基于 Hadoop 的分布式朴素贝叶斯文本分类①

卫 洁，石洪波，冀素琴

(山西财经大学 信息管理学院，太原 030006)

摘 要：云计算的诞生，有效地解决了海量数据集的存储和分析处理。在云计算实现的开源 Hadoop 分布式系统集群上，使用 MapReduce 并行编程模型，设计并实现了一种对 TFIDF 改进的分布式朴素贝叶斯文本分类算法。实验结果表明，基于 Hadoop 框架的分布式朴素贝叶斯文本自动分类器不仅能处理节点失效，同时具有高效性和易扩展性的优势。

关键词：Hadoop；朴素贝叶斯；MapReduce；文本分类

Distributed Naive Bayes Text Classification Using Hadoop

WEI Jie, SHI Hong-Bo, JI Su-Qin

(Faculty of Information Management, Shanxi University of Finance & Economics, Taiyuan 030006, China)

Abstract: The emergence of the cloud computing has resolved the difficult of storing the abundant data and analysing data processing effectively. Based on the Hadoop open-source implementation, the cloud computing clusters distributable systems. Meanwhile, the usage of MapReduce parallel programming model has implemented a modified distribution on TFIDF Naive Bayes text classification algorithm. The experimental results show that improved TFIDF has chosen this unique method. The Distributed Hadoop framework has based on Bayes text which classifies automatically. This new achievement can not only handle the failure of nodes, but also possess high reliability and much more scalable advantages.

Key words: Hadoop; naive bayes; MapReduce; text classification

随着 Inernet 数据规模的急剧增加、应用类型的巨大丰富，海量数据的存储和分析处理给传统的系统框架带来重大的挑战，正所谓“算法再好，通常也难敌更多的数据”，在这种巨大潮流和趋势的推动下，云计算应运而生。Hadoop[1,3]作为云计算的实现解决了 TB 级以上数据集的存储、分析和学习问题，它是 Apache 组织的一个开源的分布式系统架构，广泛的运行于多种平台，有着低成本、可扩展性、可伸缩性、高效性、高容错性等优点，追求“Moving computation is cheaper than moving data”。它的核心组件包括分布式文件系统(HDFS)和 MapReduce 编程模型。

目前，数据挖掘中对大规模文本数据的自动分类引起了广泛关注。流行的文本分类算法有支持向量机(SVM)、K 最近邻(KNN)、神经网络和朴素贝叶斯(Bayes)等。其中朴素贝叶斯文本分类器以简单、高效成为焦点，它基于朴素贝叶斯的概率密度函数，描述分类系统中条件属性和分类属性之间的映射关系，较其他分类算法时间复杂度低、效率高。而朴素贝叶斯文本分类中属性条件概率的计算是关键，直接影响着分类器的性能，一般常用的计算方法是 TFIDF 算法，近年来，诸多学者对 TFIDF 算法做了不同改进，优化了朴素贝叶斯分类器的分类效果。本文结合云计算的优势和分布式的特点选择了一种 TFIDF 算法的改进方法，基于 Hadoop 分布式系统平台，运用 MapReduce[2] 并行编程模型，设计了基于 Hadoop 的分布式朴素贝叶斯文本分类算法，并用实验进行验证。

① 基金项目:国家自然科学基金(60873100);山西财经大学科研资助项目
收稿时间:2011-05-27;收到修改稿时间:2011-07-09

1 朴素贝叶斯文本分类

1.1 朴素贝叶斯文本分类模型

设训练样本数据集分为 m 类，记为 C={C_1,C_2,...,C_m}，以 P(C_j)表示事件 C_j 类发生的先验概率，j=1,2,...,m，且 P(Cj)>0，对于任一新文档 di=(w_1,w_2,..,$w_{|V|}$)，i=1,2,...,l，其属于 C_j 类的条件概率是 P(di|Cj)，其中 wk 为特征词，k=1,2,...,|V|，V 是特征词的集合，|V|为特征词的总个数。则贝叶斯公式计算的 C_j 类的后验概率为：

$$P(C_j \mid d_i)=\frac{P(C_j)P(d_i \mid C_j)}{\sum_{k=1}^{|V|} P(C_j)P(w_k \mid C_j)}$$

朴素贝叶斯文本分类是以贝叶斯公式为理论基础的一种文本分类学习方法，是通过某对象的先验概率和条件概率计算出其后验概率，选择具有最大后验概率值的类作为该对象所属的类。常用的模型有多变量贝努利模型和多项式模型两种。在多变量贝努利模型中，文本向量的形式是二进制向量，文档中出现的每个单词的概率与该文档其他单词出现的次数是独立的，而多项式模型考虑了文档中单词出现的频率信息，向量的表现形式为词频向量，两种模型的不同体现在 P(d_i|C_j)估计的方法。本文采用多项式模型。

