到的效率降低甚至不可行的问题，提出了一种基于哈希技术和 MapReduce 的大数据集 K-近邻分类算法，并在 4 个大数据集上进行了实验。结果表明，在保持分类能力的前提下，本文提出的算法可以大幅度地提高 K-近邻算法的效率。

2　基础知识

本节介绍将要用到的基础知识，包括 SimHash[20]和 MapReduce[21]。

2.1　SimHash

传统的 Hash 算法的目标是将原始内容尽量均匀且随机地映射为一个签名值，在原理上相当于伪随机数产生算法。传统 Hash 算法产生的两个签名如果相等，则说明原始的两个内容在一定概率上是相等的；如果不相等，除了说明原始的两个内容不相等之外，不再提供任何信息，因为即使原始内容只相差一个字节，所产生的签名也很可能差别极大。从这个意义上讲，设计一个 Hash 算法使得对相似的内容产生的签名也相似是更为艰巨的任务，因为它的签名值除了需要提供原始内容是否相等的信息外，还需要额外提供不相等的原始内容的差异程度的信息。

SimHash 算法是 Google 公司的 Manku 等[20]研究人员提出的一种网页去重的相似性估计方法。SimHash 与普通 Hash 的最大不同在于传统的 Hash 函数虽然也可以用于映射以比较文本是否重复，但是可能会将差距只有一个字节的文档映射成两个完全不同的哈希结果；而 SimHash 对相似文本的哈希映射结果也相似，在 Hamming 空间中 Hash 签名的相似程度也能反映出原空间中样本的相似程度。

SimHash 算法具有思想精巧、容易理解和实现的优点，其输入是一个向量，输出是一个 f 位的签名值。为了陈述方便，假设输入的是一个样本的特征集合 $x_i=(x_{i1},x_{i2},\cdots,x_{id})$。SimHash 算法描述如下：

(1)对于 x_i 的每一个特征分量，用传统的 Hash 算法将其变换成一个 f 位的签名值 b；

(2)如果签名值 b 的第 i 位小于 0，则将其置为 -1，否则将其置为 1；

(3)将 x_i 所有特征分量的变换码按位相加，如果和向量的某一维大于 0，则最终签名的对应位为 1；如果和向量的某一维小于或等于 0，则最终签名的对应位为 0。这就是样本 x_i 经过 SimHash 算法最终映射得到的 f 位签名值。

下面通过一个具体例子来说明 SimHash 算法的执行过程。假设样例 $x_1=(1,2,3,4)$，样例 $x_2=(1,2,3,5)$。对于样例 x_1，SimHash 算法的执行过程如图 1 所示；对于样例 x_2，SimHash 算法的执行过程如图 2 所示。

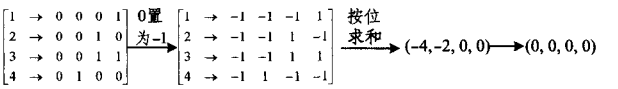

图 1　SimHash 算法的执行过程（对于样例 x_1）

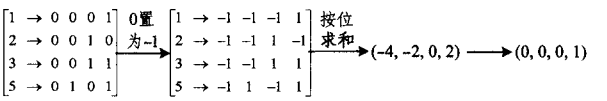

图 2　SimHash 算法的执行过程（对于样例 x_2）

从图 1 和图 2 可以看出，x_1 的 SimHash 值为(0,0,0,0)，x_2 的 simHash 值为(0,0,0,1)。从这个简单的例子可以看出，x_1 和 x_2 在原样例空间中是相似的（只有第 4 个分量不同，一个为 4，一个为 5），经过 SimHash 变换后，它们在 Hamming 空间中也是相似的。

从上面的例子可以看出，SimHash 算法的设计思想简单明了：通过降维，将高维的特征向量映射成一个 f 位的签名值，在 Hamming 空间中两个样本对应 Hash 签名的相似程度就能反映出原空间中两个样本的相似程度，在 Hamming 空间中常用 Hamming 距离来衡量两个样本的相似度。

但是通过实验发现，在 Hamming 空间中样本之间的 Hamming 距离并不能完全精确地表示原样本之间实际的相似程度。之所以存在这样的问题，是因为映射后的二值码损失了较多高维数据的信息，二值码距离难以精确地表达出原始数据之间的关系。因此，本文实验主要是利用 SimHash 算法大幅减小搜索空间，例如计算测试样本 x 的 K-近邻时，只搜索一个特定的近邻空间而非整个庞大的原空间。例如，当 Hash 签名值为 f 位时，则意味着共有 2^f 个 Hash 桶，在计算测试样本 x 的 K-近邻时，可从最近的 Hash 桶中搜索，即从映射之后二值码相同的样例集合中搜索，其次是有 1 位不同的 Hash 桶、2 位不同的 Hash 桶等。然后再从特定的近邻空间中通过欧氏距离精确地计算出测试样本的 K 个精确近邻。

