中 文 信 息 学 报
第 15 卷 第 5 期 JOURNAL OF CHINESE INFORMATION PROCESSING Vol. 15 No. 5

中文校对系统中纠错知识库的构造及纠错建议的产生算法

张仰森

(山西大学计算机科学系 太原 030006;
中国科学院自动化研究所模式识别国家重点实验室 北京 100080)

摘要:本文依据待校对文本中的常见错误类型介绍了纠错知识库的构造方法以及基于该纠错知识库的自动纠错算法。该算法通过利用出错字串的特征,结合上下文启发信息,可有效地对文本中的别字、漏字、多字、易位、多字替换等错误提供纠错建议。文中还对纠错建议的排序算法进行了讨论。

关键词:纠错知识库;纠错建议;纠错算法;似然匹配
中图分类号:TP391 A

The Structuring Method of Correcting Knowledge Sets and the Producing Algorithm of Correcting Suggestion in the Chinese Text Proofreading System

ZHANG Yang-sen[1,2]
(1. Computer science department, Shanxi University Taiyuan 030006;
2. National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences Beijing 100080)

Abstract: According to common error types in pre-proofreading text, this paper introduce the method to structure correcting knowledge sets and a automatic correcting algorithm based on this correcting knowledge sets. The algorithm makes a full use of the characteristics of wrong strings and context heuristic information. It can provide correcting suggestions for such errors as ghost word, missed Chinese characters, superfluous Chinese characters, reversed Chinese characters and substituted Chinese characters etc. The method of sorting the correcting suggestions is also discussed.

收稿日期:2000-07-12;**修改稿收到日期**:2000-12-07
基金项目:山西省自然科学基金(981031)
作者张仰森,男,1962 年生,教授,主要研究领域为人工智能和中文信息处理.

Keywords: correcting knowledge sets; correcting suggestion,; correcting algorithm; likelihood match

一、引言

中文自动校对系统中纠错建议是用来辅助用户改正所检查出的错误的，纠错建议的有效性是衡量自动纠错性能的主要指标，它有两点要求，一是提供的修改建议中应该含有正确或合理的建议；二是正确或合理的修改建议应尽可能排列在所有修改建议的前面。因此，纠错建议的产生算法及其排序算法是自动纠错研究的两个核心点。我们在[1]中已对待校对文本中的错误类型进行过分析，一般有以下几类：

①容易混淆的字词错误

如：宋庆龄于8月14日零晨1时抵达雅加达。（凌晨）

②由于编码相同或相近（包括拼音、五笔编码）引起的别字错误

如：我们这些化(hua)夏子孙。（华 hua 夏）；　　就难免必(ny)理不平衡。　（心 nt 理）

③由于编码相同或相近（包括拼音、五笔编码）引起的多字替换错误

如：对于大规模真实广西(yysg)的研究。（文本 yysg）

④漏字、多字、易位造成的错误

如：文稿中仍会遗留许多误。（错误）；　　维护建筑市场场秩序。（市场）

忽视发挥利率的杆杠作用。（杠杆）

⑤英文单词拼写错误

如：95年4月联想推出新版ofice办公软件。（office）

自动校对系统中有关纠错建议的产生方法，文献[4]采用模式匹配方法对长词进行纠错处理，但没有充分利用出错字串的特征，算法计算量大；文献[3]提出一种替换字表结合主词典，通过加字和换字对侦测出来的错误字串提供修改建议的纠错算法，但该算法的纠错建议局限于替换字表，没有考虑上下文启发信息，主要只考虑对错字这种错误类型进行纠错，对单字错只是简单地利用替换字表中的候选字进行替换，对漏字、多字、易位、多字替换、英文单词拼写等错误类型的纠错能力较弱。本文首先讨论了针对上述各种文本错误类型的纠错知识库的构造方法，然后给出了基于纠错知识库的纠错建议产生算法和纠错建议排序算法。

二、纠错思想及纠错知识库的构造

2.1　纠错思想

利用输入文本中的错误特征[1]结合似然匹配方法建立纠错知识库，然后基于纠错知识库研究纠错建议的产生算法，并根据字字同现信息排除不合理建议，对纠错建议进行排序，使合理的纠错建议尽量排在前面，增强纠错建议的可读性。

对易混淆的字词错误，纠错时查易混淆词词典，直接给出纠错建议；对输入编码相同或相近（包括拼音、五笔编码）的字词错误，用纠错候选字、词替换出错字词，给出纠错建议；对漏字、多字、易位错误，采用似然匹配法，得到纠错建议；对英文单词拼写错误，采用“骨架键法”纠正部分错误。

