ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.18, No.9, September 2007, pp.2162−2173
DOI: 10.1360/jos182162

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于多通道融合的连续手写识别纠错方法*

敖 翔+, 王绪刚, 戴国忠, 王宏安

(中国科学院 软件研究所 人机交互技术与智能信息处理实验室,北京 100080)

Error Correction of Continuous Handwriting Recognition by Multimodal Fusion

AO Xiang+, WANG Xu-Gang, DAI Guo-Zhong, WANG Hong-An

(Human Computer Interaction and Intelligent Information Processing Laboratory, Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China)

+ Corresponding author: Phn: +86-10-63951881 ext 8511, E-mail: xiang_ao@hotmail.com

Ao X, Wang XG, Dai GZ, Wang HA. Error correction of continuous handwriting recognition by multimodal fusion. *Journal of Software*, 2007,18(9):2162−2173. http://www.jos.org.cn/1000-9825/18/2162.htm

Abstract: In recognition-based user interface, users' satisfaction is determined not only by recognition accuracy but also by effort to correct recognition errors. In this paper, an error correction technique based on multimodal fusion is introduced. It allows a user to correct errors of Chinese handwriting recognition by repeating the handwriting in speech. A multimodal fusion algorithm is the key of the technique. By constraining the search for the best handwriting recognition result by speech input, the algorithm can correct errors in both character extraction and recognition of handwriting. The experimental result indicates that the algorithm is effective and efficient in computation. Moreover, evaluation also shows the correction technique can help users to correct errors in handwriting recognition more efficiently than the other two error correction techniques.

Key words: error correction; multimodal fusion; handwriting recognition; speech; phoneme; weighted phoneme

摘 要: 在基于识别的界面中,用户的满意度不但由识别准确度决定,而且还受识别错误的纠正过程的影响.提出一种基于多通道融合的连续手写笔迹识别错误的纠正方法.该方法允许用户通过口述书写内容纠正手写识别中的字符提取和识别的错误.该纠错方法的核心是一种多通道融合算法.该算法通过利用语音输入约束最优手写识别结果的搜索,可纠正手写字符的切分错和识别错.实验评估结果表明,该融合算法能够有效纠正错误,计算效率高.与另外两种手写识别错误纠正方法相比,该方法具有更高的纠错效率.

关键词: 错误纠正;多通道融合;手写识别;语音;音素;加权音素

中图法分类号: TP391 **文献标识码**: A

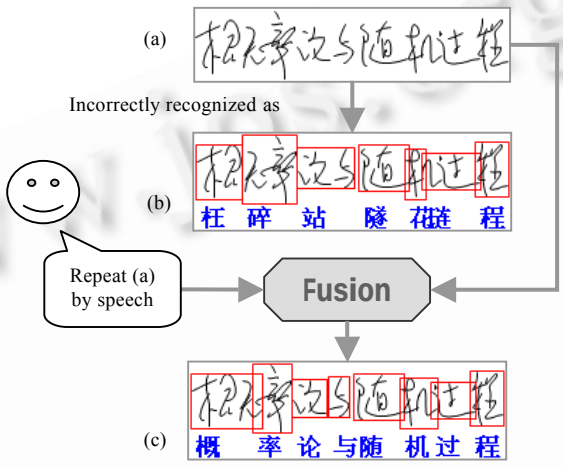
通过手写将信息录入计算机,正逐渐成为被广泛使用的输入方式.一般来说,书写的笔迹会被识别成正文.然而,由于手写识别经常出错,手写输入的自然性和效率都大受影响[1,2].尽管有不少研究致力于提高手写识别

* Supported by the National Basic Research Program of China under Grant No.2002CB312103 (国家重点基础研究发展计划(973)); the National Natural Science Foundation of China under Grant No.60503054 (国家自然科学基金); the Key Innovation Project from Institute of Software, the Chinese Academy of Sciences (中国科学院软件研究所创新基金重大项目)

Received 2006-04-25; Accepted 2006-05-18

的正确率,但识别错误仍难以完全消除.因此,基于手写识别的系统必须支持识别错误的纠正.事实上,在基于识别的系统中,用户的满意度不但受识别正确率的影响,而且还受识别错误纠正过程的影响.错误纠正是否自然、高效,直接影响用户体验.

本文将介绍一种基于多通道融合的连续手写笔迹识别错误的纠正方法.该方法可以让用户通过语音复述书写的内容,纠正手写笔迹识别错误,简称为"语音纠错".图 1 示例了该方法的使用过程:图 1(a)中的手写句子被错误地识别为图 1(b)所表示的结果——字符的提取和识别都存在错误.为了纠正这些错误,用户复述这句话.通过融合笔迹与语音对同一内容的输入,手写识别错误得以纠正(如图 1(c)所示).该纠错方法的核心是一个笔迹与语音的多通道融合(multimodal fusion)算法.该融合算法的主要思想是利用用户的语音约束对最优手写识别结果的搜索.实验结果表明:该纠错方法能够有效地纠正手写识别错误;与另外两种纠错方法相比,该方法的纠错效率较高.



(a) The original handwriting; (b) The incorrect handwriting recognition result;
(c) The correct recognition result by fusing the handwriting and speech
(a) 原始笔迹; (b) 错误识别结果; (c)通过融合用户语音与笔迹获得的的正确识别结果

Fig.1 An example of error correction of handwriting by speech

图 1 一个利用语音纠正手写识别错误的例子

本文第 1 节给出研究动因.第 2 节回顾相关工作,并与本文方法进行比较.第 3 节详述纠错方法中的多通道融合算法.第 4 节给出实验结果并加以讨论.第 5 节总结全文.

