基于 N-Gram 模型的高速汉字编码识别系统

李继锋　刘　群
（中国科学院计算技术研究所软件研究室，北京 100080）
E-mail：jifengli@software.ict.ac.cn

摘　要　该文提出了一个应用 n 元语法模型（N-Gram）自动识别文档中汉字编码的方法，并介绍了一个已投入使用的汉字编码自动识别系统的设计和具体实现。该系统采用的是以字为基本单位的一元语法模型 Uni-Gram，建立在语料库的基础上，仅用输入前 N 个字的字频计算输入串的生成概率，可以高速、准确识别。

关键词　编码识别　n 元语法模型　一元语法模型

文章编号 1002-8331-(2004)03-0039-03　**文献标识码** A　**中图分类号** TP391.4

N-Gram Based High Speed Chinese Encoding Recognizing System

Li Jifeng　Liu Qun
(Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080)

Abstract: This paper describes a new algorithm based on N-Gram which can automatically recognize Chinese internal encoding.The design and implementing of a completed system based on the algorithm are introduced.The algorithm uses Uni-Gram whose basic cell is Chinese character.The system can recognize quickly and accurately only using the first N Chinese characters.

Keywords: Codeset Recognizing, N-Gram, Uni-Gram

因为历史和地域原因，汉字有不少编码标准，其中最常见的是 GB2312 和 BIG5，在 Unicode 被完全接受前，他们将共存相当长的一段时间，而相当多的任务，比如实时信息处理等，需要准确地判断当前文档（或信息流）中的汉字编码，随着海峡两岸及世界华人文化交流的日益频繁，自动汉字编码识别已经变得十分重要。但由于商业等各方面的因素，公开的算法并不多，该文以已完成并投入使用的自动识别系统为例，介绍了一种使用 N-Gram 模型的算法。

该文第一部分首先介绍现在已知的一些码制识别算法；第二部分介绍 N-Gram 模型以及在码制识别中的应用；第三部分讲述系统参数选择及调整策略；第四部分描述系统结构；第五部分解释系统处理流程，包括知识库训练、参数调整、识别过程等。最后对算法的性能进行了分析并给出了系统的测试结果。

1　常用码制识别算法

1.1　汉字编码统计法[1]

这是一个在 Internet 上公开的算法，主要用来识别 GB2312 和 Big5 文件的编码。

GB2312 和 Big5 的编码范围虽然有重复，但常用字还是有一定差别，GB2312 编码的码区比 Big5 高，所以整体算下来均值会有所差别。该算法正是利用了这一点：取文件中所有字编码的平均值（将字节内容转化为无符号整数计算），再利用该平均值和训练得到的阈值比较，判断文档编码是 GB2312 还是 Big5（大于阈值为 GB2312，小于阈值为 Big5）。

该方法的作者还尝试了下面三种具体的方法，即只累加汉字的第一个字节；只累加汉字的第二个字节；累加两个汉字字节。通过实验[1]给出三种统计方法得到的阈值都是 184，但效果最好的是第二种方法。

这个方法的优点在于：只需统计文件本身信息，使用简单。但也有明显的缺点，比如只适用于字数较多的文件（根据算法设计者的统计，至少要 80 字左右），字数较少的文件不够准确；汉字和 ASCII 等共存的文件会对统计结果产生很大的影响；利用了 GB2312 和 Big5 两种具体编码方式的码区特点，不具有普遍性，难以推广到普遍意义上的汉字编码识别。

1.2　利用各编码方式非重叠区识别的方法[2]

虽然各种编码方式有交叉重叠的区域，但一般不是完全相同的，有人就利用各编码方式的非重叠部分进行识别，如果文档中出现了只存在于一种编码空间的字符，则可以排除其他编码方式的可能性。显然，这种方法限制性比较大，不能保证能区分，所以作者还以字符串语义有效性识别予以辅助。

1.3　常用词组识别法

两岸汉字编码不同，常用语也不同，那么常用词组也不同。因此有人提出如果从常用词组分析差别可能会更大，识别率也就更高。由于分析词组涉及到词法甚至句法分析，开销较大，目前还没有看到有这种思路的实现，但这应该是一个比较好的思路。

2　N-Gram 及其在汉字编码识别中的应用

2.1　N-Gram 模型是 N 阶 Markov 过程

一列随机变量 $S_1, S_2, \ldots, S_m$ 中，如果其中任何一个随机变量 S_i 发生的概率只与其前面的 n 个变量 $S_{i-1}, S_{i-2}, \ldots, S_{i-n}$ 有关，即：