2.2　MapReduce

对于中小型数据集分类问题，K-NN 算法的效率很高；但若要在大数据环境中寻找测试样例 x 的 K 个近邻，则需计算它到训练集中每一个样例之间的距离，此时 K-NN 算法会变得非常低效，甚至不可行。因为 MapReduce 大数据处理技术

是当前最有效、最简单的海量数据处理方式，本文以加快相似度计算速度为目标，使用了 MapReduce[21] 分布式编程模式来实现 K-NN 算法。

MapReduce 是针对大数据处理的一种并行编程框架，它的基本思想包括以下 3 个方面。

(1)MapReduce 采用分治策略自动地将大数据集划分为若干子集，并将这些子集部署到不同的云计算节点上，并行地对数据子集进行处理；

(2)基于函数编程语言 LISP 的思想，MapReduce 提供了两个简单易行的并行编程方法：Map 和 Reduce，用以实现基本的并行计算；

(3)MapReduce 能自动完成许多系统级的处理细节，这些细节包括：

1)计算任务的自动划分和自动部署；

2)自动分布式存储处理的数据；

3)处理数据和计算任务的同步；

4)对中间处理结果数据的自动聚集和重新划分；

5)云计算节点之间的通讯；

6)云计算节点之间的负载均衡和性能优化；

7)云计算节点的失效检查和回复。

MapReduce 处理数据的流程如图 3 所示。

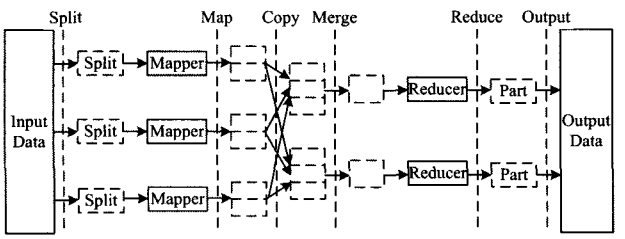

图 3 MapReduce 处理数据的流程示意图

3 基于哈希技术和 MapReduce 的大数据集 K-近邻算法

一般而言，基于哈希技术的近邻查找方法主要包括两个阶段：1)哈希映射，用于将数据点映射为二值码；2)针对二值码设计近似距离，从而进行排序。为了提高基于哈希技术的 K-近邻查找的计算效率和准确率，主要从这两个方面出发来研究如何获得更优质的二值码和更准确的度量距离。本文主要研究内容和创新成果如下：

(1)在数据映射阶段应用 SimHash 算法。该算法通过降维将高维的特征向量映射成一个 f 位的签名值，在 Hamming 空间中两个样本对应 Hash 签名的相似程度能够反映原空间中两个样本的相似程度。

(2)在近邻搜索阶段引入桶的概念，从而大幅度地缩小了近邻搜索空间。例如当 Hash 签名值为 f 位时，意味着共有 2^f 个 Hash 桶，在计算测试样本 x 的 K-近邻时，可从最近的 Hash 桶中搜索，即从映射之后二值码相同的样例集合中搜索；为了进一步提高准确率，也可以再引入 1 位不同的 Hash 桶、2 位不同的 Hash 桶等。

(3)在距离度量阶段采用精确的欧氏距离计算方式。尽管哈希方法具有存储和计算上的优势，但是在 Hamming 空间中样本之间的 Hamming 距离并不能完全精确地表示出原样本之间的实际相似程度，因此本文舍弃了传统上 Hamming 距离的近似距离度量方式。

(4)本文以加快相似度计算速度为目标，使用 MapReduce 分布式编程模式来提高计算的效率。

下面介绍所提出的基于哈希技术和 MapReduce 的大数据集 K-近邻算法，为描述方便，记该算法为 H-MR-K-NN。算法的基本实现简单，但能有效解决 K-NN 算法处理大数据时遇到的问题。

算法的具体实现过程为：对于给定的大数据集，首先，利用 SimHash 算法将大数据集通过 MapReduce 大数据处理技术从原样例空间映射到 Hamming 空间，得到二值码集合；然后，在 Hamming 空间中寻找与测试样例 x 在同一个桶中的训练样例，所谓同一个桶是指映射之后二值码相同的样例的集合；最后，对同一个桶中的训练样例，利用 MapReduce 大数据处理技术，通过计算欧氏距离寻找测试样例 x 的 K 个精确近邻，并用这 K 个精确近邻对 x 进行分类。所提算法的伪代码如算法 1 所示。