1 研究动因

采用笔迹与语音融合的方式来纠正手写识别错误,基于以下几个原因:第一,语音纠错自然.人们通常采用默读的方式来校对文档,语音纠错与此方法类似(区别只在于是否读出声).研究表明,模仿人们日常习惯的纠错方法更能被用户接受[3];第二,语音纠错高效.通常,利用多个通道进行交互效率比较高[4,5].此外,使用语音的操作代价小,让用户复述一遍书写的内容并不会明显增加操作负担.更重要的是,在使用计算机时,用户的双手繁忙,采用语音纠错可以避免增加用户双手更多的工作负担;第三,语音纠错效果好.研究发现,利用两个或多个互补通道的融合结果作为输入的系统,能有效降低识别错误发生率,因而具有较好的鲁棒性[6,7].唇读识别(audio-visual speech recognition,简称AVSR)[8,9]正是成功利用多通道融合的例子.此外,利用跨通道相关性(cross-modal dependency)的多通道融合,能显著提高单通道识别正确率[10].本文提出的语音纠正手写识别错误的方法,正是利用了笔迹与语音两个输入通道的跨通道影响(cross-modal influence)而达到纠错目的.

2 相关研究

关于纠正识别错误的研究已经进行了多年,这些研究主要集中在语音识别错误的纠正上."复述(respeaking)"是一类常用的识别纠错策略.用户复述被识别错误的内容,计算机识别用户的复述,将已有的识别结果替换为复述的识别结果."复述"的优点是交互非常自然.但由于用户复述的内容仍可能被识别错,因此,"复述"在实际使用中效果并不理想[10]."拼写(spelling)"是一类主要应用于西文文字识别的纠错策略.用户通过口述单词的字母序列达到纠错的目的.然而在实际应用中,"拼写"既不自然也不高效[10,11],因而也非理想的纠错策略."候选列表(N-best list)"是另一类典型的识别纠错方式.识别器通常并不只返回单一识别结果,还返回多个识别候选.用户通过在识别候选中选择正确结果,达到纠错目的[12].然而,如果候选列表中不包含正确结果,纠错就不能进行.

如果考虑多通道交互,纠错效率将得以提高.一种直接的方式是为用户提供多种输入通道选择,让用户能在不同通道间切换[11].例如,如果语音识别容易出错,用户就可以选择笔迹输入.另一种利用多个通道输入纠错的策略,称为"多通道纠错(multimodal correction)"或"跨通道纠错(cross-modal correction)".该策略并非简单地利用一个通道输入的识别结果代替另一个通道输入的识别结果,而是利用通道间的互补性,通过不同识别结果的融合来纠正识别错误[10,13−16].多通道纠错技术已经应用在语音与笔迹[15]、语音与传统GUI通道[13]、语音与眼动[14]等通道间的纠错上.

本文的纠错方法在使用上与"复述"类似,但它是一种多通道纠错方法,与采用朴素替代策略的"复述"纠错并不相同.到目前为止,还鲜有研究致力于手写识别错误的多通道纠正.与本文工作最接近的是我们之前的一项研究,该研究提出了一种利用语音纠正连续手写识别中字符识别错的方法[15],它并不能纠正字符提取错,而字符提取错在连续手写识别中很常见.本文的方法既可以纠正字符识别错,也可以纠正字符提取错.

利用多通道融合控制识别错误的融合策略可归为 3 类:"早期融合(early fusion)"、"晚期融合(late fusion)"以及"混合融合(hybrid fusion)".早期融合又称为"特征层融合(feature-level fusion)",指的是将不同通道输入的特征组合为一个特征向量,然后进行识别[18].早期融合通常适用于时间上同步的通道融合,如AVSR中语音与唇动的融合[8].晚期融合又称为"语义层融合(semantic level fusion)",指的是融合各个通道独立识别的结果.许多多通道输入系统都采用晚期融合[19,20],这是因为晚期融合比早期融合更加灵活.混合融合通常是指利用一个通道的识别结果去限制、指导和优化另一个通道的识别[13,15,21,22].

我们之前的工作[15]是一个混合融合的例子.它利用多个字的手写识别结果候选矩阵作为语言模型,去限定语音识别的结果.本文中的多通道融合算法采用的也是混合融合,与用手写识别结果限定语音识别相反,该方法是利用语音识别的中间结果去优化手写笔迹识别结果的搜索.

3 融合算法

本文提出的纠错方法的核心是一种多通道融合算法.该算法的基本思想是利用语音限制和优化最优笔迹识别结果的搜索,原理如下:考虑句子Y,它的连续手写笔迹表示X_{hw},用户对它的语音输入表示为音频信号X_{sp}.我们用$\{W\}$表示所有可能的X_{hw}的笔迹识别结果,W^*表示笔迹与语音的融合结果(W和每一个W_i都为字符序列).我们有

$$W^*=\text{argmax}_W P(W|X_{hw},X_{sp}) \tag{1}$$

利用贝叶斯公式,式(1)改写为

$$W^*=\text{argmax}_W P(W|X_{hw})P(X_{sp}|W,X_{hw}) \tag{2}$$

由于X_{hw}和X_{sp}对W条件独立,因此式(2)化简为

$$W^*=\text{argmax}_W P(W|X_{hw})P(X_{sp}|W) \tag{3}$$

我们用S_W表示W的发音.因为$P(S_W|W)=1$,所以

$$P(X_{sp}|W)=P(X_{sp}|W,S_W) \tag{4}$$

由于W和X_{sp}可看成对S_W条件独立,即

$$P(X_{sp}|W,S_W)=P(X_{sp}|S_W) \tag{5}$$

因此,式(3)改写为

$$W^*=\text{argmax}_W P(W|X_{hw})P(X_{sp}|S_W) \tag{6}$$

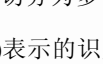
在式(6)中,$P(W|X_{hw})$可看成对X_{hw}的笔迹识别,$P(X_{sp}|S_W)$可看成是用笔迹识别结果的发音去匹配用户输入的语音.因此,本文多通道融合的目的可明确为:搜索发音最匹配用户语音X_{sp}的笔迹X_{hw}的识别结果W^*.