基金项目：国家 863 高技术研究发展计划资助（编号：2002AA142110）
作者简介：李继锋，男，硕士研究生，当前研究领域为自然语言处理和机器翻译。刘群，男，副研究员，当前研究领域为机器翻译。

$P(S_i|S_{i-n}S_{i-n+1}\cdots S_{i-2}S_{i-1})=P(S_i|S_1S_2\cdots S_{i-2}S_{i-1})$

则称其为 n 阶 Markov 过程。N-Gram 模型是把所有的连续可重叠的 n 个词作为一个单元，并假设其为一个 N 阶 Markov 过程。对于一个由 m 个词 $w_1w_2w_3\cdots w_{m-1}w_m$ 组成的语句 S，定义：

$P(s)=P(w_1)P(w_2|w_1)P(w_3|w_1w_2)\cdots P(w_m|w_1w_2w_3\cdots w_m)$

作为一个语句的出现概率。由前面的假设可得：

$$P(s)=\prod_{i=1}^{m}P(w_i|w_1w_2\cdots w_{i-1})$$

$$\approx P(w_1w_2\cdots w_{N-1})\prod_{i=N}^{m}P(w_i|w_{i-N+1}w_{i-N+2}\cdots w_{i-1})$$

式中的 $P(w_i|w_{i-N+1}w_{i-N+2}\cdots w_{i-1})$ 是训练集合中 N 词串 $w_{i-N+1}w_{i-N+2}\cdots w_{i-1}w_i$ 在所有以 $w_{i-N+1}w_{i-N+2}\cdots w_{i-1}$ 开头的词串中所占的比率。

当 N=1 时，$P(S)=\prod_{i=1}^{m}P(w_i)$，这个就是要用的 Uni-Gram 概率模型的公式。

2.2 N-Gram 参数训练

该文的算法只用到了一元语法模型(Uni-Gram)，这里只给出 Uni-Gram 的训练方法：

```
For each word w in the Training-Set
  If P(w)>threshold
  Then add w to 1-Gram-Set
```

更具普遍意义的 N-Gram 参数训练，请参考[3]。

2.3 Uni-Gram 在编码识别中的应用

对于输入的字节流 S，假设它是按照作者需要判断的编码方式编码的，就可以将其理解为一系列该编码方式下的字 w_i，考虑到使用二元语法模型 Bi-Gram 会造成极大的系统开销而且一元语法模型 Uni-Gram 的处理能力对这个问题已经足够，所以以这些字为基本单位，文中采用 Uni-Gram 即：

$$P(S|Code)=\prod_{i=1}^{m}P(w_i)$$

$P(w_i)$ 是按照上面 1-Gram-Set 的计算方法统计语料库中出现次数达到一定量的字频信息。

该文本实际对应的码制：

$$Code^*=\arg\max_{code}P(Code|S)=\arg\max_{code}\frac{P(S|Code)P(Code)}{P(S)}$$

$$=\arg\max_{code}P(S|Code)P(Code)$$

假设各种码制可能出现的概率相同，则：

$$Code^*\approx\arg\max_{code}P(S|Code)$$

所以，使得文本出现概率最大的码制就是要选择的码制。

3 系统参数选择及调整策略

3.1 各码制字频表

字频表是最核心的数据结构，各码制的字频表均按照上面 1-Gram-Set 的计算方法得到。各种编码方式的语料库非常关键，只要有某种码制的语料库，算法就能对该码制的识别提供支持，具有很好的可扩展性。为了论述的方便，该文仅以 GB2312 和 Big5 为例。

这里使用的 GB2312 语料库共 4000 万左右汉字，Big5 语料库 6000 万字，而且覆盖面很广，保证了信息真实、可靠。

为了方便计算，字频表中存储的内容不是原始的频度信息，而是经过了计算的对数值(也可以乘以适当的系数)。

3.2 索引方式

在识别过程中，需要频繁查找字频表，字频表在内存中的存储、索引方式非常重要。作者根据各编码码区的不同特点，分别处理，以达到节省空间、提高索引速度的目的。

对 GB2312、Big5，都是采用一种没有冲突的 hash 表进行索引，搜索时间复杂度为 $O(1)$。

对 GB2312，采用内码直接定位，16~87 区为汉字，都是连续的编码空间，所以可以根据计算出的汉字的区位码直接定位(需要多出 5 个空位)，结构为：

```
#define TRAINEDGBCOUNT 6768 //GB 码的总汉字数(里面包含5个空位)
double m_fGBPercent[TRAINEDGBCOUNT];//国标的字频信息
```

