实验二：只对韵母进行右决策时的结果：分裂终止的条件是帧数小于门限 400，此时的模型数目 417,输出分布数 1160

实验三：同时对声母和韵母进行左右决策分裂时的结果：分裂终止的条件为声母帧数小于 100，韵母帧数小于 400。此时模型数 2706 输出分布数 3768

表 4.7 用于孤立词识别实验

实验条件	模型数	分布数	识别率
上下文无关模型	65	257	56.0%(91.1%)
传统声母细化模型	138	549	67%(94.7%)
声母左决策	331	1122	53%(91.1%)
韵母右决策	417	1160	60.6%(93.3%)
全面决策	2706	3768	65.6%(95.2%)

从表 4.7 看到，对声母左决策分裂后比起上下文无关模型来，识别率不但没有提高，反而略有所下降，同对韵母左分裂结果有一定的相似性(表 4.5)。我们认为这同上一章的结论比较一致，也就是左边的音素确实对右边的音素影响比较少。在这情况下，如果门限取得不是合适的话，它由于训练数据的减少反而训练不充分，抵销分类带来的好处，从而降低识别率。同左决策对门限比较敏感相反，右决策我们可以看到总能提高识别率。解决这个问题可以有两个方案，第一对左决策右决策分别设置门限，通过高门限限制左决策的分裂而通过低门限加强右决策分离。这些都说明本方法通过门限设置，它又能退化、进化到任何程度的模型上去。**而利用本方法巨大建模能力的最佳最好途径是增加训练数据。**

4.8 融入声调决策信息

传统的声调识别方法是与音节的识别分离的，即声调是作为音节识别的后处理过程来进行的，这在孤立音节的识别中似乎是非常自然的事情。但在连续语音识别中，这样的处理方法就显得勉为其难了。连续语音识别的中间结果形式千差万别，可以是音节串、音节图、词串或词图等。在这样的中间表示中加入声调的得分无法考虑声调之间的相互协同作用，而忽略声调的相互协同作用其声调的信息就不十分可靠了。其二声调作为汉语语音识别的一个非常重要的知识源，它对指导搜索，特别是增大搜索路径之间得分的差距，从而减少搜索量具有非常重大的意义。其三，在连续语音识别中，基频的提取更加困难。由于这些原因，声调的识别特别是声调信息的融入一直被认为是汉语连续语音识别中的一个难点问题。

上一章我们的声调有关的声学模型研究中，提出了声调识别与音节识别同步完成的一个思路。这种思路的基本思想是在声调的主要载体韵母模型上就按声调进行分类。然后通过加入高阶谱估计或基频及其一些动态特征于特征空间中。在连续语音中，声调的相互协同作用是比音素协同发音现象更为明显的一个

事实,因而在考虑上下文有关的模型时，考虑声调的相互影响是很自然的事情。这样，模型的个数又为原来的 16(4*4) - 25(5*5)倍，如果直接对这些模型进行估算或进行聚类是一件很困难的事情。在基于语音学知识和数据驱动下的决策树分裂的框架下，则能比较好地解决这个问题。

在这个框架下，有两种融入声调识别的方法。一种采用有调韵母作为基本的音素，而把左右韵母的声调作为上下文语境的一部分在控制分裂的问题集中出现，这样就比较明确地区分了不同声调的音节，这正如我们上一章在中小词表识别中采用的方法那样；另一种思路还是采用无调韵母作为基本的音素，而在控制问题集中加入左右韵母和自身韵母的声调，统一作为 Context 的一个问题。这种做法可能把不同声调的韵母分裂开，也可能没有分裂开。由于 COSDIC 库没有达到韵母按声调单独建模的规模，因而我们采用后者这种软判决的方法。即能区分的区分，不能区分的就不区分。下表列出了我们在词识别上的一个初步结果，见表 4.8 。：

表 4.8 按声调分类后的识别结果

实验条件	模型数	分布数	有调首选	无调首选	前 10 选
不考虑声调	2706	3768		65.3 %	95.6%
考虑声调	2569	1934	61.5%	65.8%	95.3%

在这个实验中，我们额外地加入了一个分裂终止条件，即当对数概率增加小于 200 时认为模型已经有足够的精度，不再分裂。从上面的结果可以看到，我们在没有加入基频特征的条件下，有调词的识别准确率可达到 61.5%,而音对声调不对也统计在内的话，即无调识别率达到 65.8%。尤其重要的是，不考虑声调的前 10 名实际词的个数和考虑声调的前 10 名实际包含的词的个数相比已经大不一样，后者要远小于前者。这实际上降低了后面语言处理的困惑度，也说明了声调信息的重要性。

4.9 总结

本章讨论了基于聚类的自下而上的模型建立过程和基于语音学知识和数据驱动相结合的决策分裂树建模过程，尤其着重讨论了后者。获得了一些非常有价值的结果和结论:

。根据汉语语音学的特点设计了包括声调在内的控制完成决策分裂过程所必须的 64 个问题，这些问题或粗或细地为声学模型的逐步优化和细化提供了宏观指导。

。分析研究了基于聚类和基于决策树的两种声学模型建立框架，提出了直接的用于合并和分裂的概率测度准则和语音数据直接融合方案。这些框架和方案在汉语上下文建模中获得了一定的成功。其识别率高于上下文无关的模型识别率，

。研究提出了基于决策树的声调有关声学模型的建立，提出了声调的软判决方法。在框架下，获得的词候选个数减少，降低了语言处理的困惑度。

但我们看到，本方法虽然同上下文无关模型相比识别率有明显提高，但同传

统的 138 模型相比，提高不是很多．我们认为，造成这个原因是因为 COSDIC 数据库的规模用 138 个模型已经足够了．随着语音数据规模的扩大，声调基频特征的加入，我们有理由相信这种框架将发挥更大的优势．

第五章 汉语孤立词搜索算法研究

孤立词的发音方式比较起一字一顿的单音节方式来无疑自然了许多，但是由于汉语中部分分词带有一定的模糊性，并且实际使用过程中用户有可能并不知道某词词汇是否一定在词典中，大量的单字词的使用频度也很高[115]，因而从听写机应用看，这种发音方式比起连续语音来也并非非常符合人类的自然说话习惯，单独用之进行文字录入有很多的不方便．按该发音方式产生文本的速度也要比自然语音来要慢．

但有几个理由需要我们对这个问题进行研究．首先在连续语音中有协同发音引发的减缩、同化、异化等绝大数语音现象在孤立词的发音中都存在，只是在程度上略微有区别．就建立影响某音素发音特性的Triphone模型，则完全可以在孤立词语音中有充分的体现．这样研究了词的上下文有关的声学模型也就基本研究了连续语音中的声学模型的建模问题，只需换相应的数据库即可。由于孤立词的搜索空间相对来说比较小，使得我们有可能设计出搜索算法，使其引起的误差几乎为零，从而把研究的问题单纯化，更能客观地评估反映上下文和多说话人特性的声学模型的精度；其次，正如我们第二章指出的，汉语的词无论在声学层面上的匹配还是语言层面的处理都是至关重要的，脱离词表的搜索是一种比较盲目的搜索．因而在技术上，孤立词表的搜索技术是连续语音识别搜索的基础，也是我们研究搜索算法的起点；另外，在某些应用就只需要大词表孤立词的识别，例如当语音输入与其他输入方式结合时，电话号码的查询(Yellow Page)等。

本章将从语音识别最基本的搜索算法及特点开始，着重探讨利用汉语单音节特点构建的高效二遍启发式算法及一些实验分析结果．这些算法作一定的修正后还可适用于连续语音的识别．

5.1 语音识别的基本搜索算法介绍

在任何HMM框架下的语音识别系统中都采用符合语音特点的由左向右模型，因此搜索也是由左向右展开的，搜索的起点是任何合乎文法的句子中的第一个词的第一个模型的第一状态．评价各种搜索算法优劣的标准是：第一，它所给出的识别结果使产生输入测试语音特征序列的目标概率达到最大，即它是最优的；第二，算法的计算量必须在一定的合理范围内，这样才有可能实时实现．对于第一条准则，在实际的识别中会略有变通．因为可以有多条路径或状态序列产生同样的词或音节序列，绝大多数情况底下次优的状态路径也能产生正确的识别结果。对于算法的计算量来说，我们认为有一个“容忍”限度．我们不能以牺牲

大量的识别精度来谋求实时，同时也不能把需要数十倍甚至数百倍现有计算能力的算法称为合适的算法。此外句子或词中各个音素或音节的分界点不可能事先确定，搜索是在这一前提下进行的。搜索的本质就是找到一个语音的最佳分割。在实际中，我们也可利用一些比较可靠的分割信息来加快搜索过程，也就是在搜索中加入某些边界条件[116]。

目前基本的搜索算法可以归为两类。基于时间同步的宽度优先的算法，或称为时间同步Viterbi-Beam搜索，和非时间同步的基于启发函数的深度优先的算法，或称为A*算法。