为了实现式(6)所提出的目标,有 3 个问题需要解决:

1) 由于W^*从所有可能的识别结果W中选择,那么集合$\{W\}$是什么?通常,字符识别会返回多个候选结果,字符提取的方式也可以有多种.这些多样性都表明存在一个巨大的手写识别结果空间.第 3.1 节将讨论此问题.
2) S_W与X_{sp}匹配中的“比较”如何进行?“比较”只能在可比的对象间进行,因此,S_W和X_{sp}需要同样的表示格式.本文融合算法使用“音素”(phoneme)来表示S_W和X_{sp}.音素是文字发音的符号化表示,第 3.2 节给出音素及其之间的相似度的定义.由于用户语音的音素表示是靠语音识别获得的,因此,由于语音识别会出错,所以,其音素表示可能不准确.为此,我们引入“加权音素”来更准确地表示用户输入的语音.加权音素将在第 3.5 节中加以介绍.
3) 融合过程是一个搜索过程,怎样保证搜索的效率?用户很关心纠错效率,如果某种纠错方式非常耗时,那么它就没有任何意义,因为用户完全可以选择其他高效的纠错方式.第 3.3 节指出,如果采用朴素的穷举搜索来融合,效率极低.然而,如果采用第 3.4 节中介绍的分治策略,效率则会大为提高.

3.1 手写识别的错误和候选

连续手写识别的错误可以分为两类:字符识别错误和字符提取错误.字符识别错误,是指手写字符被识别为非其对应的正文字符.例如,手写字符  .被错识为“棍”,而非正确结果“概”.通常,字符识别并不只返回单一识别结果,还返回多个识别候选.正确的识别结果极可能存在于候选列表中.

字符提取错误,是指笔迹在切分为多个手写字符时出现的错误,提取出的字符或是缺失其应有的笔划或是包含了不属于它的笔划.图 1(b)表示的识别结果就包含了多处字符提取错误,例如,  被错切为,其实应为  .一般来说,如果字符提取有错,后续的字符识别是没有意义的.

与字符识别返回多个候选一样,我们让字符提取也有多种候选.“过切分(over-segmentation)”是一种产生多个字符提取候选的方法,它是指将一行笔迹切分后,提取出的手写字符或是完整字符,或只包含完整字符的一部分.我们把通过过切分而得到的字符称为“片断(fragment)”.图 2 给出了对图 1(a)中笔迹的过切分结果(图 1(a)中的句子被过切分为 13 个片断).

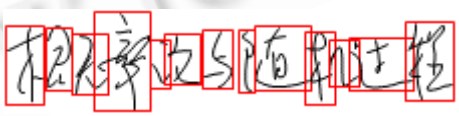


Fig.2 Over-Segmentation of a sentence

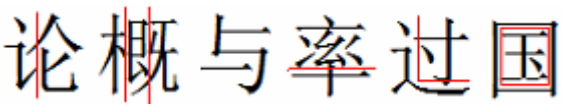
图 2 笔迹的过切分

设句子S被过切分为片断序列$F=f_0f_1\ldots f_{T-1}$,其中,f_i代表一个片断.易见,序列F的任何一个子序列$f_jf_{j+1}\ldots f_k$,$0\le i\le k<T$都可能构成一个字符,因此,S的一个有M个字符的切分结果$\boldsymbol{S}$可表示为

$$\boldsymbol{S}=(f_0\ldots f_{k_1})(f_{k_1+1}\ldots f_{k_2})\ldots(f_{k_{M-1}+1}\ldots f_{T-1}) \tag{7}$$

我们关心一个构成汉字的片断子序列至多包含几个片断.由汉字结构可知,汉字字形结构可归为 6 种模式,图 3 示例了这 6 种模式(其中,左中右结构对切分的影响最大).对于水平方向书写的笔迹来说,左中右结构对字符提取的影响最大,因此,可认为一个字至多由 3 个片段构成(左中右结构可看成有 3 个水平排列的部件),因此,片

段子序列的最大长度为 3.



(a) (b) (c) (d) (e) (f)

(a) Left-Right; (b) Left-Middle-Right; (c) Single; (d) Up-Bottom; (e) Half-Surrounded; (f) Fully-Surrounded

(a) 左右; (b) 左中右; (c) 独体; (d)上下; (e) 半包围; (f) 全包围

Fig.3 Six graphemic patterns of Chinese characters

图 3 汉字的 6 种字形结构

我们把片段组织成有向图$G=\langle V,E\rangle$.G中的顶点为各个片段$\{f_0f_1\ldots f_{T-1}\}$和附加顶点f_T.每个顶点都与其 3 个后续顶点有边相连(如果后续顶点存在),顶点间的顺序由其对应片段之间的顺序决定,有

$$\begin{cases} G=\langle V,E\rangle \\ V=\{f_i \mid i\in[0,T-1]\}\cup\{f_T\} \\ E=\{\langle f_i,f_j\rangle \mid (j\le T)\wedge(0\le i<j\le i+3)\} \end{cases} \tag{8}$$

图 4 示例了包含 7 个片段的G,其中f_7是一个附加顶点,它并不对应片段.我们可以很容易地从G枚举所有可能的字符提取结果.每种提取结果都与G中一条从f_0开始到f_T结束的路径相互对应.例如,在图 4(注意,f_7是附加节点)中,路径$f_0\to f_1\to f_4\to f_5\to f_7$对应字符提取结果$(f_0)(f_1f_2f_3)(f_4)(f_5f_6)$.

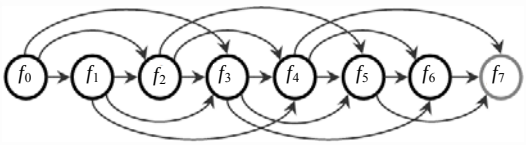


Fig.4 The graph composed of seven fragments

图 4 包含 7 个片断的图

3.2 音 素

音素是字符发音的符号化表示.我们使用汉语拼音来表示汉字的音素.普通话中每个字都有其拼音.拼音由声母(initial,*in*)、韵母(final,*fn*)和声调(tone)构成(无声母的拼音可看成带有空(null)声母).例如,“逃”的拼音为“táo”,其中声母为“t”,韵母为“ao”,声调为“´”.音素 *ph* 是一个“声母-韵母”对,其表示为

$$ph=[in,fn] \tag{9}$$

笔迹识别结果的音素表示可通过查字典的方式获得;语音的音素表示可通过转换语音识别结果,或是直接利用语音识别的中间结果(如果语音识别器支持输出发音识别结果)未获得.汉语拼音包含 23 种声母和 38 种韵母.为避免混淆,我们用in^i表示第i种声母,fn^j表示第j种韵母;用in_k和fn_k分别表示音素ph_k的声母和韵母.