Big5 的编码方式稍显复杂，高字节是连续的空间，A4~F9，可以作为二维数组的第一维；低字节是两个连续的空间，40~7E，A1~FE，可以作为二维数组的第二维，计算时，高字节直接以 A4 为基础，低字节按取值范围分别以 40、A1 为基础计算在二维数组中的下标。

```
#define B5HIBYTECOUNT 86//B5 码按照高字节值先索引 A4~F9
#define B5LOWBYTECOUNT 157 //B5 码按照低字节值索引 40~7E,A1~FE
double m_fB5Percent[B5HIBYTECOUNT][B5LOWBYTECOUNT];//大五码的字频信息
```

3.3 有效识别字数

一般的码制识别方法都是全文处理的，这样可以保证处理的准确率，但系统实时性要求很高，通过实验，作者发现该方法可以在字数比较少的情况下就取得很好的准确率，所以，只处理输入的前 N 个汉字，对于前 N 个字不足以判断码制的，采用增量式扫描，直到做出判断或者扫描完毕，这样，在不损失准确率的前提下，极大提高了处理速度。

N 值的选择对系统效率有很大的影响，根据经验，取 N=10，实验表明，这样可以判断 90%输入串的码制，而最多需要的字数为 40。

3.4 各码制概率阈值

对各种编码方式下输入的生成概率的简单比较并不能保证识别的准确性，这样可能把人们处理能力之外的编码方式(比如日语、韩语等)识别成能处理的一种。为了提高准确率，对各种编码方式都训练出确认所需的最小概率，只有生成概率是各种编码方式中的最大值，且高于这个阈值，才能确认是这种编码方式。

4 系统结构

该系统的完整结构图如图 1。

5 系统处理流程

5.1 字频表训练

根据汉语的规律，一小部分的常用汉字覆盖了大部分的文本，900~1200 个汉字就可以得到很好的覆盖率，所以一种观点认为只做出常用字的字频表，根据常用字出现的频率计算就能取得很好的效果，但是由于“常用字”是个不太确切的概念，需要语言学家做进一步的研究，所以这里采用纯统计的方法，以大规模通用语料库(GB2312 语料：45,376,453 字；Big5 语料：61894734 字)为基础进行字频统计。最后得出的字频表包含

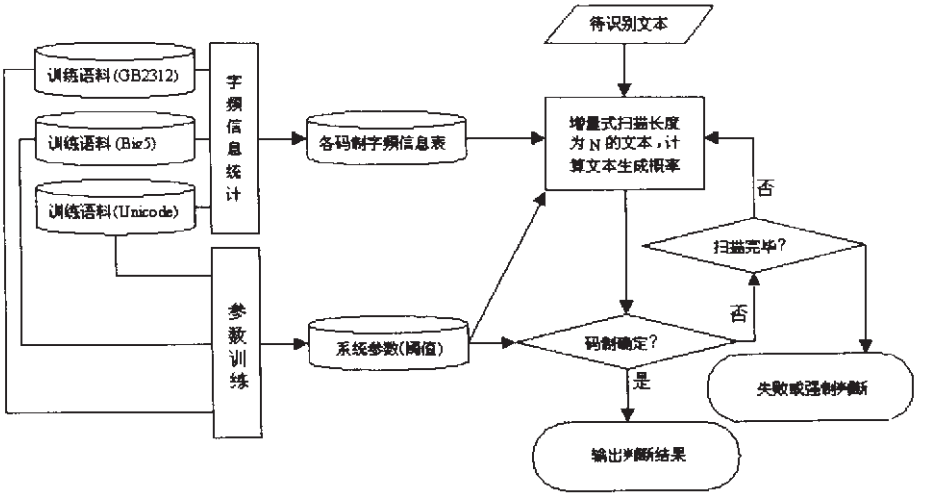

图 1　系统结构图

GB2312 汉字 6303 个,Big5 汉字 13046 个。

5.2　编码识别过程

编码识别的过程为:从头开始对输入串进行扫描,分别按照人们关心的所有编码方式寻找前 N 个“字”,查找每个字的字频信息,根据 Uni-Gram 的算法求得该子串的生成概率;最后根据这个概率值和阈值的比较结果以及各码制之间的比较结果做出判断。能判断出码制,结束;否则增量式继续扫描 N 个“字”...直到做出判断或者扫描结束。