5.1.1 时间(帧)同步Viterbi-Beam搜索算法

基于词汇(或字、词单元)连接的连续动态模式识别问题对于语音识别来说是非常重要的，对于这一问题的解决措施皆以动态规划技术为基石。1989年Chin Hui Lee在前人工作基础上，在特定的搜索空间中，按节点间词的连接方式，对连续语音识别的搜索问题进行了深入的研究，发表了对帧同步网络搜索算法的详细研究报告[112]。帧同步网络搜索算法的理论基础大多来自于Ney.H的One-Stage DP算法。帧同步搜索算法首先设计一个FSN(有限状态网)。该FSN由节点和连接节点的弧组成，节点和弧连接代表一定的声学、语音学和语法知识以及相互的作用。语音识别的任务就是在FSN中搜索一定的最优路径。基于Bellman原理，可以使用一个DP过程来得到最优路径，在语音识别搜索过程里，这个动态规划过程即由Viterbi算法代替。

在帧同步搜索算法中，其FSN可以认为是一个双层网。这两层网络具有截然不同的性质。网络的上层起到了一个语法约束的作用，其节点表示语法单元的边界，弧表示语法单元及其转移。网络越简单，表明语法约束越松弛；反之，表明更强的语法约束。网络的下层是对上层网络的弧的展开，正是这样一个双层网络为帧同步搜索算法提供了搜索空间。

在查找局部最优子路径的过程中，大部分子路径的得分非常低。显然，完全保留这些得分较低的子路径在空间和时间上是不现实的，也是不必要的。这提示我们可以在帧同步搜索过程中动态地对路径进行剪枝，抛弃希望不大的子路径，从而大大减少计算量。束搜索(Beam Search)就是这样一种动态局部剪枝的搜索策略。

假如帧同步搜索算法处理到了t时刻，此时对应的子观测序列为：$Y_1 Y_2 \cdots Y_t$，此时该子路径W_i所处的基元模型和基元内部的状态节点分别为λ_t和s_t，则子路径的得分可以定义为：

$$\Pr(W_l) = \prod \Pr ob(Y_i|\lambda_t, s_t, W_l)$$

当前最优子路径表示为:

$$W_M = \arg\max(\Pr(W_l))$$

我们可以通过设定一个门限BEAM，所有得分在$\Pr(W_M)$与BEAM*$\Pr(W_M)$之间的子路径将得到保留，进行下一步扩展，并删除其余子路径．这样BEAM SEARCH将搜索量大大减少(大概只有Beam*输入观测矢量序列长度的数量级).

Beam搜索是一个次最优的搜索算法．如果剪枝过于紧凑的话，在搜索过程中，可能会由于总体最优路径的子路径的某一个状态节点上得分太低，在搜索过程中被抛弃掉，从而，永远不可能发现全局路径最优．所幸的是，在语音识别中，这种次最优的算法往往是成功的[68].

5.1.2 时间不同步A*搜索算法

时间不同步的A*算法对于一个孤立词树的搜索可以描述如下[10]．在算法的每一步迭代过程中，有一个排了序的列表，该列表中每一元素记录着一条局部路径和该条路径的启发式得分．选择得分最高的路径按照词树进行扩展并计算扩展部分的似然得分和重新估算启发函数得分，把似然得分和启发函数得分相加，即为该条路径的评估函数，并按照评估函数得分高低插入列表中．评估函数描述了该条局部路径扩展完整后的最大可能概率得分，即为整条路径概率的上限估算．当出现在列表头上的路径是一条完整的词时，算法结束．只要搜索过程的启发函数满足一定的条件，以上算法保证能找出最优的路径．并可以非常容易地得到任意个候选。

假设路径f(完整或不完整)在时间t时得分为$\alpha_t(f)$,其未扩展部分的启发函数得分估算为$\beta_t^*(f)$,则路径f的评估函数定义为:

$$h^*(f) = \max_{0\le t\le T} \alpha_t(f) * \beta_t^*(f)$$

搜索的目标是找出一条完整的路径其概率得分h(f)最大．我们说启发函数是可采纳的(Admissible)当且仅当$h(f) \le h^*(f)$,其中$h(f)$为路径f扩展后的任何一条路径的概率，等号当f为完整的路径时成立。当选择的启发函数满足可采纳的条件时，我们能保证识别结果能按照概率从大到小的次序产生。简单证明如下:

在排序的堆栈中，假设一条完整的路径出现在栈顶，则对栈中的任何其它一条路径f',显然有: h(f)>=h*(f'),而对f'的任何一条扩展路径(包括完整路径)f'',皆有h*(f')>=h(f''),所以有h(f)>h(f'')。这个证明过程的思路对我们程序调试也大有好处。

在A*搜索过程中，由于堆栈深度的几何级扩展，我们也需要对堆栈进行裁剪，例如只取前固定个数的局部路径进行进一步的扩展．也可采用Beam Search的办法，取一个门限与最优路径进行比较以决定取舍．

上述算法与帧同步搜索算法的最大区别是它按最优路径扩展，而非所有路径同时扩展．正因为在A*算法中，每条局部路径所经过的时间历程不同，因而需要加入未扩展部分的启发函数才能有可比性．

5.2 利用启发信息的搜索算法

对高效准确的搜索需求随着语音识别任务集的复杂而变得十分迫切．特别是在ARPA计划Wall Street Journal 20000词非特定人连续语音识别任务的推动下，近几年连续语音识别的搜索框架获得了极大的重视．先后出现了N-Best框架[2]，Forward-Backward框架[22]到现在的Multi-Pass搜索框架[28,32]．这些框架的一个基本思想就是逐渐但是充分加入高级的声学模型知识和语言模型知识，利用前一遍的搜索结果来启发加快后一遍的搜索过程．如：在第一遍搜索中，可利用相对简单的上下文无关的声学模型和二元文法，在接下来的搜索中再用复杂的Triphone和三元文法．或在第一遍利用词内上下文有关的声学模型，在接下来的搜索中再用词间上下文有关的声学模型．下面我们对现有的一些算法作一简单介绍．

5.2.1 向前－向后算法(Forward-Backward)

该算法首先由BBN公司的Austin等人提出[22]，它吸收了HMM中Forward-Backward算法中向前向后合一的思想．在普通的Viterbi-Beam搜索中，我们需要根据语法约束扩展新的状态，也需要根据最新的声学匹配和语言处理结果来裁剪状态．状态的激活与裁剪直接影响着识别速度和精度．但由于我们的裁剪只是根据起始时间点到当前点t的得分情况来判断，就有可能出现几种影响识别及对计算不利的影响：第一种情况是某条路径其总体匹配概率很好，概率很大，但由于前面几帧匹配不佳，而后面绝大部分匹配很好，被裁剪；第二种情形是某条路径前面匹配很好，但其扩展部分匹配不好，总体部分得分很底而可以被裁剪，这造成前边计算的浪费．这两种情况都是搜索过程所不愿意看到的．其根源都在于我们不知道时间t以后路径的匹配的大致情况．

Forward-Backward算法正是较好地解决了这些问题．第一遍采用正向的快速的时间同步的匹配，第二遍再进行反向搜索，包括时间序列的反向和搜索空间的反向，并利用第一步的信息．具体算法描述如下：

我们在第一遍正向搜索中记录在时间点t时最后一个HMM状态处于激活的HMM集合为Ω',记录每个HMM最后一个状态的概率为$\alpha(w,t)$.语音结束时间T的前向概率最大值为α^T，即

$$\alpha^T = \max_w \alpha(w,T)$$

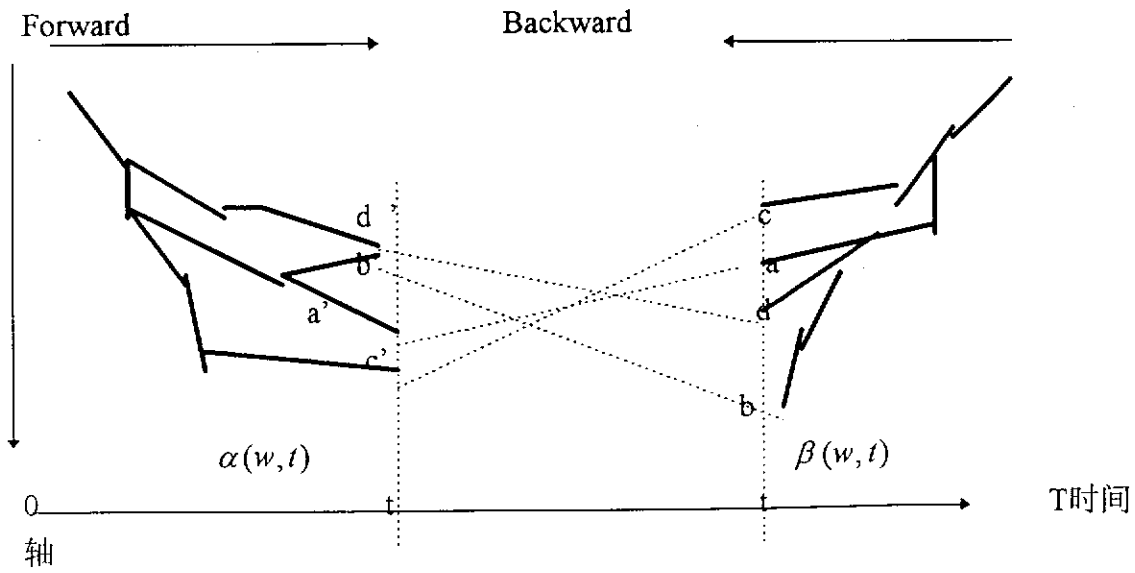
在第二遍反向搜索中，观察在时间t的状态扩展，我们把这些待扩展的HMM w'的概率记为$\beta(w',t)$,则有如下的裁剪策略:

a.如果w'未被激活且不在Ω'中，则说明从起始到t这一段时间中从HMM w'出发的路径概率很小，所以w'可以裁剪掉；本身这个策略因为非常有效的预测了未来匹配的情况，已经可以大大减少了激活的HMM个数;

b. 进一步，如果w'未被激活，但在Ω'中，或者w'处于激活状态，我们可以判决在时间t经过HMMw'的路径是否可以裁剪。此时在时间t经过w'的最大概率为$\alpha(w',t)\beta(w',t)$,所有路径的最大概率为α^T，这样如果数值

$\frac{\alpha(w',t)\beta(w',t)}{\alpha^T}$小于某一设定阈值，该状态w'也可以裁剪调.