给定两个音素$ph_1=[in_1,fn_1]$和$ph_2=[in_2,fn_2]$,它们的相似度$S(ph_1,ph_2)$定义为

$$S(ph_1,ph_2)=sIn(in_1,in_2)+sFn(fn_1,fn_2) \tag{10}$$

其中,$sIn(in_1,in_2)$和$sFn(fn_1,fn_2)$分别表示声母间的相似度和韵母间的相似度.$sIn(in_j,in_k)$和$sFn(fn_j,fn_k)$的定义来自于经验.直观地,如果声母in_1和in_2发音很相似,那么$sIn(in_1,in_2)$接近 0;如果发音差别很大,则$sIn(in_1,in_2)$相似度接近 1. $sFn(fn_j,fn_k)$同理.

再定义音素序列间的相似度.因为音素ph表示为$[in,fn]$,所以音素序列$PH=ph_1ph_2\ldots ph_{N-1}$可表示为符号序列$PH=in_0fn_0in_1fn_1\ldots in_{N-1}fn_{n-1}$,其中每个符号是一个声母或者韵母.对于两个符号序列的相似性比较,Levenshtein距离(Levenshtein distance)[23]是一个合适的度量.Levenshtein距离计算的是为了将一个符号串变换成另一个符号串所需要的字符插入、删除和替换的次数.由于计算的是操作的次数,因此在Levenshtein距离中,3 种符号变换的代价都为 1.我们采用一种改进Levenshtein距离$LD(PH_1,PH_2)$作为两个音素序列PH_1和PH_2的相似度.

$LD(PH_1,PH_2)$中,替换操作的代价被重定义为

$$cost\_sub(a,b)=\begin{cases} sIn(a,b), & \text{if both } a \text{ and } b \text{ are intials} \\ sFn(a,b), & \text{if both } a \text{ and } b \text{ are finals} \\ \infty, & \text{otherwise} \end{cases} \tag{11}$$

由于声母替换韵母或者韵母替换声母没有意义,因此在式(11)中,当替换的两项不同类时,替换代价为无穷大.

3.3 采用穷举搜索的融合

一种朴素的融合策略是:枚举所有可能的笔迹识别结果,对于每种识别结果,获得它的音素序列表示,然后将此音素序列与语音输入的音素序列匹配,匹配度最大的音素序列对应的笔迹识别结果即为融合结果.假设笔迹表示为片断序列$F=f_0f_1\ldots f_{T-1}$,F的每一种切分结果表示为手写字符序列 $\boldsymbol{S}_i=w_{i,0}w_{i,1}\ldots w_{i,|\boldsymbol{S}_i|-1}$,其中,每个$w_{i,j}$是$F$的一个子序列.每个$w_{i,j}$被识别为$k$个候选汉字,每个汉字的发音表示为音素$ph_{i,j,k}$.因此,切分$\boldsymbol{S}_i$的一个可能的音素序列$PH_{hw}$为 $ph_{i,0,t_0}ph_{i,1,t_1}\ldots ph_{i,|\boldsymbol{S}|-1,t_{|\boldsymbol{S}|-1}}$ $(0\le t_r<k)$.再假设用户关于这段笔迹的语音输入的表示为音素序列$PH=ph_1ph_2\ldots ph_{N-1}$.因此,采用穷举搜索的融合算法可用图 5 中的伪代码表示.图 5 中,函数$ExFusion(F,PH)$返回的是最小融合代价.

```
ExFusion(F,PH) return min_cost
    for every segmentation S_i of F
        for every phoneme sequence PH_hw of S_i
            cost=LD(PH_hw,PH)
            min_cost=min(cost,min_cost)
        end for PH_hw
    end for S_i
end procedure
```

Fig.5 The pseudo code of the exhaustive fusion of fragment sequence F and phoneme sequence PH

图 5 采用穷举搜索的融合代码.F 为笔迹,PH 为语音

我们估计 $ExFusion$ 的时间复杂度.由图 4 可知,对于有 T 个片断的笔迹来说,其可能的字符提取结果数 $C(T)$ 可以按式(12)计算.

$$\begin{cases} C(T)=\sum_{i=1}^{3}C(T-i) \\ C(1)=1;C(2)=2;C(3)=4 \end{cases} \tag{12}$$

可见,$C(T)$是一个Tribonacci数[24],它等于$|\alpha\times\beta^{T+1}|$.其中,$|x|$表示最接近$x$的整数,$\alpha\approx0.336$,$\beta\approx1.839$.计算可知,一个包含 24 个片断的笔迹——对应 8~24 个字——就有上百万种可能的切分方式.

假设单字符识别返回k个候选字,那么,一个含有M个字符的切分结果就有k^M个可能的音素序列(此处我们假设每个字的k个识别候选的发音各不相同).我们又知道音素序列匹配$LD(PH_1,PH_2)$的时间复杂度为$O(|PH_1|,|PH_2|)$[8].因此,$ExFusion$的时间复杂度为

$$O\left(\sum_{i=0}^{|\alpha\times\beta^{T+1}|}k^{|\boldsymbol{S}_i|}(|\boldsymbol{S}_i|\times N)\right) \tag{13}$$

其中,$|\boldsymbol{S}_i|$表示$\boldsymbol{S}_i$中的字符数,并且$T/3\le|\boldsymbol{S}|\le T$.可见,$ExFusion$的效率很低.