为了提高效率,避免额外的内存开销,所有码制的处理是在一次扫描过程中完成的,并且不保留切分结果,只保留概率值的乘积。

考虑到汉字和 ASCII 字符共存的情况,对扫描过程中判断出的 ASCII 字符跳过不做处理,对处理串进行切分的过程中主要的问题就是常见的‘半个汉字’问题,比如高位为‘1’的 ASCII 字符的影响、含有半个汉字的非正常文本。考虑到这种干扰性字符不是很常见再加上系统对实时性要求比较高,所以采取了下面较为简单的策略:

Unicode 编码对所有字符都是双字节编码(UTF-8 这种可变长编码在该系统中有另外的模块处理),处理起来相对简单;GB2312 和 Big5 的共同特点是高字节的最高位是“1”,所以遇到高位为‘1’的就认为遇到了汉字,一次扫描两个字节,查找字频信息,如果不在自己的编码范围内,加权处理(出现了不在自己编码范围内的字符,很大程度上可能不是这种编码);一般高位为‘0’的字节认为是 ASCII 字符,直接跳过,不做计算。

下面以一个实例说明识别过程:

比如输入字节流为(十六进制):

“D3 C3 4E 2D 47 72 61 6D CA B6 B1 F0 B1 E0 C2 EB BC F2 B5 A5 D3 D0 D0 A7 A1 A3”。

按照 GB2312 理解,这句话是“用 N-Gram 识别编码简单有效。”

按照 Big5 理解,这句话是“ N-Gram 梗晤 等岫”。

显而易见,如果按照 GB2312 理解都是些常用字,按照 Big5 理解都是生僻字,计算的概率值肯定会差别很大。

$P(S \text{ as GB2312})=-3.1303533657277>>P(S \text{ as Big5})$

$=-9.5469436316574$

这个结果是作者方便计算,取对数(并乘了系数)后累加并取均值的结果。

而且 $P(S \text{ as GB2312})$超过了一个串确定是 GB2312 所需达到的最小概率(阈值)同时 $P(S \text{ as Big5})$没有达到确定是 Big5 的最小概率,所以笔者认定该串是用 GB2312 编码的。

6　实验结果

6.1　性能分析

该文从以下几个方面来分析系统的性能:

(1)识别准确率

N-Gram 是一种简单但很实用的模型,因为它是基于大规模语料库的,所以具有真实、可靠的特点,实验证明用在该文的问题上也有很好的效果,该系统目前已经投入正式运行,效果令人满意。

(2)运行效率(时间复杂度、空间复杂度)

该系统的开发有很强的实时性要求,所以对时间、空间上的效率有很高的要求,因为字频训练、参数调整没有实时性要求,所以运行效率分析主要针对识别模块:

空间:程序需要的内存空间主要是初始化模块导入的各个字频表,时间上的考虑,对最常用的 GB2312 和 Big5 的信息采用了上文提到的索引方式,占用内存 158K。

时间:

扫描时间:识别时对输入待识别串采用一次扫描的方式,没有回溯,时间复杂度为 $O(n)$,其中 n 为作者设定的有效识别所需汉字数的阈值(该系统中设置为 10)。

字频表查询时间:对 GB2312 和 Big5 采用的是内码索引,直接定位(没有冲突的 Hash),查询复杂度为 $O(1)$。

6.2　测试结果

这里设置有效识别汉字数 N=10(滤掉 ASCII 等字符后的汉字数),表 1 列出了作者在已知码制的语料上的测试结果:

表 1

语料来源	规模	已知编码	正确率	处理时间/核心处理时间(秒)	N=10 成功率	最大判断字数
Internet 采集	498M (29496files)	GB2312	100%	435.356000/1.603000	90.5%	40
Internet 采集	71.2M (4799files)	Big5	100%	36.392000/0.079	100%	10

(下转 177 页)

```
{/* 私有属性 */
 private ServletConfig m_cServletConfig=null;
 public static CPBO_ORBClient m_cORBClient=null;
 public static NamingContext m_cNameContext=null;
 public CPBD_DBFetchManager m_cDBFetchManager=null;
 public CPBE_EventConsumer m_cConsumer=null;
 public CPBE_EventSupplier m_cSupplier=null;
 ...
 /** 有关认证、版本、一些路经信息的属性略 */
 ...
 /**
 * 初始化。init 方法初始化静态属性 m_Orb,...,m_ExHandle,
 *@param p_cServletConfig ServletConfig 对象,
 * 获得在 servlets.properties 设置的 initparams 参数
 *@return void
 *@exception ServletException
 */
  public void init (ServletConfig p_cServletConfig)throws ServletException
 {
 }
 /** 继承来的 doGet 方法,由继承类实现
 *@param request HttpServletRequest 对象
 *@param response HttpServletResponse 对象
 *@return void
 *@exception IOException,ServletException
 */
  public void doGet (HttpServletRequest request,HttpServletResponse response)throws IOException,ServletException
 {
 }
 /** 继承来的 doPost 方法
 *@param request HttpServletRequest 对象
 *@param response HttpServletResponse 对象
 *@return void
 @exception IOException,ServletException*/
 public void doPost (HttpServletRequest request, HttpServletResponse response)throws IOException,ServletException
 {
 doGet(request,response);
 }
}
```