从步骤b可以看到，在普通的一遍搜索中，如果w'未被激活，则直接就被裁剪掉了；而在本算法中，还要观察考虑w'以后的概率，只要前段时间概率足够大，则还需保留，同样即使w'被激活，但只要前段时间概率比较小，还有可能被裁剪。在示意图5.1中，d点概率虽然很小，但dd'可能激活；而c虽然很大，但有可能cc'被裁剪。



对接点

图5.1前向－后向搜索示意图

5.2.2 树格算法(Tree-trellis算法)

该算法正如它的名字指出的，把基于A*的树搜索同基于Viterbi的格搜索有机地结合起来了[21,29]。其算法示意图见图5.2。

该算法的第一遍同Forward-Backward算法的第一遍类似。输入的声学观测序列首先在HMM上进行匹配并得到每个状态的似然估计值。然后自左向右地在空间中进行时间同步的Viterbi-Beam搜索。在这遍搜索中，产生每一时间到达每一节点的局部路径的概率值,这一遍搜索称为格搜索。格搜索结束后，后向的A*树搜索开始启动，并按照评估函数挑选最优的路径进行扩展，直至产生N个候选句子送至高层处理。

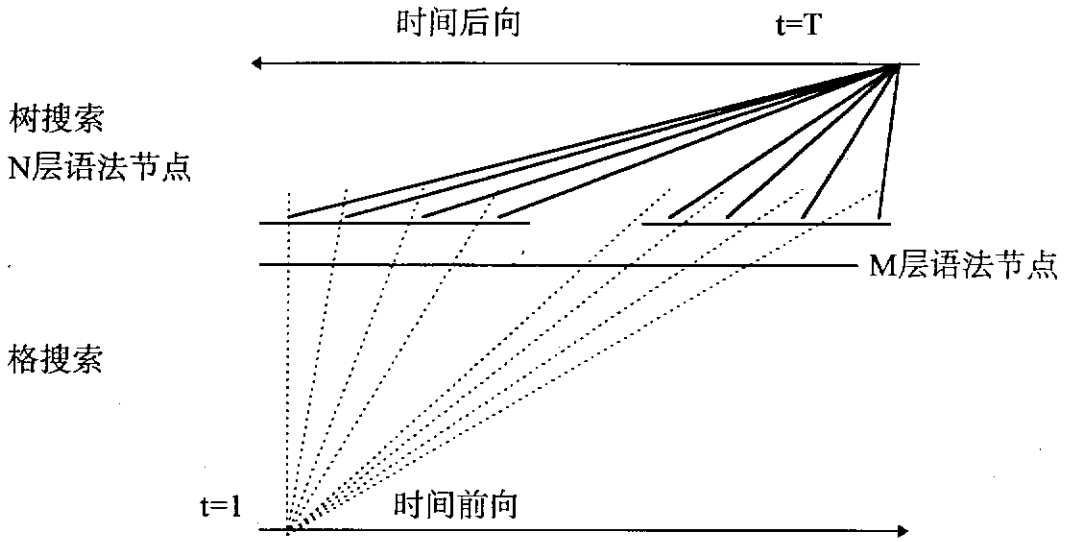


图5.2树格搜索示意图

上述两种算法都利用第一遍的搜索信息来启发第二遍搜索，但树格算法的第二遍搜索不同于Forward-Backward搜索的第二遍。前者用于激活或裁剪状态的扩展，每个状态在同一时间同步扩展；而后者则用之生成启发函数进而形成评估函数来指导扩展，它是按照最优评估函数的路径作深度扩展，其他路径保持不动。我们将在后面对这两种算法的效率进行比较。

5.2.3 多遍搜索机制和A*算法可采纳条件

在上述两种算法的启发下，人们又发展出了多遍搜索算法，其基本思想都是从利用前一遍的搜索结果来加快搜索速度，提高搜索可靠性。但与上面两种算法略有不同。首先它在每一个阶段一般利用不同的声学模型或语言模型，声学模型精度越来越高，语言模型的复杂度越来越大。由于识别的最后结果出现在最后一遍搜索中，前几遍只是提供一些启发信息和中间约束，所以在保证一定包含率的前提下，头一二遍尽可能地用简单的声学模型和简单的语言模型。例可减少HMM的状态数和混合数，甚至一个音素就只用一个状态；可以只用上下文无关的模型或词内上下文有关模型等等；第一遍采用右上下文有关模型，在第二遍采用左上下文有关的模型，最后在第三遍采用真正的Triphone模型；而有些系统在前遍采用只考虑词内部协同发音的声学模型，在后遍再采用考虑词之间协同发音的声学模型。在语言模型的递进使用中：如第一遍不使用任何语言约束或只使用词的Bigram，第二遍使用词的Trigram直至语义、语用知识等等。

第二个不同点是Forward-Backward算法和Tree-trellis算法它们的第一遍搜索结果是以启发信息的形式给出的，而作为多遍搜索算法每一遍的接口，也就是它们的中间结果的表示形式，除了声学得分的启发信息外，一般给出更直接的搜索空间的约束。如N-BEST句子候选，下一遍搜索只要对这N句重新打分即可，又如词格、词图等形式，则直接给出了所有可能的词的连接关系以及词之间的分割点和声学得分等。

从A*算法的角度出发，在同样声学模型条件下，前向可以是一个有较少语法约束的FSN图，而反向则可以是一个有较强约束的FSN图，这种框架保证了A*算法的可采纳条件。因为后者的路径空间始终是前者路径空间的一个子空间，因而后者的最大概率肯定小于等于前者的最大概率。

5.3 基于高效启发式搜索的汉语孤立词识别研究

在对现有的算法充分分析并了解其关键点基础上，我们开始研究汉语孤立词的搜索问题。孤立词识别的其中一条思路可以首先识别出音素序列或音节序列后，再根据词表组词。但正如我们在第二章指出的那样，直接在词表约束下进行声学层面上的匹配能大大减少搜索的困惑度。词表直接来自于93年<<人民日报>>共约32000词。其中绝大数为单字词和双字词。

5.3.1 词库的组织－孤立词搜索空间的构造

我们及国外的研究已经证明树状构造是词识别效率最好的一种组织形式。它充分地把词之间的相同音素进行合并，从而大大减少了匹配量；虽然它在后面的分叉很多，但由于Viterbi-Beam搜索，事实上绝大多数叉被裁剪。另外一个好处是词的这种组织方式能非常容易地将词内的上下文有关的声学模型考虑进去。事实上，我们研究上下文有关的声学模型都是在这个框架下进行的。关于多叉词树的构造算法我们在这里不加以讨论，而只讨论一下词首、词尾和词中间静声的处理问题。

5.3.1.1 只考虑词首、词尾静声情形的处理

语音识别中很重要的一个步骤是端点检测。通常利用能量和过零率两个参数的组合来判断。但是汉语中的轻辅音能量极低，稍微有一点背景噪声就很容易淹没，而过零率更容易受通道零漂和背景噪声的影响。**所以端点的检测不能绝对化，只能作为减少语音数据处理量的一种辅助手段。**这个问题就象识别中的切割问题一样，切割准确了就意味着识别70％的成功，但这本身就同识别一样的难。基于这样的认识，我们在端点检测时不求非常准确，而是采用高阈值，然后在信号的两边前推后移许多帧的办法。通过大量的观测，这样的处理办法能保证所有的信号都在范围之内，无丢失现象，这为后面的精确处理打下了结实的基础。

但是静身的加入需要我们对此进行特殊的处理，而且端点检测的不确定性又表明任何语音首尾可能有静声，也可能没有静声，这就在建模上要求这些静声是可选的。对于起始点静声模型我们放置在词树的根节点上，而对于结束点的静声我们在每一个词的末尾加接一个静声HMM模型。为了表示这些首尾静声是可跳过去的，我们在搜索初始化时把根节点及树第一层的节点都放入堆栈中，如图5.3。而在搜索结束时，所有的树的叶节点及叶节点的前一节点都看作可以合法的结束点，如图5.4所示，经实验比较这样处理后有3-4%的识别率提高。