算法 1 H-MR-K-NN 算法

输入：大数据集 $D=\{(x\_i,y\_i)\mid x\_i\in R^d, y\_i\in L\}$，$1\leqslant i\leqslant n$；测试样例 x；参数 K

输出：x 的类标

```
For (i=1;i≤n;i=i+1)
{
  利用 SimHash 算法将训练样例 x_i 变换到 Hamming 空间;
}
利用 SimHash 算法将测试样例 x 变换到 Hamming 空间;
For (i=1;i≤n;i=i+1)
{
  在 Hamming 空间中,利用 MapReduce 大数据处理技术寻找与测试样例 x 在同一个桶中的训练样例;
}
从同一个桶中再寻找测试样例 x 的 K 个精确近邻,并用这 K 个精确近邻对 x 进行分类;
输出 x 的类标。
```

算法 1 中 Map 函数和 Reduce 函数的伪代码分别如算法 2 和算法 3 所示。

算法 2 Map 函数

```
输入:(k_1,v_1).
输出:(k_2,v_2).
//利用 setup 函数进行资源的初始化,将训练样例经 SimHash 映射后的信息存储到容器中;
trainSet. add(simtraindata);
//在 Hamming 空间,计算测试样例 x_i 与所有训练样例之间的 Hamming 距离;
For (i=1;i≤n; i=i+1)
{
  For (j=1;j<trainSet. size( );j=j+1)
  {
```

```
    Distance-HaiMingDistance(x_i,trainSet.get(j));
      //若 Hamming 距离为 0,即在同一个桶中,然后计算测试样例
    与训练样例之间的欧氏距离;
    If(HaiMingDistance=0)
    {
      Distance-EuclideanDistance(x_i,trainSet.get(j));
      //将距离测试样例 x_i 的 K 个欧氏近邻存到容器 NearArray-
      List 中;
      NearArrayList.set(Knear-traindata)
    }
  }
//输出测试样例与其相应的 K 个欧氏距离最近的训练样例的标签
  For (i=1;i≤K;i=i+1)
  {
    context.write(testsample,trainLable.get(i));
  }
}
```

输出：⟨k_2,v_2⟩.

算法 3　Reduce 函数

```
输入:⟨k_2,v_2⟩
输出:⟨k_3,v_3⟩
//遍历所有测试样例;
For (i=1; i≤n; i=i+1)
{
  //对于测试样例 x_i,将其前 K 个欧氏近邻的训练样例的类标签添
  加到容器中;
  ArrayList.add(Knear-trainlable);
  //应用 K-近邻算法对测试样例 x_i 进行分类;
  predictlable=MostFrequent(ArrayList);
  //将测试样例与相应的 predictlable 输出;
  context.write(testsample,predictable);
}
```

输出：⟨k_3,v_3⟩

在 Map 函数的伪代码中，⟨k_1,v_1⟩分别为：⟨起始偏移量，测试样例⟩，⟨k_2,v_2⟩分别为：⟨测试样例，训练样例标签⟩。

在 Reduce 函数的伪代码中，⟨k_2,v_2⟩分别为：⟨测试样例，训练样例类标签⟩，⟨k_3,v_3⟩分别为：⟨测试样例，测试样例预测标签⟩。

4　实验结果

为了验证本文提出的算法的有效性，在 4 个 UCI 大数据集上将其与基于 MapReduce 大数据处理技术的 K-近邻算法[22]进行了实验比较。对于每一个大数据集，按 7∶3 的比例将其随机地划分为训练集和测试集，并重复实验 5 次，结果取 5 次实验的平均值。在未来的工作中，还将继续在其他大数据集上做更多的实验。为描述方便，将基于 MapReduce 大数据处理技术的 K-近邻算法记为 MR-K-NN。实验所用的 4 个大数据集的基本信息如表 1 所列。实验源代码请参见网站：http://pan.baidu.com/s/1slWsL4p。

表 1　实验所用数据集的基本信息

数据集	样例个数	属性个数	类别个数
Forest CoverType	581012	54	7
Skin Segmentation	245057	3	2
Poker Hand	1025010	10	7
Statlog	58000	9	6