2.2　纠错知识库的构造

2.2.1　易混淆词词典的构造

收集常见的易混淆字词，建立易混淆词词典，在纠错时先查该词典，可直接得到部分易错字词的纠错建议，另外，可以动态地将人工给出的纠错建议加入该词典中。该词典的格式为：

错误字串　纠错建议

零晨　　凌晨

2.2.2　输入编码相同或相近(包括拼音和五笔编码)字词候选集的构造

1.拼音编码相同或相近的字词候选集的构造

拼音编码相同或相近的字取自 UCDOS 中文平台的拼音方案，其中共有编码 11894 组，取一个拼音对应多个字或词的 9922 组编码为拼音编码纠错建议候选集 Spy = Spyzi ∪ Spyci

(1)音同、音近候选字集 S_{pyzi} 6714 组：

如：音同"bei 北被倍备背辈贝杯卑臂悲碑"等；

(2)音同、音近候选词集 S_{pyci} 3208 组：

如："baohan 包含 饱含 包涵"。

2.五笔字型输入编码相近字词候选集的构造

我们设计了一个五笔字型编码相似函数，用于从 UCDOS 中文平台的五笔字型输入方案(其中共有编码 22482 组)中提取编码相同或相近的字词共 36837 组，构成五笔字型编码纠错建议候选集 $S_{wb} = S_{wbzi} \cup S_{wbci}$，其中候选字集 S_{wbzi} 5916 组，候选词集 S_{wbci} 30921 组。五笔字型编码相似函数定义如下：

设编码 $Code_i = A_{i1}A_{i2}\cdots A_{ik}(1 \leqslant k \leqslant 4)$ 和编码 $Code_j = A_{j1}A_{j2}\cdots A_{jp}(1 \leqslant p \leqslant 4)$。

定义 1：键位关系

① 同一键：如"Q"与"Q"

记为 $P(A_{im}, A_{jm}) = Same; (1 \leqslant m \leqslant 4)$

② 同一行且键位相邻：如"F"与"G"，"D"与"S"。

记为 $P(A_{im}, A_{jm}) = SRN; (1 \leqslant m \leqslant 4)$

③ 不同行且键位相邻：如"F"与"V"("B")，"J"与"l"。

记为 $P(A_{im}, A_{jm}) = \sim SRN; (1 \leqslant m \leqslant 4)$

④ 键位不相邻：如"M"与"P"。

记为 $P(A_{im}, A_{jm}) = \sim N; (1 \leqslant m \leqslant 4)$

定义 2：键间权重 $Q(A_{im}, A_{jm})$

$$Q(A_{im}, A_{jm}) = \begin{cases} 1.0 \text{ 当 } p(A_{im}, A_{jm}) = Same \\ 0.9 \text{ 当 } p(A_{im}, A_{jm}) = SRN \\ 0.7 \text{ 当 } p(A_{im}, A_{jm}) = \sim SRN \\ 0.6 \text{ 当 } p(A_{im}, A_{jm}) = \sim N \end{cases}$$

定义 3：五笔字型编码相似函数 $W(Code_i, Code_j)$

$$W(Code_i, Code_j) = \sum_{m=1}^{Min(k,p)} Q(A_{im}, A_{jm}) \quad (k,p \text{ 分别为 } Code_i, Code_j \text{ 的码长})$$

当相似函数 $W(Code_i, Code_j)$ 满足下列条件时，我们称 $Code_i$ 和 $Code_j$ 为相似码：

当 $Min(k,p) = 1$ 且 $Max(k,p) - Min(k,p) \leqslant 1$，$W(Code_i, Code_j) \geqslant 0.9$；

当 $Min(k,p) = 2$ 且 $Max(k,p) - Min(k,p) \leqslant 1$，$W(Code_i, Code_j) \geqslant 1.7$；

当 $Min(k,p) = 3$ 且 $Max(k,p) - Min(k,p) \leqslant 1$，$W(Code_i, Code_j) \geqslant 2.6$；

当 $Min(k,p) = 4$ 且 $Max(k,p) - Min(k,p) \leqslant 1$，$W(Code_i, Code_j) \geqslant 3.6$。

下面的几个编码相似的例子就是用上述相似函数推导出的：

①是(j)－上(h)　　(W＝0.9≥0.9)

②字(pb)－安(pv)　　(W＝1.9＞1.7)

③标本(sfsg)－根本(svsg)　　(W＝3.7＞3.6)