5.3.1.2.考虑词内音节间静声情况的处理

汉语的组词不同于西文。在英文中，词内部音素之间是没有停顿的，只需要在词间加入可跳跃的静声即可。但汉语中词是由音节拼接而成，而音节之间是可以有一定的停顿。虽然说话的节奏或停顿与分词有一定的联系，但任何一个识别系统不能根据说话过程的停顿来分割词汇，也不能假设词之间有一定的停顿。所以需要我们在构造词树时考虑这个问题。有一些中文识别演示系统就沿用了西

文的一些处理方法，没有考虑音节之间的静声．识别时，只要在词中间作一停顿或者发音速度略慢，识别结果就会有大量的错误．

对这个问题我们有两种处理办法．第一种是对这个问题进行直接处理，就是把一个词根据中间可能有否静声表达成几个可能性．例如对一个两字词来说，分中间有或没有静声就可以派生出两个组合，对一个三字词则可派生出4个组合，四字词派生出8个组合．这种处理办法造成树的空间会急速膨胀，这对词树空间是灾难性的．但在这种处理办法中，每个音节的前后音是确定的，这就很容易处理上下文有关的声学模型．

另外一个办法是间接的办法．就是改变每一个声学模型的拓扑结构，把可以跳跃的静声模型与原模型进行合并，如图5.5所示:

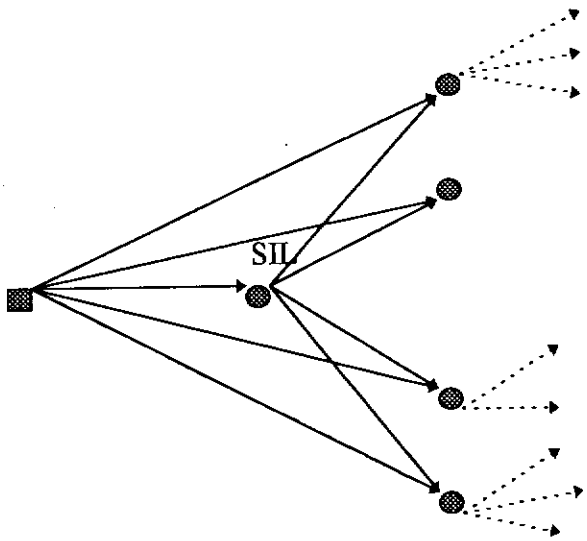


图5.3 前端可选静声处示意

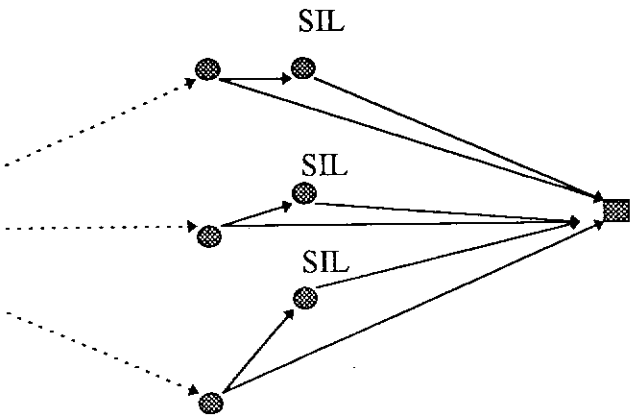


图5.4 尾端可选静声处理示意

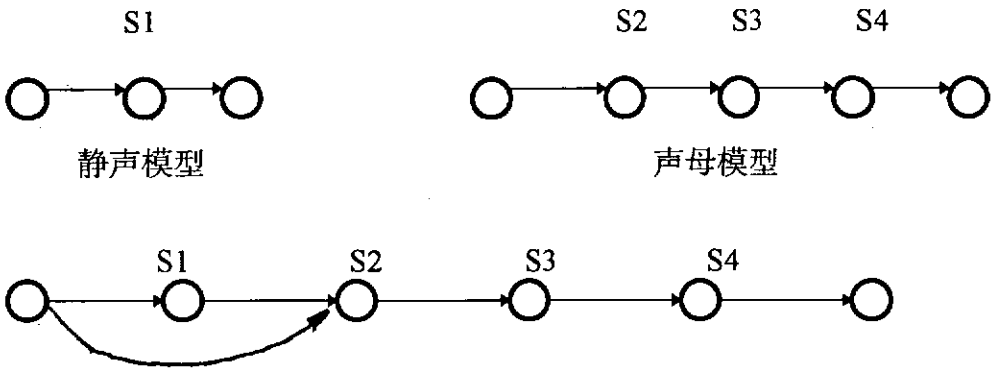


图5.5 模型合并以后的声母模型拓扑结构

这种处理方式不改变树的任何结构，但从上我们也可以看到，这种处理方式不适合于上下文有关的声学模型，尤其是与静声有关的上下文相关模型。

为了研究音节间这种静音模型的影响，在不考虑音节间协同发音的138个模型条件下，我们进行了比较实验．由于没有考虑音节间的上下文有关，所以可以采用上述间接的办法加入噪声．两者相比，识别率提高了约2%。

5.3.2 基于汉语单音节结构的宽紧式两遍搜索机制

我们的BaseLine识别系统从特定人孤立词的单遍搜索开始。模型个数为138个，包括100个基本声母模型，37个韵母模型和一个静声模型。搜索未作程序上的任何优化。训练集为3000左右的词发音，其中绝大多数为二字词，对模型均匀初始化(HMM/VQ方法)，采用Viterbi算法作训练，在486－66机器上实现。识别率及响应速度、错误率分析如下表。

表5.1 搜索时间及识别率

搜索算法	平均识别时间		识别率	搜索引起误差百分比
	一字词	二字词		
一遍正向搜索	6s	15s	72.1%	约25%
一遍反向搜索	30s	60s	71.3%	约25%
二遍F－B搜索	1s	1.9s	78.2%	约2.0%

通常来说，识别错误有两种因素引起。一种是声学模型本身的不够准确，例如声韵母之间内在的混淆，训练数据不够或模型太简单等等；另外一种错误是由搜索引起的，包括从最大似然概率到最大状态序列的近似，搜索裁剪引起的错

误等. 为了考察分析误识的词汇由多大比例有搜索引起，由多大比例由声学模型的不准确引起，我们首先必须对所有错误的原因作仔细的分析.

理论上，要分析这种误差就必须进行全搜索. 但由于我们不可能对所有的词汇进行全匹配，**所以我们作了个近似：对某个识别来说，假设已经识别出的前10名的识别率比词库中所有其它的词的得分高.** 我们对一遍正向搜索的结果进行了分析. 在这个近似下，我们只需观察前10名的情况，若正确结果未在前10名则加入之. 对前10名或前11名进行Viterbi重新打分排序. 结果发现25％的词候选由后几名升到了第一名. **这大致可以说明搜索引起的误差约为25%.**

值得一提的是，一般来说，韵母的识别率比声母要高，所以先识别韵母有利于提高裁剪的可靠性，我们在搜索过程中还尝试了一遍反向搜索，就是时间倒序、词树组织按照倒序产生. 试验结果表明：虽然改进了原来错误的一些结果，但也产生了大量新的错误，主要是复合元音的识别错误，大量介音丢失. 如"uan"被识别成 "an"， "ian" 识别成 "an",识别速度也大大降低. 这个结果说明，虽然韵母总体的混淆度比声母要小得多，但自右向左的这种单向搜索局部韵母的混淆度比声母更大. 比较高的搜索误差和正向反向的这种互补性要求我们采用更好的搜索策略来减少这种误差.

针对上述问题及Forward-Backward原理，根据汉语的单音节特点我们首先设计了二遍搜索思路. 第一遍的搜索空间没有词库约束，也就是说在全音节空间中进行，同时为了加快速度，我们把全音节空间组织成树的形式，静声当作独立的一个音节；这遍搜索并不分割语音，也不产生候选序列，而是产生一些中间启发信息；第二遍搜索我们在词汇空间中进行，同时按照拼接组织成逆序的形式.

在具体实现上需要说明的是，逆序包括三个方面的内容. 首先是**语音序列的倒序**，其次是**HMM模型拓扑结构的的倒序**，最后是**语法连接的倒序**. 在第一遍搜索中，我们记录在某一个时间所有处于最后一个状态的激活的集合为Ω^t，概率为$\alpha(w,t)$，则第二遍算法如下:

```
t=T;
while(t>0)
        t=t-1;
        for each w ActiveWords
                Up date Al lStates Within Hmm();
                X=Get Se tOf Al lBackward Linked Hmms ();
                for each w' in X
                        β (w',t)=Get Hmm Score With Grammar (w,w');
                end
                MaxBeta=Max(β (w',t))
                for each w' in X
                        if(β (w',t)> Prunning Beam width*Max Beta AND
                                w' ∈ Ω^t AND
```

$\alpha(w',t)*\beta(w',t)/\alpha^{T}$>Pruning Beam Width)
begin
Add Hmm To Active Words(w', β (w',t))
end
end
end
end

应用上述算法后，正如表5.1所示，我们可以看到识别的速度及误差都大大降低，其中由于搜索带来的误差降低到了2%,充分说明了该算法的有效性。

针对上述尚存在2％的搜索误差，我们还研究了第一遍搜索结果对第二遍搜索的影响。我们采取两种措施来改善第一遍的搜索：第一个办法是放宽第一遍搜索的裁剪的阈值，第二个措施是将测试集数据加到第一遍模型训练中去(未加到第二遍识别用模型中去)，我们可以看到最后搜索误识率降低的情况如表5.2所示。