实验环境是具有 6 个节点的云计算平台，这些节点的基本配置信息如表 2 所列。

表 2　云计算平台节点的基本配置信息

软硬件项目	配置情况
处理器(CPU)	Inter Xeon E5-4603 2.0GHz(双核)
内存	8GB RDIMM
硬盘	1 TB
网卡	Broadcom 5720 QP 1Gb 网络子卡(四端口)
网络设备	华为 S3700 系列以太网交换机
操作系统	Ubuntu kylin 13.04
云计算平台	Hadoop0.20.2
JDK 版本	Jdk-7u71-linux-i586
程序开发环境	Eclipse-Java-luna-SRI-linux

节点的具体规划如表 3 所列。

表 3　云计算平台节点的功能规划

节点号	主机名	IP 地址	节点类型
1	hadoop1	10.187.84.50	NameNode,SecondaryNameNode
2	hadoop2	10.187.84.51	JobTracker
3	hadoop3	10.187.84.52	DataNode,TaskTracker
4	hadoop4	10.187.84.53	DataNode,TaskTracker
5	hadoop5	10.187.84.54	DataNode,TaskTracker
6	hadoop6	10.187.84.55	DataNode,TaskTracker

本文算法 H-MR-K-NN 与 MR-K-NN 在测试精度方面的比较结果如表 4 所列，在运行时间方面的比较结果如表 5 所列。从表 4 可以看出，H-MR-K-NN 算法在精确度上虽然低于 MR-K-NN，但是两者相差并不大。然而，H-MR-K-NN 在运行时间上却远远优于传统的 MR-K-NN 算法，原因是 H-MR-K-NN 算法通过充分利用 SimHash 大幅减小了 K-近邻的搜索空间。实验结果表明，本文提出的算法是很有效的。

表 4　MR-K-NN 与 H-MR-K-NN 在精确度上的比较/%

数据集	MR-K-NN	H-MR-K-NN
Forest CoverType	99.7	94.7
Skin Segmentation	91.2	87.2
Poker Hand	72.8	68.4
Statlog	84.4	79.6

表 5　MR-K-NN 与 H-MR-K-NN 运行时间的比较/s

	Forest CoverType	Skin Segmentation	Poker Hand	Statlog
第一次	16:3	2095:48	440:154	67:12
第二次	17:4	2143:54	465:158	62:10
第三次	16:3	2111:72	464:161	65:11
第四次	16:3	2224:51	474:154	63:12
第五次	16:3	2081:50	475:159	62:11

为了说明 H-MR-K-NN 算法在运行时间上的巨大优势，表 6 列出了两种算法的平均运行时间。

表 6　MR-K-NN 与 H-MR-K-NN 平均运行时间的比较/s

数据集	MR-K-NN	H-MR-K-NN
Forest CoverType	16	3
Skin Segmentation	2131	55
Poker Hand	463	157
Statlog	63.8	11.2

结束语 针对 K-NN 算法在大数据环境下的不足，提出了一种基于哈希技术和 MapReduce 的大数据集 K-近邻算法。与同类算法相比，所提算法具有如下两个特点：1)算法思想简单，易于编程实现；2)算法运行效率高，在保持分类能力的前提下，所消耗的运行时间远远少于 MR-K-NN 所消耗的运行时间。未来的工作包括：1)在更多、更大的数据集上进行实验，并对实验结果进行统计分析；2)对 SimHash 算法做进一步改进，以提高其计算精度。

参 考 文 献

[1] WU X D, KUMAR V, QUINLAN J R, et al. Top 10 algorithms in data mining [J]. Knowledge & Information Systems, 2007, 14 (1): 1-37.

[2] COVER T, HART P. Nearest neighbor pattern classification[J]. IEEE Transactions on Information Theory, 1967, 13(1): 21-27.

[3] ZHANG X G. Pattern Recognition(Third Edition)[M]. Beijing: Tsinghua University Press, 2010. (in Chinese)
张学工. 模式识别(第 3 版) [M]. 北京：清华大学出版社，2010.

[4] 王晓东. 计算机算法设计与分析(第 4 版)[M]. 北京：电子工业出版社，2012.

[5] ALSUWAIYEL M H. Algorithms Design Techniques and Analysis(English Edition, Second Edition)[M]. Beijing: Publishing House of Electronics Industry, 2013. (in Chinese)
ALSUWAIYEL M H. 算法设计与分析(英文版，第 2 版)[M]. 北京：电子工业出版社，2013.

[6] CORMEN T H, LEISERON C E, RIVEST R L, et al. Introduction to Algorithms(English Edition, Second Edition)[M]. Beijing: Higher Education Press, 2001. (in Chinese)
CORMEN T H, LEISERON C E, RIVEST R L, 等. 算法导论(英文版，第 2 版)[M]. 北京：高等教育出版社，2001.