3.4 采用分治策略的融合

我们采用分治策略来降低融合搜索过程的计算复杂度.注意到,片断序列$F=f_0\ldots f_{i-1}f_if_{i+1}f_{i+2}\ldots f_{T-1}$中,$f_{i-1}f_if_{i+1}$和$f_{i+2}$这 4 个片段不可能同在一个字符里,这是因为我们已经规定一个字符至多含有 3 个片断.如果两个相邻的片断不在同一字符里,它们被称作是“分离”的.在F中,f_{i-1}与f_i, f_i与f_{i+1},以及f_{i+1}与f_{i+2}这3对片断里,肯定有一对是分离的.两个分离的片断将序列F分为前后两个子序列,这称为序列F的一个分割.例如,如果f_i与f_{i+1}分离,则F被分割

为两个子序列:

$$\begin{cases} F_{0,i} = f_0 \cdots f_{i-1} f_i \\ F_{i+1,T-1} = f_{i+1} f_{i+2} \cdots f_{T-1} \end{cases} \tag{14}$$

采用分治策略的融合算法如下:我们定义分治融合代价为$DCFusion(F_{i,j},PH_{k,l})$,其中,$F_{i,j}$为片断序列(第$i$个片断到第$j$个片断),$PH_{k,l}$为语音音素序列.我们有

$$DCFusion(F_{i,j},PH_{k,l}) = \begin{cases} ExFusion(F_{i,j},PH_{k,l}), & \text{if } j-i<threshold \\ \min\limits_{t-1\le p\le t+1}(DCCost(p,F_{i,j},PH_{k,l})), & \text{else} \end{cases} \tag{15}$$

其中,

$$t = (i+j)/2 \tag{16}$$

且

$$DCCost(p,F_{i,j},PH_{k,l}) = \min_{k\le q\le l}(DCFusion(F_{i,p},PH_{k,q}) + DCFusion(F_{p+1,j},PH_{q+1,l})) \tag{17}$$

式(15)和式(16)表明:如果$F_{i,j}$的长度小于阈值$threshold$,那么融合就直接采用第 4.3 节中的穷举方法;否则,对其 3 种分割结果递归地采用分治融合.在算法实现中,阈值$threshold$=5.式(17)表明:一种分割的两个片断子序列要尝试匹配所有可能的语音音素序列的切割结果.

我们可以缩小式(17)中的参数q的变化范围,而在绝大多数情况下不失融合精度,因而可以提高融合效率.事实上,片段子序列$F_{i,p}$只需要去匹配和它指示相同字符范围的音素子序列,而不必匹配由式(17)给出的所有可能的音素子序列.我们定义$Len(i,p)$为从片断f_i到片断f_p的几何距离,定义式(18)为f_p在$F_{i,j}$中归一化后的距离.

$$Pos(p) = Len(i,p)/Len(i,j) \tag{18}$$

事实上,$Pos(p)$也指示了潜在字符序列中片断f_i所属于的字符的归一化距离.由于每个字符对应$PH_{k,l}$中的一个音素,因此,音素子序列$PH_{k,PhIdx(p)}=ph_k ph_{k+1}\ldots ph_{PhIdx(p)}$(其中,$PhIdx(p)=k+|PH_{k,l}|\times Pos(p)$)是最应被$F_{i,p}$所匹配的音素子序列.图 6 示例了$PhIdx(p)$的计算(在本例中,最应该与片断子序列$f_0f_1\ldots f_6$匹配的音素子序列是$ph_0ph_1ph_2ph_3$).现在,我们把式(17)中参数$q$的变化范围设置为$PhIdx(p)-\lambda\le q\le PhIdx(p)+\lambda$.在算法实现中,$\lambda$=2.

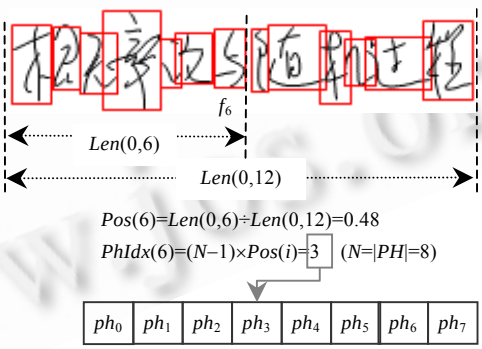


Fig.6 Finding of the sub-sequence of speech phoneme sequence matching the fragment sequence

图 6 寻找最应该与片断子序列匹配的音素子序列

综上所述,采用分治搜索的融合算法如图 7 所示(该算法返回最小融合代价).

我们估计 *DCFusion* 的执行效率,看它是否比 *ExFusion* 更高效.如果对图 1 中的笔迹进行 *DCFusion* 融合,整个问题会分解为 775 个 *ExFusion* 子问题,这些子问题的参数 *T*,*M* 和 *N* 都约为原问题中各参数的四分之一.分析可知,*DCFusion* 的时间复杂度约为

$$O(E(t)\times 30^{\log_2(|F|/t)}) \tag{19}$$

其中,t 是 *DCFusion* 式中的 *threshold*,$E(t)$是在 t 个片断上执行 *ExFusion* 执行时间的期望.比较 *DCFusion* 和

ExFusion 的时间复杂度可知,*DCFusion* 比 *ExFusion* 高效得多.

```
DCFusion(F_{i,j},PH_{k,l}) return min_cost
    if j−i≤threshold
        min_cost=ExFusion(F_{i,j},PH_{k,l})
    else
        for m=(i+j)/2−1 to (i+j)/2+1
            for n=PhIdx(m) to PhIdx(m)+2
                cost=DCFusion(F_{i,m},PH_{k,n})+
                    DCFusion(F_{m+1,j},PH_{n+1,l})
                min_cost=min(cost,min_cost)
            end for n
        end for m
    end if
end procedure
```

Fig.7 The pseudo code of the divide-conquer approach to minimize the cost of matching a fragment sequence $F_{i,j}$ to a phoneme sequence $PH_{l,m}$

图 7 采用分治搜索的融合算法的伪代码,最小化片断序列$F_{i,j}$与语音音素序列$PH_{l,m}$的匹配代价

3.5 加权音素

用户语音的音素是靠语音识别获得的.与手写识别一样,语音识别也会出错,这使得语音音素不准确.不过,语音识别往往也会返回多个识别候选,利用这些候选能够更准确地表示用户语音.我们使用“加权音素(weighted phoneme)”来表示综合语音识别候选的语音音素.