3.纠错候选集的排列格式

令纠错候选字集为 $S_{zi}=S_{wbzi}\cup S_{pyzi}$，候选词集为 $S_{ci}=S_{wbci}\cup S_{pyci}$，其内容排列方法为：

a.纠错候选字集 $S_{zi}=S_{pyzi}\cup S_{wbzi}$，按字的内码大小排序，每个字的候选字集占一行；字在候选字集中的行号以哈希函数 addr＝(Byte1－0xb0)＊94＋(Byte2－0xa1)来定位，其中 Byte1Byte2 为该字的内码。设 Z 为字，它及其候选字的排列格式为：

Z　$C_{Z1}\cdots C_{Zi}\cdots C_{Zj}\cdots C_{Zm}$　　这里 C_{Zi} 为第 i 个纠错候选字。

在整个候选字集中，平均候选字数为 38 个，最多为 97 个；候选字按字频排列。

b.纠错候选词集 $S_{ci}=S_{pyci}\cup S_{wbci}$，按词的首字内码大小排序，所有首字相同的词按平均字字同现频率大小排列在同一行；若某一词有多个候选词，则这些词之间用“;”隔开，其次序按平均字字同现频率大小排列。在整个候选词集中，平均纠错候选词个数为 2.7 个，最多为 49 个。

2.2.3　漏字、多字的纠错建议获取

漏字、多字错误情况比较复杂，没有一定的规律可循，不能通过简单的换字操作得到纠错建议，本文取错误字串的首字和尾字做为启发信息，采用似然匹配的方法，将与出错词“相似”的词作为纠错建议。

1.似然匹配算法

对字串 $Z_1Z_2\cdots Z_n$ 和词 $C_1C_2\cdots C_m$；令函数

$$Same(Z_i,C_i)=\begin{cases}1 & \text{当 } z_i=c_i;\\ 0 & \text{当 } z_i\neq c_i\end{cases}$$

若当 m≥n 时，有：

$\Sigma Same(Z_i,C_i)\geqslant n-1$　　$(1\leqslant i\leqslant n;Z_1=C_1)$

或 $\Sigma Same(Z_{n-i+1},C_{m-i+1})\geqslant n-1$　　$(1\leqslant i\leqslant n;Z_n=C_m)$

或 $\Sigma Same(Z_i,C_i)+\Sigma Same(Z_{n-i+1},C_{m-i+1})\geqslant n-1$　　$(1\leqslant i\leqslant n;Z_1=C_1;Z_n=C_m)$

当 m＜n 时，有：

$\Sigma Same(Z_i,C_i)+\Sigma Same(Z_{n-i+1},C_{m-i+1})=m$　　$(2\leqslant i\leqslant m-1;Z_1=C_1;Z_n=C_m)$

那么，称字串 $Z_1Z_2\cdots Z_n$ 与词 $C_1C_2\cdots C_m$ 似然匹配，记为 $Match(Z_1Z_2\cdots Z_n,C_1C_2\cdots C_m)$；若 $Match(Z_1Z_2\cdots Z_n,C_1C_2\cdots C_m)$，则 $C_1C_2\cdots C_m$ 可作为字串 $Z_1Z_2\cdots Z_n$ 的纠错建议。

似然匹配可纠正任一字与原串不同的词条、首尾漏字、词中多字、少字等错误。

如：match(“当务之争”,“当务之急”)，match(“而走险”,“铤而走险”)等。

2.字驱动双向词典的构造

为便于查找和对词进行似然匹配，重组词典信息，构造以字为驱动的双向词典。该词典以词的首字内码大小排列。词典的数据结构定义如下：

```
struct Item
{  char * Character[2];//字
   int    Frequency;  //字频
   int    IsDanZiCi;  //是否为单字词,是为1,否则为0
   int    BeFirst;    //能否做词典中某词的首字,是1,否则0
```

```
    int     BeSecond;   //能否做词典中某词的第二字,是1,否则0
    int     BeThird;    //能否做词典中某词的第三字,是1,否则0
    int     BeFourth;   //能否做词典中某词的第四字,是1,否则0
    long *  BiHead;     //指向以该字为首字的二字词
    long *  BiTail;     //指向以该字为尾字的二字词
    long *  TriHead;    //指向以该字为首字的三字词
    long *  TriTail;    //指向以该字为尾字的三字词
    long *  QuarHead;   //指向以该字为首字的四字词
    long *  QuarTail;   //指向以该字为尾字的四字词
    long *  MultiHead;  //指向以该字为首字的多字词
    long *  MultiTail;  //指向以该字为尾字的多字词
}Zi;
```