[7] HART P E. The condensed nearest neighbor rule[J]. IEEE Transaction on Information Theory, 1968, 14(5): 515-516.

[8] WILSON D R, MARTINEZ T R. Reduction techniques for instance-based learning algorithms[J]. Machine Learning, 2000, 38(3): 257-286.

[9] BRIGHTON C, MELLISH C. Advances in instance selection for instance-based learning algorithm[J]. Data Mining and Knowledge Discovery, 2002, 6(2): 153-172.

[10] OLVERA-LÓPEZ J A, CARRASCO-OCHOA J A, MARTÍNEZ-TRINIDAD J F, et al. A review of instance selection methods[J]. Artificial Intelligence Review, 2010, 34(2): 133-143.

[11] BENTLEY J L. Multidimensional Binary Search Trees Used for Associative Searching[J]. Communications of the Acm, 1975, 18 (9): 509-517.

[12] OMOHUNDRO S M. Efficient algorithms with neural network behavior[J]. Journal of Complex Systems, 1987, 1(2): 273-347.

[13] UHLMANN J K. Satisfying general proximity / similarity queries with metric trees[J]. Information Processing Letters, 1991, 40(4): 175-179.

[14] KNUTH D. Art of Computer Programming, Volume 3: Sorting and searching[M]. New Jersey: Addison-Wesley Professional, 1997.

[15] GIONIS A, INDYK P, MOTWANI R. Similarity search in high dimensions via hashing [C]// Proc. of 25th International Conference on Very Large Data Bases. 1999: 518-529.

[16] SALAKHUTDINOV R, HINTON G. Semantic hashing [J]. International Journal of Approximate Reasoning, 2009, 50 (7): 969-978.

[17] JUN W, SANJIV K, SHIH F C. Semi-supervised hashing for large-scale search [J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2012, 34(12): 2393-2406.

[18] LI W J, ZHOU Z H. Learning to hash for big data: Current status and future trends [J]. Chinese Science Bulletin, 2015, 60(5/6): 485-490. (in Chinese)
李武军，周志华. 大数据哈希学习：现状与趋势[J]. 科学通报，2015, 60(5/6): 485-490.

[19] WANG J, LIU W, KUMAR S, et al. Learning to Hash for Indexing Big Data-A Survey [J]. Proceedings of the IEEE, 2016, 104 (1): 34-57.

[20] MANKU G S, JAIN A, SARMA A D. Detecting near-duplicates for web crawling [C]// International Conference on World Wide Web. ACM, 2007: 141-150.

[21] DEAN J, GHEMAWAT S. MapReduce: Simplified data processing on large clusters [J]. Communications of the ACM, 2008, 51(1): 107-113.

[22] 黄宜华，苗凯翔. 深入理解大数据-大数据处理与编程实践[M]. 北京：机械工业出版社，2014.

(上接第 184 页)

[13] CHEN W T, ZHANG X M, LI Z J. Analysis of Topic Models on Modeling MicroBlog User Interestingness[J]. Computer Science, 2013, 40(4): 127-130, 135. (in Chinese)
陈文涛，张小明，李舟军. 构建微博用户兴趣模型的主题模型的分析[J]. 计算机科学，2013, 40(4): 127-130, 135.

[14] DI L, DU Y P. Application of LDA Model in Microblog User Recommendation[J]. Computer Engineering, 2014, 40(5): 1-6, 11. (in Chinese)
邸亮，杜永萍. LDA 模型在微博用户推荐中的应用[J]. 计算机工程，2014, 40(5): 1-6, 11.

[15] BLEI D M, NG A Y, JORDAN M I. Latent Dirichlet Allocation [J]. Journal of Machine Learning Research, 2003, 3: 993-1022.

[16] FOX C, PARKER A. Convergence in variance of Chebyshev accelerated Gibbs samplers[J]. Siam Journal on Scientific Computing, 2014, 36(1): A124-A147.

[17] ROBERTSON S E, WALKER S, JONES S, et al. Okapi at TREC-3[C] // Proceedings of the Third Text Retrieval Conference(TRCE 1994). Gaithersburg, USA, 1994.

[18] SHAO K, ZHANG J W. Research on personalized recommendation of Web text mining based on BM25F model[J]. Information Studies: Theory & Application, 2013, 36(11): 118-122. (in Chinese)
邵康，张建伟. 基于 BM25F 模型的 Web 文本挖掘个性化推荐研究[J]. 情报理论与实践，2013, 36(11): 118-122.