多项式模型[4]将每篇文档看作一系列有序排列的词的集合，假定文档的长度对于给定类是独立的，且文档中的每个词与它在文中的位置及上下文关系也是独立的。设 N_{ik} 是 d_i 中 w_k 出现的次数，那么测试文档 d_i 属于 C_j 类的概率为：

$$P(d_i \mid C_j;\theta)=P(|d_i|)|d_i|!\prod_{k=1}^{|V|}\frac{P(w_k \mid C_j;\theta)^{N_{ik}}}{N_{ik}!}$$

其中参数每类中每个特征词出现的概率 $P(w_k \mid C_j;\theta)=\hat{\theta}_{w_k|C_j}(0\le\hat{\theta}_{w_k|C_j}\le 1,\sum_k\hat{\theta}_{w_k|C_j}=1)$，为了避免 $P(w_k \mid C_j;\theta)=0$，对其估计采用了 Laplace 平滑技术：

$$\hat{\theta}_{w_k|C_j}=P(w_k \mid C_j;\hat{\theta}_j)=\frac{1+\sum_{i=1}^{|D|}N_{ik}P(C_j \mid d_i)}{|V|+\sum_{k=1}^{|V|}\sum_{i=1}^{|D|}N_{ik}P(C_j \mid d_i)}$$

式中 D 是带有类标签的训练集，|D|是训练集中文档的数目，$\sum_{i=1}^{|D|}N_{ik}P(C_j \mid d_i)$ 是 C_j 类中 d_i 文档中特征词 w_k 出现的概率，一般采用 TFIDF 算法进行计算，所得结果为特征词 w_k 的 TFIDF 值。

1.2 TFIDF 及其改进

Salton[7]于 1973 年提出了 TFIDF 算法，经过多次验证，标准的 TFIDF 公式为：

$$TFIDF=tf_{ik}\times idf_{ik}=tf_{ik}\times\log(N/n)，$$

其中 tf_{ik} 是特征词 w_k 在文档 d_i 中出现的次数；idf_k 是出现特征词 w_k 的文档的倒数，N 为总文档数，n 为出现特征词 w_k 的文档数。主要思想是：TF 指某词在特定文档中出现的频率越高，说明它在区分该文档内容属性方面的能力就越强；IDF 指某词在文档中出现的范围越广，说明它区分文档内容的属性就越低。

很多学者对标准 TFIDF 算法进行分析，基于不同思想做了各种改进，并通过实验验证了改进的 TFIDF 算法的有效性，具有代表性的有以下三种：

1) 张玉芳、彭时名[8]等考虑很多情况下训练集中的数据都存在类间、类内分布偏差，而经典的 TFIDF 是将文档集作为整体来考虑的，其中 IDF 的计算并没有考虑到属性在类间、类内的分布情况。如果某属性在某类中大量出现，而在其他类别中出现很少，此属性的分类能力显然很强，但根据 IDF 的定义：若某属性在较多文档中出现，IDF 值则变小，否则变大。将 IDF 的计算方法进行如下改进，设某一类中出现 w_k 的文档数为 t，除此类外，包含 w_k 的文档数为 h。则 IDF 计算公式为：

$$IDF=\log(t\times N/n)=\log[t\times N/(t+h)]。$$

2) Bong Chin How 和 Narayanan K[9]基于数据集偏斜思想，即数据集关于类别的分布往往是偏斜的，提出了 Category Term DescriptorCTD 改进 TFIDF，公式为：

$$CTD(w_k,C_j)=TF(w_k,C_j)\times IDF(w_k,C_j)\times ICF(w_k)$$

$$=TF(w_k,C_j)\times\log\frac{|D(C_j)|}{DF(w_k,C_j)}\times\log\frac{|C|}{CF(w_k)}$$

其中 D（C_j）指类 C_j 中的文档数，DF(w_k,C_j)指类 C_j 中出现 wk 的文档数，CF（wk,Cj）指出现 wk 的类别数。

3) RobertoBasils[10]提出了 TF×IWF×IWF 公式：$tf_{ik}\times(\log(Z_k/Z))^2$，公式中 Z_k 为 w_k 在总文档中出现的次数，Z 是总文档中所有特征词出现的次数之和。

本文结合 Hadoop 分布式平台和 MapReduce 并行编程框架的特点，选择了第一种对 TFIDF 改进的方法称为 ATFIDF，进行了设计与实现。

2 基于Hadoop分布式朴素贝叶斯文本分类设计与实现

在 Hadoop 平台上利用五个 MapReduce 实现分布式朴素贝叶斯文本自动分类器的训练和测试两部分，其中训练部分采用四个 MapReduce 对训练集进行学习得出分类模型，测试部分根据模型用一个 MapReduce 对测试文档进行分类，并分析分类结果，算法设计过程中充分利用了 MapReduce 的并行优势、HDFS 的分布存储数据集特点。