如:Character="安"　Frequency=5067　IsDanZiCi=1　BeFirst=1
BeSecond=1　BeThird=1　BeFourth=1
BiHead→插 定 顿 放 分 抚 好 家 静 居 康 乐 眠 民 宁 排 培 全 然 设
身 生 适 泰 危 慰 稳 息 闲 祥 详 享 歇 心 逸 葬 置 装 瓿
BiTail→保 不 伏 公 苟 毫 建 柳 偏 平 请 偷 晚 微 问 相 招 治
TriHead→乐窝 理会 眠药 全部 全带 全阀 全帽 全线
TriTail→NULL
QuarHead→安静静 分守己 家落户 居乐业 民告示 全系数 然无恙 营扎寨 于现状
QuarTail→NULL　MultiHead→NULL　MultiTail→NULL

2.2.4　英文单词拼写纠错词典:骨架键词典的生成[5~7]

1.纠错思想

利用英文单词的冗余信息对单词进行重构,拼写纠错时,将报错的单词按照骨架键似然函数构造骨架键并在骨架键纠错词典中查找相应骨架键,若命中,相应单词为纠错建议。

2.骨架键生成算法

设字母集$\sum=\{'a','b',\cdots'z','A','B',\cdots'Z'\}$,英文单词词典记为$\Omega$;$W_k=L_1L_2\cdots L_m$为某一英文单词,$L_i(1\leqslant i\leqslant m)\in\sum$,m为词长,$W_k\in\Omega$;元音字母记为Vowel={'a','e','i','o','u','A','E','I','O','U'};辅音字母记为Consonant=$\sum$-Vowel;单词第一个字母记为FirstLetter=L_1;单词中元音字母的唯一序列记为Vow_seq;辅音字母的唯一序列记为Con_seq;则单词Wk的骨架键生成算法如下:

```
Vow_seq=Φ;Con_seq=Φ;
for i=2 to m
{if (Li∈Vowel and Li∈~Vow_seq)Vow_seq=Vow_seq+Li;
  if (Li∈Consonant and Li∈~Con_seq)Con_seq=Con_seq+Li;
}
```

单词W_k的骨架键即为:$Skeleton\_key_k$=FirstLetter+Con_seq+Vow_seq。

骨架键词典格式为:	骨架键(Skeleton_key$_k$)	单词(W_k)	
如:	Clfrnaio	Californian	California
	Plstnaei	Palestinian	Palestine

三、纠错建议产生算法和候选建议排序方法

3.1 纠错建议产生算法

设字串 $Z=Z_1Z_2Z_3\ldots Z_s\ldots Z_m$ 为查出的错误字串，错误字串 Z 中字 Z_s 的某一候选字为 $C_{si}\in\{C_{s1},C_{s2},\ldots C_{sl}\}$，$\{C_{s1},C_{s2},\ldots C_{sl}\}$ 为 Z_s 的所有候选字集。则 Z 的纠错候选建议产生算法如下：

1. 纠错建议计数器 i 初始化为 0；

2. 若错误为英文单词拼写错误，将出错单词按骨架生成算法生成骨架键，查骨架键词典，若命中，对应的单词加入纠错建议缓冲区。*{纠正英文单词错误}*

3. 查易混淆词词典，若命中，将该纠错建议加入纠错建议缓冲区；计数器 i 加 1。*{纠正"易混淆词"错误}*

4. 查纠错候选词集 Sci，若命中，则将该纠错建议加入纠错建议缓冲区；纠错建议计数器 i 加 1。若纠错建议不止一个，循环做 3，直至取完所有纠错建议。*{纠正"多字替换"错误}*

5. 若字串 Z 的长度 Length(Z)=1(以字计)，即字串 $Z=Z_1$

   ①取候选字集 S_{z1} 中相应候选字中的前 20 个加入纠错建议缓冲区，i 加 20。*{纠正"别字"错误}*

   ②若 Z_1.IsDanZiCi=0，即 Z_1 不是单字词，取 Z_1 为首字的二字词和以 Z_1 为尾字的二字词加入纠错建议缓冲区，计数器 i 加 1。*{纠正"漏字"错误}*

6. 若字串 Z 的长度 Length(Z)=2(以字计)，即字串 $Z=Z_1Z_2$

   ①若 Z_2Z_1 成词，则将 Z_2Z_1 加入纠错建议缓冲区；纠错建议计数器 i 加 1。*{纠正"易位"错误}*

   ②若 $C_{1i}Z_2$ 成词或 Z_1C_{2j} 成词，则将 $C_{1i}Z_2$ 或 Z_1C_{2j} 加入纠错建议缓冲区；计数器 i 加 1。*{纠正"别字"错误}*

   ③取以 Z_1 为首字的三字词和以 Z_2 为尾字的三字词中与 Z_1Z_2 词似然匹配的词加入纠错建议缓冲区，纠错建议计数器 i 加 1。*{纠正"漏字"错误}*