从上述结果可以看到，第一遍搜索对前5名的识别率影响较大，而对第一名影响不大。**因而可以肯定本搜索的识别率特别是前5名的识别率主要取决于第一遍搜索，而第一名的识别率只有通过改进第二遍的声学模型才行。**

为此，为保证第一遍搜索的可靠性，特别是进一步提高前五名的识别正确率，我们在第一遍搜索中加入了简单的音节频率信息，以防止高频词的过早裁剪。当然这种知识对我们固定测试集的识别率不会产生较大的影响，但在实际使用场中则会大大提高某些高频词的识别可靠性。

表 5.2 第一二遍搜索关系

第一遍搜索条件	最后误识率降低	
	top-1	top-5
第一遍搜索裁剪阈值从80改为200	5.56%	47.14%
测试数据加入第一遍模型的训练中去	12.15%	61.8%

5.4 基于汉语单音节结构的A*算法二遍搜索机制

为了研究A*算法在语音识别中的应用，我们在上述搜索的基础上实现了利用汉语单音节结构的A*二遍搜索机制，其第一遍搜索与基于Viterbi时间同步的

BEAM搜索相类似，只是在记录点不尽相同。A*算法作孤立词识别，启发函数即为第一遍搜索后的概率记录$\alpha(w,t)$。在本实验中，A*算法可表示如下:

Step1:初始化搜索扩展列表和输出列表。在搜索扩展列表中加入起始节点s,其路径为空，置其评估函数为无穷大。置输出列表为空;

Step2:从搜索扩展列表栈顶中弹出评估函数最大的路径，其部分路径的尾节点设为n

Step3:检查该条路径是否为完整路径，若是输出完整路径至输出列表；如果输出路径总数达到N,算法结束;

Step4:否则对节点n进行扩展：对所有n的后继节点n'

a.计算扩展节点n'的HMM模型声学匹配概率$h_{t+1}^{f}(t')$,其中路径f'=f+n',$h_{t+1}^{f}(t')$表示节点n'从时间t+1到t'的声学概率，则节点n'的似然函数为:

$$\alpha_{t'}(f') = \max_{t<t'} \alpha_t(f')h_{t+1}^{f}(t')$$

b. 假设启发函数为$\beta_t^*(f')$,则路径f'的评估函数为

$$\alpha_{t'}(f') = \max_{1\le t\le T} \alpha_t(f')\beta_t^*(f')$$

c 把路径f'及其评估函数按概率大小插入队列中

Step5:跳到Step2.

此算法实现的关键有二点。第一是关于扩展节点的似然概率估计。就是在第二遍中对一个Phone其起始时间为Ts,结束时间为Te(已经逆序)时的得分，我们在程序实现中用函数CalPointScore(Ts,Te)计算。计算的目的是在不同的起始时间Ts和不同的结束时间Te时对某个Phone的Viterbi得分。因而分成二个循环来计算。首先固定结束时间Te,然后改变起始时间Ts<=Te。为提高计算效率，我们又一次逆着计算。这样一次Viterbi扫描就能计算出当结束时间为Te时不同的起始时间的得分PointScore[tt](1<=t<=te).因为此时本身的时间已经逆序，所以从Te到Ts计算时，模型的拓扑结构不需要逆序。第二关键点是启发函数的寻找，由于在上算法中似然函数的计算已经到了节点n'的末尾，因而启发函数即为在第一遍搜索中记录时间t时节点n'开始状态的概率。

从上述算法的实现也可以看到，每一步状态n的所有后续节点皆被扩展，造成扩展列表以几何级数增加，这大大提高了计算量。那么如何使扩展的状态个数最少呢？我们这儿提出了一种预测扩展的算法。预测扩展的关键是加强启发函数的约束条件，把时间t时节点n'开始状态的概率变为经状态n''结束到状态n'的起始点的概率。此时我们需要对上述算法的Step4进行修改。

Step4:否则对节点n进行扩展：对所有n的后继节点n'中评估函数最好的进行扩展

a.如果n的所有后续节点已经扩展，则可放弃这条局部路径，否则修改该条路径的评估函数，节点n的评估函数为后续尚未扩展节点n'中预测评估函数最大的那个。

b.计算扩展节点n'的HMM模型声学匹配概率$h_{t+1}^{f}(t')$,其中路径f'=f+n', $h_{t+1}^{f}(t')$表示节点n'从时间t+1到t'的声学概率，则节点n'的似然函数为:

$$\alpha_{t'}(f') = \max_{t<t'} \alpha_t(f') h_{t+1}^{f}(t')$$

c.求产生扩展的后续节点n'的评估函数，假设启发函数为$\beta_t^*(f')$,

对n'的所有后续节点计算则路径f'的预测评估函数

$$\alpha_{t'}(f') = \max_{1\le t\le T} \alpha_t(f') \beta_t^*(f')$$

并把其中最大值作为路径f'的评估函数

d.把路径f'及其评估函数按概率大小插入队列中

同其基本算法相比，此时的启发函数必须在时间t时从节点n''结束扩展到n'开始状态时的概率。

我们对上述算法都进行了实验比较，同等条件下，由于A*算法的搜索误差为零，其精度比Viterbi略高，但其速度则明显地要慢得多。下面是实验结果表:

表5.3 各种搜索算法的性能比较

	Viterbi-Beam搜索	基本A*算法	改进A*算法
识别精度	78.2%	79.7%	79.7%
平均相应时间	1.5s	15s	10s

事实上，改进后的A*算法本质上是向前预测了经过一个特定声韵母后路径的启发函数．本算法还可以进行进一步的优化，包括控制展开路径，进行更长距离的预测；对局部路径的时间分割作一定范围的限制(在上述算法中我们假设某条局部路径的时间结束点是不定的，实际上可以根据最大值时的时间点左右按一定的偏差进行限制)等等．但由于在我们的实现中其搜索效率明显不如Viterbi，所以我们未对此算法作进一步的探索．A*算法在声学层面上运用效率不高的根本原因是其搜索的深度太深而引起的。它不广对路径的扩展作假设，还需要对每条局部路径的时间点作假设。另外一个让我们在声学层面放弃A*搜索的理由是二遍搜索的声学模型必须一致，否则不一定能满足可采纳的条件。这也使我们深信Viterbi-Beam搜索确实非常适合于语音识别的需要。

5.5 总结

本章在对各种算法作分析综述后，针对汉语的语音识别的特点，主要完成了以下工作:

。提出了一种高效的启发式搜索算法－宽紧式两遍树状搜索。该搜索算法充分利用了汉语的单音节结构特点，采用音节树的宽松搜索提供启发

信息，而采用词树的紧约束搜索最后结果。该算法保证了算法的实时性，又使得搜索误差几乎为零。

. 研究比较了A*算法在孤立词声学层面的搜索应用．虽然A*算法能保证搜索误差为零，但由于声学层面搜索的深度原因，得出了它不适合直接用于帧层面上的搜索的结论。

. 研究了静音在词首、词间及词尾时的建模问题。

在上述核心算法和第三、四章模型的基础上，加上语言模型搜索后处理，就构成了完整的一个听写机系统，对此我们将在第六章加以描述。对以上算法进行进一步的扩展，就可以应用到连续语音识别中去，对此我们将在第七章中作进一步的详细讨论。

第六章　汉语非特定人听写机系统研究及集成

对于非字符的汉语来说，用于办公室环境的文字听写系统是语音识别极富潜力的应用领域，所以也是语音识别工作者不断努力探索的研究目标。得益于汉语的单音节结构，早期的听写系统采用单音节作为发音单元[108,109,110]。虽然这种方式可以只通过修改词表，不改变声学解码部分而达到无限词汇的识别，但是这种一字一顿的不自然的发音方式妨碍了它的应用，且特定人系统的限制使用户需要经过冗长的注册(Enrollment)训练。从1990年起，我们开始研究面向信息咨询的中词汇量连续语音识别和面向听写机的大词汇量多音节词的识别[117,118]，并在这之后获得科学院“85”重大科研攻关项目“汉语人机语音对话系统工程”的资助，取得了进一步的进展[119,120]。在这些研究工作的基础上，94年我们正式获得“863”计划资助，研究“基于词输入模式的非特定人听写机及其语言模型”，这大大推动和加快了我们的研究工作。不认人的以词为发音方式的听写比较起单字发音来，大大提高了自然度。更重要的是，它无论从技术上还是市场上都可以认为是走向最终完全连续语音听写的关键一步。在“95”年底我们完成了这样的一个完整系统并参加了“863”的正式评测，在识别率及识别速度上都取得了优异的成绩。

构造这样的一个系统涉及许多方面的工作。**总体上，该系统以作者前几章的一些核心模型和核心算法，加上作者在通道归一化处理、快速算法实现、语言模型的后处理搜索以及系统总体设计方面的工作，广泛吸收利用、集成课题组内的一些其它研究成果而成。**本章将着重对系统的预处理、语言模型及处理、通用自适应方法、系统的结构以及实时实现等问题进行讨论[121]。