2.1 分类器训练

分布式朴素贝叶斯的训练采用四个 MapReduce 作业类得出模型，其中前一个 MapReduce 的输出作为后一个 MapReduce 的输入，训练完毕后，贝叶斯分类模型建立并保存在分布式系统上，分类模型共三个文件：trainer-tfidf、trainer-weights、trainer-thetaNormalizer。

第一个 MapReduce 统计训练集中的总文档数、训练集中出现特征词 w_k 的文档数，计算每类中出现特征词 wk 的文档数和每类中每个特征词的 TF 值，其中计算 TF 值所采用的公式为：$TF = \sum \log[(1.0 + r_{ki})/(\sum r_{ki}^2)^{1/2}]$，式中 r_{ki} 是特征词 w_k 在文档 d_i 中出现的次数。reducer 对 mapper 的中间内容进行合并，结果以<key value>键值对的形式输出四个文件，每个文件及相关内容的表述见表 1，表中的 label 指类标签即 C_j，token 是特征词即 w_k。

表 1 第一个 MapReduce 的输出情况

filename	key	value
wordFreq	_WT,label,token	类 C_j 中特征词 w_k 的 TF 值
termDocCount	_DF,label,token	类 C_j 中出现 w_k 的文档数
featureCount	_FC,token	训练集中出现 w_k 的文档总数
docCount	_LC	训练集中文档的总数

第二个 MapReduce 根据表 1 的输出文件计算每类中每个特征词的 TFIDF 值，mapper 中采用公式：IDF=log(t×N/n)进行 IDF 的统计，reducer 依据 TF×IDF 计算 TFIDF 值，输出两个文件，其键值对及相关内容的表述见表 2。计算完毕后自动删除第一个 MapReduce 得出的 wordFreq、termDocCount、featureCount 三个文件。

表 2 第二个 MapReduce 的输出情况

	output filename	key	value
1	trainer-tfIdf	_WT,label,token	TFIDF 值
2	trainer-vocabCount	_FS	特征词总数

第三个 MapReduce 仅调用了第二个 MapReduce 输出结果中的 trainer-tfIdf，统计同一类标签 C_j 中所有特征词的 TFIDF 值，不分类别的同一特征词的 TFIDF 值之和，所有特征词的 TFIDF 总和，得到的输出结果有三个文件，文件内容中的键值对及相关内容的表述见表 3。

表 3 第三个 MapReduce 的输出情况

	output filename	key	value
1	Sigma_j	_SJ,token	每个特征词的 TFIDF 总和
2	Sigma_k	_SK,label	每类中所有特征词的 TFIDF 总和
3	Sigma_kSigma_j	_SJSK	所有特征词的 TFIDF 值总和

第四个 MapReduce 按照公式：$\sum \log[(TFIDF + 1.0)/(sigma\_k + VocabCount))]$ 对表 2 中的两个文件进行计算，并输出结果。

2.2 分类器测试

分类器测试时使用一个 MapReduce，mapper 返回的值是测试文档属于 C_j 类，即所有 w_k 在 C_j 类下的 $\sum frequency \times \log[(TFIDF + 1.0)/(Sigma\_k + VocabCount))]$ 值与在其他类下的值进行比较，取出最大值所属的类标签值，文档就属于此类。frequency 是测试文档中特征词 w_k 出现的次数。reducer 的工作是对 mapper 的结果做了合并，得出测试文档各类中正确分类的文档数。

3 实验结果与分析

实验搭建的环境由 sc706-26、27、28、29 共 4 台普通 PC 构成，操作系统是 Linux 的 fedora12，jdk 的版本为 jdk-6u19-linux-i586，hadoop 采用的版本是 hadoop-0.20.2，其中 sc706-26 作为主节点，启动 NameNode 和 JobTracker 进程，其余三台作为 DataNode 和 TaskTracker，为从节点。

实验采用的数据是文本分类算法测试中最常用的 20 个新闻组数据集，即来源于 UCI KDD Archive 的 20 Newsgroups 数据集，网址为：http://kdd.ics.uci.edu/。数据集中包括 20 个新闻组，本实验训练时使用数据集中所有类的文档，测试时随机抽取了六类新闻数据组，分别为：rec.motorcycles、talk.politics.misc、sci.electr- onics、talk.religion.misc、rec.sport.baseball、comp.sys.mac.hardware，为了方便以下表 4 中的表述，依次简记为 motorcycle、politic、electronic、religion、baseball、

hardware。数据集通过 FileFormatter 将文档合并即每类下的所有文档转换成一个文件，文件的内容经过分词，词之间加入空格，所有文件再加上类标签，预处理成 MapReduce 所需的按行处理的训练集。训练集的格式是类标签作为文件名的一类一个文件，一行一篇文档的<key value>键值对的文档文件。实验分类测试结果与传统算法结果对比见表 4，其中分类器正确分类的文档数与测试集文档总数的百分比是正确率，本实验的正确率为 99.56%，比传统算法提高了 0.96%，且分类器的召回率也得以改善。同时，实验也体现了基于分布式设计实现的朴素贝叶斯文本分类器的高数据容量、高效性、可靠性和易扩展性。