加权音素 *wph* 可看成是多个音素的组合,它由一对“加权声母-加权韵母”[*win*,*wfn*]构成,其中,*win* 是加权声母,*wfn* 是加权韵母.定义

$$\begin{cases} win=[w_0,...,w_{22}],\ \forall w_{j\in[0,22]},\ w_j\geq 0,\ \sum_{j=0}^{22} w_j\leq 1 \\ wfn=[v_0,...,v_{37}],\ \forall v_{k\in[0,37]},\ v_k\geq 0,\ \sum_{k=1}^{37} v_k\leq 1 \end{cases} \tag{20}$$

其中,w_i和v_j代表权值.因此,加权声母*win*是全部声母的线性组合;加权韵母*wfn*是全部韵母的线性组合.在不失含义的情况下,*win*和*wfn*的表示可简化为

$$\begin{cases} win=[w_0\cdot in^0,w_1\cdot in^1,...,w_{M-1}\cdot in^{22}] \\ wfn=[v_0\cdot fn^0,v_1\cdot fn^1,...,v_{N-1}\cdot fn^{37}] \end{cases} \tag{21}$$

一般来说,大多数w_i和v_j都等于 0.因此,我们可以略去权值为 0 的项.音素也是加权音素,它的*win*和*wfn*都只有 1 项,且权值为 1.

设字符*C*的语音识别结果为音素候选$ph_0,ph_1,\ldots,ph_{k-1}$.每个候选音素$ph_i=[in_i,fn_i]$的置信度(confidence)为$t_i$.我们用加权音素*wph*来表示这个语音识别结果,*wph*的计算为

$$\begin{cases} w_i=\sum_{j=0}^{k}[t_j\times equal(in_j,in^i)] \\ v_i=\sum_{j=0}^{k}[t_j\times equal(fn_j,fn^i)] \end{cases},\ \text{where } equal(a,b)=\begin{cases}1, & \text{if } a \text{ and } b \text{ are same}\\ 0, & \text{else}\end{cases} \tag{22}$$

例如,“逃”的语音识别为[*t*,*ao*],[*t*,*iao*],[*d*,*ao*],[*t*,*ou*],置信度分别为t_0,t_1,t_2和t_3,则该识别结果的加权音素为

$$wph_{逃}:\begin{cases} win_{逃}=[t_2\cdot d,(t_0+t_1+t_3)\cdot t] \\ wfn_{逃}=[(t_0+t_2)\cdot ao,t_1\cdot iao,t_3\cdot ou] \end{cases} \tag{23}$$

加权音素还能表示语音识别中的切分不确定性.例如,语音识别结果为候选(ph_0),(ph_1),$(ph_{2,0},ph_{2,1})$,置信度分别为t_0,t_1和t_2.候选 1 和候选 2 都只有一个音素,而候选 3 有两个音素.这表示语音识别器不确定用户的发音对应一个字还是两个字.我们用加权音素wph_1和wph_2来表示这个识别结果.先在候选 1 和候选 2 中分别加入空音

素$ph_{0,1}$和$ph_{1,1}$,此时,识别候选形如($ph_{0,0}$,$ph_{0,1}$),($ph_{1,0}$,$ph_{1,1}$),($ph_{2,0}$,$ph_{2,1}$),然后利用式(22),用$ph_{0,0}$,$ph_{1,0}$,$ph_{2,0}$计算wph_1,用$ph_{0,1}$,$ph_{1,1}$,$ph_{2,1}$计算wph_2.由于$ph_{0,1}$和$ph_{1,1}$是空音素,因此,wph_2的加权声母权重之和小于 1(等于t_2),其加权韵母也是如此.这表明,wph_2指出了在语音识别中存在的切分不确定性,因此更准确地反映了原始语音.通常,一个句子的语音识别结果为词组(phrases)序列,每个词组至少有一个字.词组内的语音识别结果可以用上述加权音素表示.把这些加权音素合起来,就得到了一句话的语音识别的加权音素序列表示.

我们将融合算法中的音素替换为加权音素进行计算.为此,还需要定义加权音素间的相似度.给定加权音素$wph_1=[win_1,wfn_1]$和$wph_2=[win_2,wfn_2]$,其相似度$S(wph_1,wph_2)$的计算为

$$S(wph_1,wph_2)=swIn(win_1,win_2)+swFn(wfn_1,wfn_2) \tag{24}$$

其中,$swIn(win_1,win_2)$是两个加权声母的相似度,$swFn(wfn_1,wfn_2)$是两个加权韵母的相似度.我们有

$$\begin{cases} swIn(win_1,win_2)=\sum_{j=0}^{M-1}\sum_{k=0}^{M-1}w_{1j}w_{2k}sIn(in^j,in^k) \\ swFn(wfn_1,wfn_2)=\sum_{j=0}^{N-1}\sum_{k=0}^{N-1}v_{1j}v_{2k}sFn(fn^j,fn^k) \end{cases} \tag{25}$$

4 评估与讨论

该纠错方法的实验评估关注 3 个方面的问题:a) 该方法是否有效?b) 它在计算上是否高效?c) 与其他方法相比,该方法能否拥有更高纠错效率.在实现中,手写单字符识别采用汉王手写汉字字符识别器[25],用于获取语音的音素和加权音素表示的语音识别采用微软SAPI[26].配备了 1.8GH CPU,512M内存的Tablet PC作为实验用机,其LCD显示屏支持笔输入,内置麦克风支持语音输入.实验数据为 60 个含有字符提取和识别错误的手写句子,它们分为 3 组:*D*1,*D*2 和*D*3,每组 20 个句子.定义单句识别错误率(error rate)为

$$\text{error rate}=\left(1-\frac{|\text{correctly recognized characters}|}{|\text{all characters}|}\right)\times 100\% \tag{26}$$