6.1 面向语音识别鲁棒性的前端技术

在任何语音识别系统的设计中，声学的前端(Front-end)处理的优化是非常重要的，它直接影响到识别器的性能。它包括抗噪音、语音特征参数的选择及变换、与输入通道无关的处理等方面。这些设计应该尽可能地滤除对语音自身无关的信息，例如：讲话者的特性、讲话的方式、背景噪音和信道畸变等；另一方面应该尽量强调那些对音素差异有很大贡献的信息。好的前端处理能去除大量冗余特征，不仅可以减少存储空间，而且可以减少运算量，加快识别速度，提高系统的识别率。

现代语音识别器的一个重要假设就是语音信号是短时平稳的(十几毫秒)。这样前端处理的第一步就是将语音信号分成相互重叠的许多块，再对每一块估算出平滑的谱估计来。在我们的系统中，16K采样率，每帧384点，重叠192点，即每个窗口24ms,移动12ms.象所有的预处理一样，我们对每一块用Hamming窗加以平滑，然后加入一个预加重滤波器来进行高频提升，以补偿由于语音信号平均功率谱受口鼻辐射引起的跌落。预加重系数取为0.97。接下来，我们对每帧数据作

FFT运算，经过一个按MEL分布的24个三角型滤波器获得谱的功率谱分布．该滤波器组在0－1000HZ按线性分布，而在1000HZ-8000HZ内按对数分布．为了使获得的功率谱分布符合我们识别器对此所作的符合Gaussian分布的假设，对此作对数压缩．MFCC特征提取的最后一步是作离散余弦变换．一方面它把24个滤波器的输出压缩成12维的倒谱向量，一方面它也去掉了特征维之间的绝大数的相关性．这使得符合我们对Gaussian分布对角化的模型简化．整个系统的前端处理如见图6.1所示:

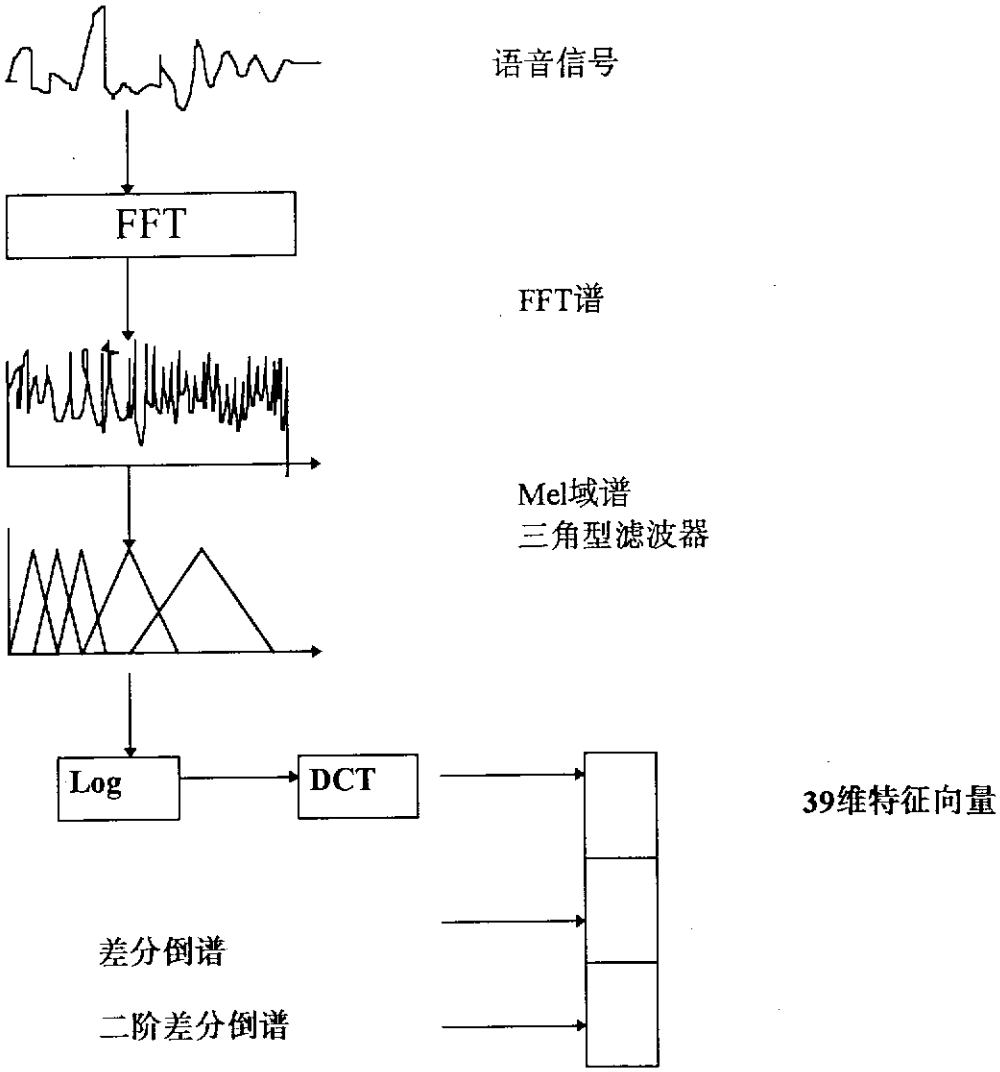


图 6.1 前端处理过程

值得一提的是，上述的MFCC前端处理过程,象Log压缩，DCT变换和差分倒谱,是经过逐年演变后比较符合当今以HMM为声学建模框架的一种优化结果。

6.1.2 LPCC参数和MFCC参数的比较[136,137]

语音信号的频谱既包含有激励源信息(谱的精细结构),又包含有声道系统信息(谱包络). 其中，声道系统信息就是语音识别中要提取的频域特征参数. 目前，国际上一般趋向于以倒谱参数来表征声道特性. 实验已经证明，这种参数作为识别特征是比较理想的. 基于线性预测的倒谱系数(LPCC,Linear Prediction Cepstrum Coefficient) 是比较典型的倒谱系统.

许多语音识别器在低噪音的环境下有好的识别效果. 但是，随着噪声的提高，它们的识别性能急剧下降. 相反，人类在语音信号受到巨大损伤时，仍然能得到正确的识别结果. 因此，适当地引入人耳的一些听觉特性可能会有助于识别率的提高. 根据有关的生理、心理实验，人耳对不同频段信号的敏感程度不同,这就是“美”域. 在“美”域上获得的倒谱为MFCC[122].

我们对MFCC和LPCC两种参数进行了比较实验. 实验分别在高识别率的特定人全音节识别Baseline系统和500词非特定人词识别系统中进行. 从结果来看，MFCC无论在干净语音还是含噪语音中，都较LPCC能够有效地提高识别器的性能(见表6.1，6.2). 特别在特定人全音节识别中，Baseline的识别率已经达到了95%左右，即使在这样的情况下，改用MFCC参数后误识率还有进一步的下降[138].

表 6.1: 采用MFCC参数后误识率变化情况

任务集	特定人全音节	非特定人500词识别
误识率的降低	14.7%	17.8%

表6. 2:不同环境噪音中采用MFCC特征参数后误识率变化情况

SNR(DB)	40	30	20	10
误识率降低	24%	16.6%	18.4%	3.8%

从我们的不同实验中得到MFCC参数是一种比较有效可靠的参数. 目前我们的系统都采用这种参数. 根据高信噪比成份能够提高语音识别鲁棒性的假设,我们还通过对MFCC的模型作进一步的修改，对不同的滤波器输出结果给予了不同的加权系数，可以进一步提高了某些成分的作用.

6.1.2通道归一化技术(Cepstrum Mean Normalization)

另外一个影响识别器性能的因素是通道的变化，包括声音采集卡A/D通道和Microphone的变化。如果我们将通道看成一个线性时不变系统，通道特性给语音信号带来的失真就相当于一个线性滤波器的影响。这样这种失真就可以通过对语音信号的逆滤波(滤波器的参数与失真信号有关)来消除。在倒谱域中，由于倒谱参数是语音信号对数谱的时域表示，通道特性带来的失真在倒谱域值中变为加性。我们就可以通过减去与通道滤波器有关的倒谱值来去除通道失真。如果令C_x, C_y, C_d分别表示原始信号倒谱，观察到的语音信号倒谱和失真倒谱，则重建的语音信号倒谱为:

$$C_x = C_y - C_d$$

估计失真倒谱的方法有两种：长时倒谱平均和短时倒谱平均。在第一种方法中，我们假定失真倒谱在整个观察区间内都是不变的，并且观察时间足够长，以便使该区间内语音信号倒谱的均值主要包含通道特性带来的失真倒谱信息。在第二种方法中，假定通道失真是变化的，但是相对于语音信号是慢变的。这样我们就可以通过合适的移动窗，得到一个时变的倒谱均值，来作为失真倒谱的估值。

我们对特定人同一语音输入通道的几组数据进行了谱平均值的统计。结果绝大多数组数据的谱平均值是非常接近的，因而采用通道归一化后对识别率无明显影响。下表列出了每一组谱的平均值与总库均值的偏差。