表 4 分类结果对比

类标签	测试文档数（篇）	正确分类文档数（篇）	
		TFIDF	ATFIDF
motorcycle	202	200	201
politics	187	185	187
electronics	178	175	177
religion	187	187	187
baseball	202	200	201
hardware	189	183	187

在高效性测试中，由于本实验训练集文档数目的限制，采用不同节点数处理不同数据量的时间对比图仅对比了 2 节点和 4 节点。数据的文档都是由 20 类新闻组数据构成的，数据处理成一类一个文件，一行一篇文档的<key value>键值对格式的文档文件。在图 1 中，当训练文档总数是 9600 万篇时，在 2 个节点和 4 个节点的训练分类模型耗时是 378 秒和 597 秒；当训练集为 36600 万时，2 节点和 4 节点运行作业所用时间分别是 970 秒、1664 秒。从图 1 整体可以看出，同大小数据集的训练时间随节点的增加减少，同时数据集越大，节点的数目对实验时间的影响越大，很好的体现了分布式朴素贝叶斯分类器的高效性。

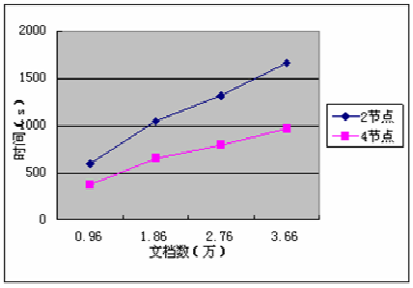

图 1 不同节点实验结果时间对比图

高可靠性测试时，设置 dfs.replication=3，即实验数据上传到分布式文件系统时备份数为 3。在程序运行时，人为的关掉任意一个从节点，造成节点故障，作业仍然成功，分类器的训练和测试结果与无故障时相同，不受影响，证明了分布式分类器的稳定可靠。

在易扩展性测试中，首先配置好新节点的系统和 Java 环境，然后将任一从节点的 Hadoop 集群配置打包发送到新节点后，只需在主节点的从节点列表中添加新的从节点地址，重新格式化 NameNode，重启 Hadoop 集群，新节点成功运行在集群中，分类器很容易得到了扩展。

4 结语

本文通过分析多项式模型的朴素贝叶斯文本分类中 TFIDF 的改进方法，采用 MapReduce 并行编程框架，基于 Hadoop 开源平台设计并实现了一种对 TFIDF 算法改进的分布式朴素贝叶斯文本自动分类器。实验结果证明了 MapReduce 编程模型应用于大规模数据集文本分类可以明显提高分类器的效率，同时 Hadoop 分布式框架使分类器具有高可靠性和易扩展性的特点。

参考文献

1 Hadoop WT. The definitive guide.O'Reilly Media,Inc, 2009.
2 Taiwan Hadoop Forum.http://forum.hadoop.tw/2009.
3 Apache Hadoop.(2009-09-12).http://hadoop.apache.org/.
4 McCallum A, Nigam K. A Comparison of Event Models for Naive Bayes Text Classification. AAAI/ ICML–98 Workshop on Learning for Text Categorization 1998:41–48.
5 Dean J, Ghemawat S. MapReduce: Simplifed Data Processing on Large Clusters. Proc.of the 6th Symposium on Operating System Design and Implementation, San Francisco, 2004.
6 Cutting D. Scalable Computing with MapReduce. Proc.of O'Reilly Open Source Convention, Poland. 2005.
7 Salton G, Clement TY. On the construction of effective vocabularies for information retrieval. Proc. of the 1973 Meeting on Programming Languages and Information Retrieval, New York ACM, 1973:11.
8 张玉芳,彭时名,吕佳.基于文本分类 TFIDF 方法的改进与应用.计算机工程,2006,32(19).
9 How BC, Narayanan K. An empirical study of feature selection for text categorization based on term weightage. Proc. of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence Washington DC: IEEE Computer Society, 2004:599–602.
10 Basils R, Moschiiti A, Pazienza M. A test classifier based on linguistic processing. Proc. of IJCAIp, Machine Learning for Information Filtering,1999.