*D*1 中句子的错误率约为 20%;*D*2 中句子的错误率约为 45%;*D*3 中句子的错误率约为 70%.对 3 种笔迹识别纠错方法*T*1,*T*2 和*T*3 进行比较.*T*1 是指使用笔手势和N-Best List的方式纠正字符切分错和识别错[15].*T*2 是我们之前的语音纠错的方法[15],它只能纠正单字符识别错误.*T*3 是本文提出的纠错方法.12 名参与实验的被试分为人数相同的 3 组*S*1,*S*2 和*S*3,根据图 8 中的方式(例如,单元格(*S*2,*D*3)为*T*1,表示*S*2 中的被试用方法*T*1 纠正数据*D*3 中的错误),使用不同的纠错方法纠正实验数据中的识别错误.值得指出的是,当被试使用*T*2 或者*T*3 时,可使用*T*1 作为辅助,这是因为*T*2 和*T*3 这两种语音纠错方法都不能保证 100%地纠正所有的错误.我们要求在使用方法*T*3 时首先使用语音纠错.

	*D*1	*D*2	*D*3
*S*1	*T*1	*T*2	*T*3
*S*2	*T*2	*T*3	*T*1
*S*3	*T*3	*T*1	*T*2

Fig.8 Rules of how subjects correct sentences

图 8 被试进行纠错的安排规则

图 9(*Y* 轴表示纠错耗时,*X* 轴上是 3 个数据集 *D*1,*D*2 和 *D*3)给出的是 3 种纠错方法的操作效率比较.结果显示,在错误率比较低的 *D*1 上,3 种方法的纠错效率相仿.然而在错误率比较高的 *D*2 和 *D*3 上,*T*3 的效率明显优于 *T*1 和 *T*2.该实验结果容易理解:*T*1 只使用笔手势进行纠错,一般来说,纠错时间是随着错误量的增加而呈线性增加的;*T*2 只能纠正字符识别错,不能纠正字符提取错.然而,当错误率上升时,字符提取错误出现得越来越频繁,它们都需要通过 *T*1 的辅助才能纠正.*T*3 因为既能纠正字符识别错也能纠正字符提取错,因此在高错误率的情况下,纠错效率比 *T*2 和 *T*1 都要好.实验结果说明,*T*3 更适合于错误密集情况下的纠错.

图 10 通过对比纠错前后的错误数,显示了方法 *T*3 的有效性.结果显示,*T*3 能够纠正测试数据中的大部分错误(剩下的错误靠笔手势纠正).该结果也表明,引入加权音素表示语音输入,对提高纠错有效性有所帮助(图 10

显示,使用加权音素比使用音素更能有效地纠正测试集中的错误).

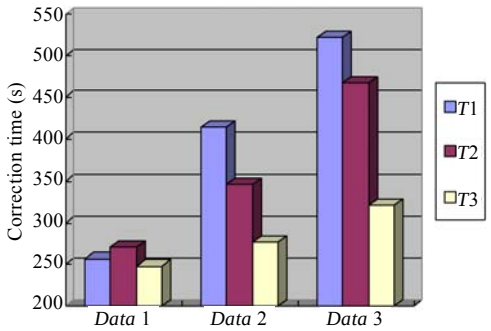


Fig.9 Comparisons of the performance on efficiency of three correction techniques on three data sets

图 9 比较 3 种纠错方法在 3 个数据集上的效率

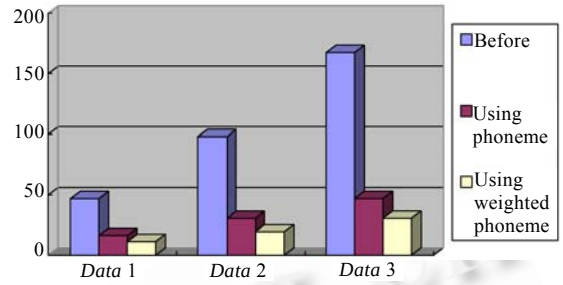


Fig.10 Comparisons of error counts before correction, after a correction by using phoneme and after a error correction by using weighted phoneme

图 10 纠错前的错误数、采用音素的语音纠错后的错误数以及采用加权音素语音纠错后的错误数比较

本文提出的纠错方法并不能保证纠正所有的错误,如果存在以下情况,纠错就可能失败:a) 某个字符的手写识别结果中未包含正确结果;b) 字符的手写识别结果候选在发音和字形上都很相似;c) 第 3.1 节中介绍的“过切分”存在错误;d) 语音识别严重出错(为了提高对语音识别容错度,我们引入了加权音素,然而,严重的语音识别错误仍无法克服).

图 11(*X* 轴表示句子中的字符数,*Y* 轴表示融合时间)给出了融合算法的执行时间.结果表明,该融合算法在处理字数不多的句子时,执行效率良好.我们还在字数更多的句子上实验该融合算法,结果表明,当字符数继续增加时,融合时间显著增加.这表明此算法需要进一步改进才能在长句纠错上具有实际应用价值.

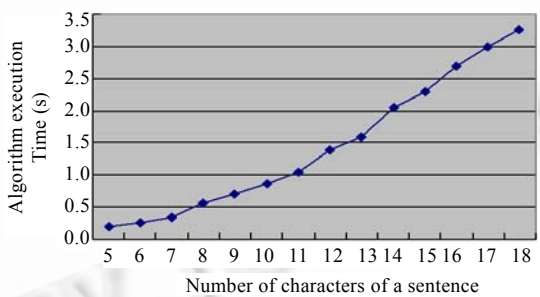


Fig.11 The time to take to for the execution of the fusion algorithm

图 11 融合算法的执行时间

5 结 论

为了自然而高效地纠正连续手写识别中的错误,本文提出了一种利用多通道融合的纠错方法.用户只需要口述所书写的内容,就可以纠正手写识别中的字符提取错和识别错.一个融合语音与手写数据的多通道融合算法是该纠错方法的核心.该融合算法通过利用寻找最匹配用户语音的手写识别结果纠正手写识别错误.实验评估表明,该多通道融合能够有效纠正错误,计算效率也较高.与其他两种纠错方法相比,本文提出的纠错方法效率更高.