7. 若字串 Z 的长度 Length(Z)=3(以字计)，即字串 $Z=Z_1Z_2Z_3$

   ①若 $C_{1i}Z_2Z_3$ 成词，或 $Z_1C_{2j}Z_3$ 成词或 $Z_1Z_2C_{3k}$ 成词，则将 $C_{1i}Z_2Z_3$ 或 $Z_1C_{2j}Z_3$ 或 $Z_1Z_2C_{3k}$ 加入纠错建议缓冲区；纠错建议计数器 i 加 1。*{纠正"别字"错误}*

   ②若 Z_1Z_2 成词或 Z_1Z_3 成词或 Z_2Z_3 成词，则将 Z_1Z_2 或 Z_1Z_3 或 Z_2Z_3 依次加入纠错建议缓冲区；纠错建议计数器 i 加 1。*{纠正"多字"错误}*

   ③取以 Z_1 为首字的四字词和以 Z_3 为尾字的四字词中与 $Z_1Z_2Z_3$ 词似然匹配的词作为纠错建议，加入纠错建议缓冲区，纠错建议计数器 i 加 1。*{纠正"漏字"错误}*

8. 若字串 Z 的长度 Length(Z)≥4(以字计)，即字串 $Z=Z_1Z_2\ldots Z_m$($4\leqslant m\leqslant 7$，这里假设错误串长为 7)

   取以 Z_1 为首字和以 Z_m 为尾字，长度分别为 Length(Z)－1、Length(Z)、Length(Z)＋1 且与字串 Z 似然匹配的多字词作为纠错建议，依次加入纠错建议缓冲区；纠错建议计数器 i 加 1。*{纠正"多字"、"漏字"、"易位"、"多字替换"和"别字"错误}*

9. 如果 i=0；提示由用户给出纠错建议，并将纠错建议加入易混淆词词典。

10. 将所得纠错建议进行排序(见候选建议的排序方法)，取前 10 个作为纠错建议。

11. 返回

3.2 候选建议的排序方法

由纠错候选集和经过似然匹配得出的纠错建议并非都是有效的。在纠错时根据用字信息及上下文相关字词的同现信息计算纠错建议的优先值 ρ，并根据优先值 ρ 对纠错建议进行排序，将优先值大的纠错建议放在前面，可提高纠错效率和可读性。

纠错建议的字字间最小同现频率在一定程度上反映纠错建议的“词频”信息，设 $S_1S_2\cdots S_m$ 为所得到的纠错建议，$r(S_i,S_{i+1})$ 为相邻字 S_i 与 S_{i+1} 的同现频率，我们令最小字字同现频率为优先值 ρ，ρ 的值越大，纠错建议的优先级就越高。ρ 的计算公式如下：

$$\rho = \min_{i=1}^{m-1} r(S_i, S_{i+1})$$

四、实验结果

利用上述算法对含有 824 个真实错误的 100 篇文本进行开放式纠错测试，这些错误中包括了拼音、五笔字型输入法造成的错误以及多字、少字、易位等错误，将这些错误由人工标记。使用所提的纠错算法对实验文本进行测试，纠错算法对 587 个错误点给出的前五选纠错建议正确，纠错正确率为 71.24%，证明本文所提纠错算法是有效的。

自动校对要求纠错算法给出纠错建议速度快、准确且数量不能太多。本文在对错误特征充分分析的基础上，结合似然匹配算法，建立纠错知识库，据此计算产生纠错建议，并利用最小字字同现频率，对候选建议进行排序和排除。实验表明，所给的纠错算法对拼音、五笔字型录入时产生的错误以及多字、少字、易位等错误进行纠错是切实可行的。

参 考 文 献

[1] 张仰森，丁冰青. 中文文本自动校对的技术现状及展望. 中文信息学报，1998，(3)

[2] 孙才，罗振声. 汉语文本校对字词级查错处理的研究. 见：第四届计算语言学会议论文集，北京：清华大学出版社，1997

[3] 郭志立等. 中文校对系统中的修改建议提供算法. 见：第四届计算语言学会议论文集，北京：清华大学出版社，1997

[4] 于勐，姚天顺. 一种混合的中文文本校对方法，中文信息学报，1998，(2)

[5] 张仰森，丁冰青. 一种英文单词拼写查错和纠错的方法——骨架键法. 电脑开发与应用. 1999，(2)

[6] Joseph J Pollock. Automatic spelling correction in scientific and scholarly text. Communication of the ACM，1984，(4)

[7] James L Peterson. computer programs for detecting and correcting spelling errors. Communication of the ACM，1980，(12)