表 6.3同一通道下几组数据的长时倒谱平均值

数据组	Sybsmp10	Sybsmp11	Sybsmp12	Sybsmp13	Lexsmp10	Lexsmp11	Lexsmp12
欧氏距离	0.019905	0.009920	0.006522	0.020665	0.007190	0.001677	0.031923

得到通道特性的倒谱估值后，可以用两种方法来实现通道等效。其中一种方法是对训练语料和测试语料分别进行倒谱归一化处理：从每帧训练倒谱矢量中减去训练通道特性作为训练倒谱特征，同时，从每帧测试倒谱矢量中减去测试通道特性作为测试倒谱特征。公式如下:

$$C_y^{'tr} = C_y^{tr} - C_d^{tr} = C_y^{tr} - \overline{C}_y^{tre}$$
$$C_y^{'test} = C_y^{test} - C_d^{test} = C_y^{test} - \overline{C}_y^{test}$$

另一种方法则只对测试语料进行倒谱归一化，将测试倒谱矢量映射到训练通道；即：不改变训练矢量倒谱，同时，即从每帧测试倒谱矢量中减去测试通道特性，又在其上加入训练通道特性，计算公式如下:

$$C_y^{'test} = C_y^{test} - C_d^{test} + C_d^{tr} = C_y^{test} - \overline{C}_y^{test} + \overline{C}_y^{tr}$$

从我们的实验结果来看，后一种方法更加简单有效．我们在非特定人孤立词识别中对该方法进行了实验．训练集我们实验室自行从声霸卡采集，男声采用了动圈式话筒，女声则为驻极体型的；测试数据由科大电子工程系采集，采集卡和Microphone等数据采集通道为未知．归一化数据是通道对所有训练集或测试集的特征参数平均得到．训练集的通道特性:

男声: {-0.249693,0.260663,0.811325,-0.485786,0.325954,0.022071,-0.001769,-0.015259,0.255286,-0.034571,0.116450,-0.246537};

女声: {-1.541558,-0.510920,-0.018272,-1.077893,0.035680,-0.623913, -0.364043,-0.327837,-0.203413,-0.270858,-0.214492,-0.292876};

而测试集的通道特性:

男声: {-1.607771,-0.937487,0.092275,-1.046245,0.144111,-0.573432, -0.425358,-0.158506,-0.319654,-0.193046,-0.419172,-0.402409};

女声: {-0.883132,-0.232120,0.514712,-0.754226,0.058672,-0.399416, -0.399181,-0.095090,0.049479,0.026323,-0.059982,-0.187654};

此时无论是男声还是女声其训练与测试通道特性差距皆大大高于我们在特定人中的通道特性．同时，训练集中男女通道的差异和测试集集中男女通道的差异皆很大．采集训练数据时，男声和女声采用了不同的Microphone,并且在男声数据采集时引入了电源较大的本地噪声，因而其通道特性的差异是比较合理的．而测试集中男女声的通道差异，如果其录制环境是一致的话，只能说明这种长时均值也只能部分部分反映声道的特性，它跟说话人和所发语音还有一定的关系．

但是这种归一化处理还是能部分地补偿训练集和测试集之间的不匹配．经过倒谱归一化后，32000词非特定人的误识率降低了12.5%．我们还对其他类似的通道不匹配情况进行了实验，其误识率的降低程度各有不同．最好的结果来自我们在电话语音识别中的一个实验．两个电话机，一次拨通录制训练和测试数据其识别率为95%以上，而同样的两个电话机重新拨号采集测试集后，识别率则降到65%到70%左右．经过倒谱归一化后，它能降低约80%的误识率．

6.2 HMM/VQ 训练算法及策略比较

在上一章中，我们的实验以及Rabiner的“Toy”模型证明连续密度的HMM对初始化非常敏感．在我们1995年离散密度HMM的实验中，则与此相反．在该实验中，非特定人识别我们将60女声全部用于训练，测试集也采用科大提供的12

人数据库，其中女声6名．无初始化中，我们直接对输出概率赋之为码字的平均分布概率．而在有初始化模型中，我们对部分声韵母进行手工的声韵切分，在模型内部采用均匀分割的方法进行．

同时我们还比较了采用正式的Baum-Welch训练算法和基于Viterbi分割的训练算法．实验结果如下：有初始化比误初始化模型误识率降低8％，见下表；采用最大似然概率训练与Viterbi训练相比误识率降低仅2.64%左右．这说明在HMM/VQ模型中，初始化及训练算法对音节和词的识别不会产生比较大的影响．

表6.4 HMM/VQ 训练策略对比

训练过程的初始化	识别率	训练算法	识别率
随机初始化	59.2％	Viterbi	61.68％
分割数据初始化	62.8％	Forward-Backward	62.72％
误识率降低	8.82％	误识率降低	2.64％

因而在我们的系统中，模型初始化采用简单的均匀初始化．训练也采用简单的Viterbi分割算法，大大加快了训练速度，缩短了实验周期．另外一个好处是避免了训练过程人的人为手工介入

.

6.3 汉语三元语言模型的建立[133]

语言处理在语音识别中具有两个方面的作用．其一是汉语中存在着大量的同音词，平均每个音为15个字，最多的象音yi可以有50多个汉字．这些同音字和同音词需要区分；其二是当前的语音识别的声学模型还不够准确，对非特定人孤立音节来说第一名大约在75%－80%左右，而前几名可能包含率则可达到99%以上，因而语言模型还需要处理更多可能的近音词．它的作用可以说是举足轻重．

6.3.1 汉语语言信息基础加工过[139,140]

针对汉语的特点，语词的定义与切分、词频及词关联概率信息统计和语词切分可以被视为汉语的语言信息的基本加工与收集过程．它是语言信息系统建立的基础性工作。无论何种实用的语言处理系统（诸如机器翻译、智能检索、汉字智能输入及文本校对等）都要利用这些语言信息，处理这些基本的语言问题．图6.2是汉语语言信息基础加工处理的一个简单的流程示意．

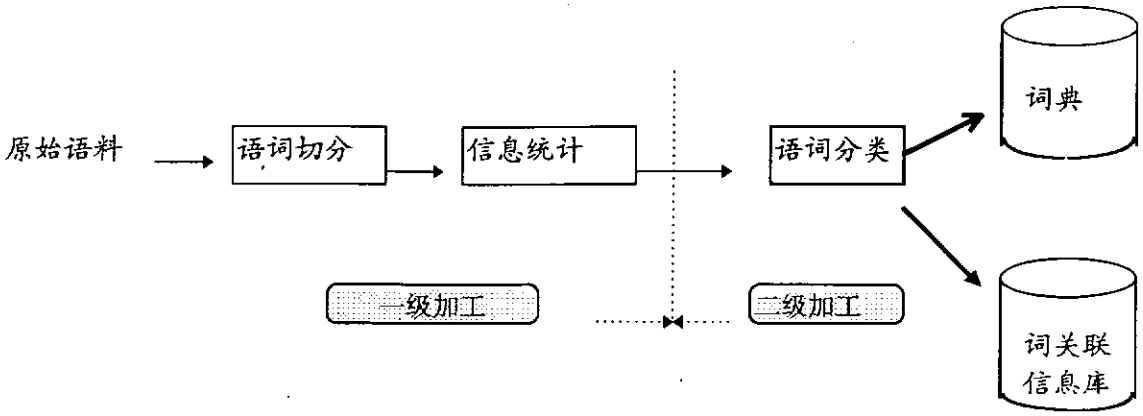


图 6.2 汉语语言信息基础加工过程示意

6.3.2 汉语的词定义与自动分词技术[139,140]

在汉语中，词至今并没有一致的定义，但已有一些汉语的分词规范以及自动分词技术能够被制定和采用．关于汉语的分词规范，国家已制定了《信息处理用现代汉语分词规范（国家标准）．这项标准将词解释为“分词单位”．对它的定义是：**在现代汉语信息处理中使用的、具有确定的语义或语法功能的基本单位．**可以看出，它更强调词的语法意义．在实际应用中，我们参照这一标准，结合语音识别的实际问题，对语词定义进行了一些局部的修改．

汉语自动分词所要解决的主要问题是切分歧义．主要包括交集型歧义和组合型歧义．交集型歧义是在被切分的字段中存在词与词的交错与重叠，而组合型歧义是在被切分字段中存在语词嵌套关系．比如：语句“我们会见面的”．可以有如下几种切分结果：我们 会见 面 的．我们 会见 面的．这是一个交集型歧义的例子．其中词与词之间存在着重叠的成分，难以自动判定其应归属于那个语词．

另外，大多数复合词都存在组合歧义问题．如“研究所”、“不是”、“决不”等．显然，单独使用语词匹配的方法是难以解决这些切分歧义问题的．

对于汉语自动分词技术的研究，国内北航、语言学院、清华大学、山西大学、北师大等许多单位已经对其进行了深入的研究．提出了一些有效的、可行的算法和策略．我们的基本任务是基于大规模真实语料，建立适用于语音识别需要的统计语言模型．所处理的对象是数千万，甚至上亿字次的语料．因此分词系统的设计主要应考虑其实时性和自动性．虽然存在一定量的切分误差，但相对于巨量的统计语料，错误是可以忽略的．同时，我们还可以用一些平滑技术对统计的结果进行平滑．图6.3是我们的自动分词与基础信息词典建立流程框图．