在今后的研究中,我们的工作将从以下两个方向扩展.第 1 个方向是对本文方法在实用层面上的比较性评估.由于目前对本文方法的评估主要在算法的有效性和性能两个方面,此项扩展研究能够进一步衡量本文方法

的实用程度.第 2 个扩展研究的方向是考虑在线手写识别中如何利用语音辅助字符的提取和识别.事实上,本文提出的多通道纠错算法已可被看成是结合语音的连续手写识别算法.如何将此算法应用到识别和书写同时进行的在线手写识别中,是一项值得研究的课题.

References:

[1] Suhm B, Myers B, Waibel A. Model-Based and empirical evaluation of multimodal interactive error correction. In: Proc. of the ACM CHI'99. 1999. 584−591.

[2] Karat CM, Halverson C, Horn D, Karat J. Patterns of entry and correction in large vocabulary continuous speech recognition systems. In: Proc. of the ACM CHI'99. ACM Press, 1999. 568−575.

[3] Mankoff J, Abowd G. Error correction techniques for handwriting, speech, and other ambiguous or error prone systems. GVU Technical Report, GIT-GVU-99-18, 1999.

[4] Dong SH, Wang J, Dai GZ. Human-Computer Interaction and Multimodal User Interface. Beijing: Science Press, 1999 (in Chinese).

[5] Wang Y, Yue WN, Wang H, Dong SH. Multi-Modal interaction in handheld mobile computing. Journal of Software, 2005,16(1): 29−36 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/16/29.htm

[6] Oviatt S. Taming recognition errors with a multimodal interface. Communications of the ACM, 2000,43(9):45−51.

[7] Oviatt S. Ten myths of multimodal interaction. Communications of the ACM, 1999,42(11):74−81.

[8] Dupont S, Luettin J. Audio-Visual speech modeling for continuous speech recognition. IEEE Trans. on Multimedia, 2000,2(3): 141−151.

[9] Luettin J. Visual speech and speaker recognition [Ph.D. Thesis]. Department of Computer Science, University of Sheffield, UK, 1997.

[10] Suhm B, Myers B, Waibel A. Multimodal error correction for speech user interfaces. ACM Trans. on Computer-Human Interaction, 2001,8(1):60−98.

[11] Oviatt S, van Gent R. Error resolution during multimodal human-computer interaction. In: Proc. of the 4th Int'l Conf. on Spoken Language Processing. 1996. 204−207.

[12] Mankoff J, Hudson S, Abowd GD. Providing integrated toolkit-level support for ambiguity in recognition-based interfaces. In: Proc. of the ACM CHI 2000. ACM Press, 2000. 368−375.

[13] Sturm J, Boves L. Effective error recovery strategies for multimodal form-filling applications. Speech Communication, 2005,45(3): 289−303.

[14] Halverson C, Horn DB, Karat C, Karat J. The beauty of errors: Patterns of error correction in desktop speech systems. In: Proc. of the INTERACT'99. IOS Press, 1999. 133−140.

[15] Wang XG, Li JF, Ao X, Wang G, Dai GZ. Multimodal error correction for continuous handwriting recognition in pen-based user interfaces. In: Proc. of the IUI. 2006. 324−326.

[16] Tan YK, Sherkat N, Allen T. Error recovery in a blended style eye gaze and speech interface. In: Proc. of the ICMI 2003. ACM Press, 2003. 196−202.

[17] Read JC, MacFarlane S, Casey C. Oops! Silly me! Errors in a handwriting recognition based text entry interface for children. In: Proc. of the 2nd Nordic Conf. on Human-Computer Interaction. ACM Press, 2002. 35−40.

[18] Vo MT, Houghton R, Yang J, Bub U, Meier U, Waibel A, Duchnowski P. Multimodal learning interfaces. In: Proc. of the DARPA Spoken Language Technology Workshop. 1995.

[19] Bolt RA. Put-That-There: Voice and gesture at the graphics interface. Computer Graphics, 1980,14(3):262−270.

[20] Cohen PR, Johnston M, McGee D, Oviatt S, Pittman J, Smith I, Chen L, Clow J. Quickset: Multimodal interaction for distributed applications. In: Proc. of the 5th ACM Int'l Multimedia Conf. ACM Press, 1997. 31−40.

[21] Kaiser EC. Multimodal new vocabulary recognition through speech and handwriting in a whiteboard scheduling application. In: Proc. of the IUI 2005. ACM Press, 2005. 51−58.

[22] Saenko K, Darrell T, Glass J. Articulatory features for robust visual speech recognition. In: Proc. of the ICMI 2004. ACM Press, 2004.152−158.

[23] 2006. http://en.wikipedia.org/wiki/Levenshtein_distance

[24] 2006. http://mathworld.wolfram.com/TribonacciNumber.html

[25] 2006. http://www.hwpen.net/product.htm

[26] 2006. http://www.microsoft.com/speech/download/sdk51/

附中文参考文献:

[4] 董士海,王坚,戴国忠.人机交互和多通道用户界面.北京:科学出版社,1999.

[5] 王悦,岳玮宁,王衡,董士海.手持移动计算中的多通道交互.软件学报,2005,16(1):29−36. http://www.jos.org.cn/1000-9825/16/29.htm

敖翔(1979−),男,重庆人,博士生,主要研究领域为人机交互技术,笔迹结构分析.

戴国忠(1944−),男,研究员,博士生导师,CCF 高级会员,主要研究领域为人机交互技术,实时智能,软件工程.

王绪刚(1976−),男,博士,助理研究员,CCF 高级会员,主要研究领域为人机交互技术,神经网络,机器学习.

王宏安(1963−),男,博士,研究员,博士生导师,主要研究领域为人机交互技术,实时智能,软件工程.