在具体开发的应用中其他类都可以继承以上基类，这样，操作变得极其方便。例如操作数据库,因为基类已初始化数据库管理对象 m_cDBFetchManage；人们只须调用 m_cDBFetchManager.createDBFetch()获得数据库对象,之后就可调用 dbaccess.idl 数据库接口中的操作来操作任意一数据库。

当前,EJB+Servlet+JSP 几乎成为电子商务的开发标准。但是如果将 SERVLET 和 CORBA 技术相结合将会如虎添翼,更加完美。(收稿日期:2003 年 6 月)

参考文献

1.T G Lewie.Where is client/server software headed?[J].Computer,1995;28(4):49~55

2.王怀民,史殿习,吴泉源.新一代分布式系统集成中间件[J].计算机学报,1997;20(增刊-并行与分布计算专辑):90~96

3.Robert Orfali,Dan Harkey,Jeri Edwards.The Essential Distributed Objects Survival Guide[M].John Wiley&Sons,INC,1996

4.Thomas J Mowbray,Raphael C Malveau.CORBA Design Patterns[M].John Wiley&Sons,INC,1997

5.R Schulte.Three-Tier Software Architecture in Perspective.SMS:K-401-1565,SMS Research Note,Gartner Group,1994-12

6.URL:http://www.omg.org/

(上接 41 页)

作者的测试平台是 P IV1.7G CPU,128M DDR 内存,Windows XP。

测试用语料是该系统的采集器在 Internet(web、telnet 各种渠道)上采集到的真实原始语料,具有普遍意义。虽然正确率达到了 100%，但也不表明该系统已经完美无缺了，一些特殊字符,比如,高位为 1 的 ASCII 字符(制表符等)、包含半个汉字的非正常文本就可能对识别造成影响,但实际应用中,这种字符连续出现的很少,基本上对识别没有影响,所以从实用的角度出发,这个效果是令人满意的。如何避开这些特殊字符的影响,也是系统下一步完善的方向。

实验表明,90.5%的文件在 10 个汉字左右就能判断出编码方式,最多只需要 40 个汉字就可以判断(Big5 文件需要的字数更少,是因为有更多的 Big5 的字不在 GB2312 的编码区内,能更快排除 GB2312 的干扰)。而且作者的字频表索引方式效率很高,这就使得系统处理的速度非常快,498M 的语料实际处理时间才 1.603 秒,这个速度可以满足实时性很强的系统的需要。

7 总结

该文介绍了一种利用一元语法模型(Uni-Gram)进行汉字内码识别的方法,并用一个已经开发完成并且运行效果良好的系统为例,介绍了处理流程、进行了性能分析并给出了测试结果。实验表明,Uni-Gram 可以高速、准确地处理该文的问题,可以满足实时处理等各种应用。

但是作者也认识到，单一的以字为基本单位的 Uni-Gram 方法在处理一些复杂情况时可能会有一些不足，结合其他方法，可能会有更好的效果，比如 GB2312 和 Big5 所对应的简体、繁体中文常用词汇有很大差别，以词为单位的 Uni-Gram 可能会对识别有所帮助;另外将字节流划分成汉字序列的方法显然过于简单,虽然实际效果不错,但是肯定还存在很大的提升空间,这些都是今后可以改进的方向。(收稿日期:2003 年 4 月)

参考文献

1.于明俭(中国科学院高能物理研究所计算中心).GB/BIG5 文件识别.http://ftp.cityu.edu.hk/pub/chinese/ifcss/data/chrecog.gb.html

2.尹宝生,潘峰,徐立军等.中日韩大字符集文字编码的比较研究.http://www.ge-soft.com/research/paper/he4.htm

3.Wang Xiaolong,Daniel Yeung,Guan Yi.An Algorithm for Constructing Higher Order N-grams of Chinese Words[C].In:International Conference on Machine Translation & Computer Language Information Processing

4.张健,李素建,刘群.N-gram 统计模型在机器翻译系统的应用[J].计算机工程与应用,2002;38(8):73~75

5.辛春生,孙玉芳.简繁汉字转换系统的设计与实现[J].软件学报,2001;11(11)