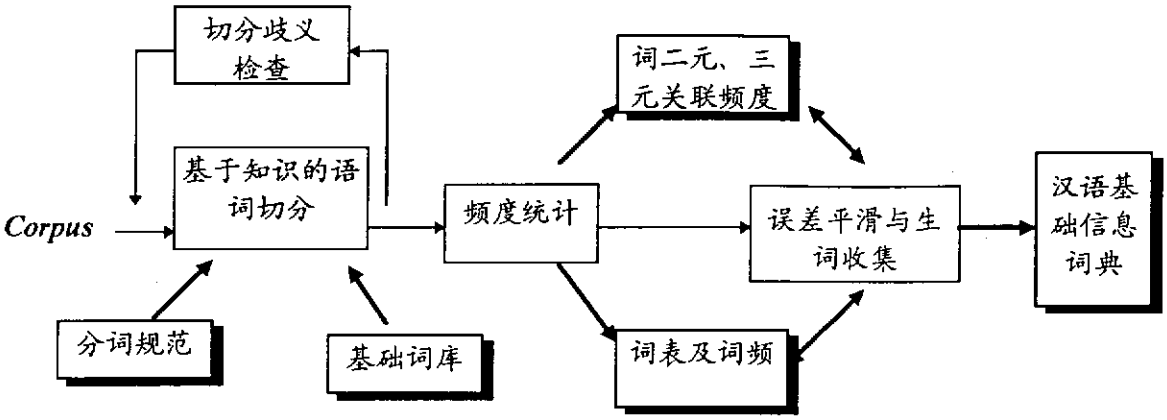


图6.3 汉语基础信息词典建立流程

6.3.3 93年《人民日报》语料的语言信息处理[141]

根据语言模型处理的需要，我们对已经可以利用的电子光盘版93年全年<<人民日报>>的语料依照上面所述的一些方法进行了处理，期望建立一个词的三元统计模型，用于一个完整的汉语听写机系统。

6.3.3.1 语料的切分过程

我们自行设计了一个“汉语语词自动预切分系统”，其目的是为了从大量语料素材中自动建立一个真实的粗加工词典．在此基础上，借助大量的统计数据，自动收集生词，同时对切分过程中因歧义而造成的统计偏差问题利用均衡算法进行自动平滑，对词典进行二次再加工．预切分方法采用双向扫描最大匹配法，结合”语音识别用汉语构词原则”，制定一些构词规则，在切分中进行辅助判断，因而它是一种基于规则的自动分词方法．切分用基础词典的建立也是影响切分质量的一个重要因素．我们采用了滚雪球的方式，即初始化以一个常用词典为基础，对二万字的语料进行预切分，然后依据构词原则，对已切分的语料人工检查调整，一个专门的新词回收机制从人工调整的语料中收集新词，并将其加入分词词典中，重复该过程对更大量的语料进行预切分，选择一定比率的语料进行人工调整使正确率基本恒定．切分系统测试结果见表6.5．

6.3.3.2 语词收集及词频、n元词同现频度统计

在对大量语料进行语词切分的基础上，我们可以动态地从切分语料中收集语词。同时，借助这些已加工过的语料，可以实时地得到一些相关的统计信息。其中，词频、词对及三元词对同现频度是最基础和最容易利用的信息．而且，它们的计算

复杂度和空间占用相对较小,因而较容易实现. 我们在对约二千万字的语料进行自动预切分后,收集到的统计信息见表6.5语料规模1,897万字

表6.5 汉语语词自动预切分系统测试一览表

实际词次	词条数目	二元词对	三元词对
10,135,827	59,040	2,429,930	6,967,226

6.3.3.3 生词收集与统计偏差平滑

在对语料进行了分词及信息统计之后，依据前面所介绍的生词发现与统计偏差平滑方法[]．我们对基础信息词典和词关联频度信息表进行了二次加工处理．共收集新语词243个．并按照信息词典评价方法对处理前和处理后的信息词典和n-gram模型进行进行了比较．结果显示经处理后的词典及模型都有一定程度的改善．表6.6是一些典型的生词收集示例。

表6.6基础统计信息

	词典 PP	二元关联 PP	三元关联 PP
粗加工	6641.82	82.25	76.07
二次处理	6176.49	81.04	75.81

6.3.3.4 音节二元、三元连接概率统计

除了我们建立词一级的语言连接统计概率外，我们还对整个语料库进行标音，统计音节之间的一些连接概率。这包括400个无调音节和1240个有调音节的音频和音联二元、三元模型，这些模型可以大大减低音节一级声学处理的语法困惑度。

6.4 语言模型用于后处理

在大词汇量识别后，我们对每一个发音选择最多至10名的不同发音的候选词，每一个候选词又可能包含几个乃至几十个的同音词，所有这些词的并集构成一个前进单元的词候选，进入语言后处理模块。

对语言模型的搜索也采用了高效的Viterbi-Beam搜索技术。同一般的Viterbi-Beam 搜索不同的是，为了达到边输入语音边输出最后汉字的效果，我们不能在整个搜索完毕后再进行回溯。所以在Viterbi搜索过程中我们直接记录整个路径的词组成和概率信息。若某一前进单元保留句子路径 n 条，后接词候选共m个，则扩展后共有句子路径m*n条，设时间t-1时某一条路径的概率为Ps,当前局部路径的最后两词为w1,w2, 扩展后的词为W3,W3词的声学得分为Pa,则扩展后句子的概率记为:

$$Ps'=Ps+C1*Pu(W3)+C2*Pb(w2,w3)+C3*Pt(w1,w2,w3)+Pa$$

然后再根据m*n个句子的概率得分Ps'进行排序和裁剪进行进一步的扩展。

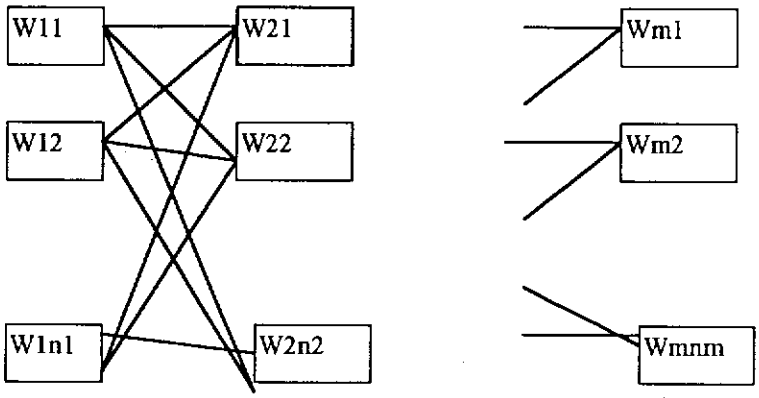


图6.3 词图搜索示意图

这样在界面设计中，我们在识别完l个发音的时候输出这些局部句子候选中的最大值，l+1个词加入后，输出此时的所有概率最大值的句子，由于后面的词会影响前面的词，因而在界面上就形成了语言模型在动态修正识别结果的直观效果。理论上，后面的可以修正所有前面的词，但因为在搜索中有大量的裁剪，因而三元模型的作用范围为前三词，而二元模型的作用范围为前二词。这样我们就可以确定路径的某些词已经固定，那些词可能被后面修改，以便用不同的背景颜色标定。

6.5 基于正则相关的自适应补偿[125]

由于训练数据永远不可能覆盖变化极大的说话人特征差异、声学环境和语音输入通道的差异，因而训练出来的模型一般不能与特定的使用场合吻合。而通过少量与实际使用情形一致的语音数据进行自适应改进识别性能，成为语音识别实用化的最切合实际的技术途径。我们研究提出了一种对说话人，声学环境和语音输入通道皆适用的统一的自适应方法，即基于正则相关分析的谱变换方法[124, 125]。

6.5.1 基于正则相关分析的算法(CCBC)

一个语音样本可以表示成为一个特征矢量序列，测试语音矢量和训练语音矢量分别可以看作是测试特征谱空间和训练特征谱空间中的点。因此，测试集和训练集的频谱特征差异可以通过谱空间的映射变换来进行补偿。不过在我们研究的CCBC方法汇中，并不是直接将1测试语音矢量映射到训练特征谱空间，而是分别将训练语音矢量和测试语音矢量同时映射到第三个谱空间(参考空间)，并使得二个空间中的矢量在第三空间中具有最大的相关性。该方法对数据的变化源未作任何假设，因而该方法可广泛适用于说话人自适应，通道自适用和环境噪音自适应。记 $X^{(1)}, X^{(2)}$分别为测试集和训练集中的P维特征矢量，不妨设:

$$U = AX^{(1)}, V = BX^{(2)}$$

式中A，B分别是对应于$X^{(1)}, X^{(2)}$的变换矩阵，U，V分别是在第三个谱空间中的映射。在保证$E\{U^2\} = E\{V^2\} = 1$的条件下，采用最小均方误差(MMSE)准则，使得:

$$D = E\{(U - V)^2\}$$

达到最小，即可得到U，V的最优估计。该该变换具有以下特性:

1. 在第三空间中，维与维之间的相关性为零
2. U，V之间的相关性达到最大
3. 方差被规一化

应该说这些特性非常符合我们声学建模的原则。说话人自适应的实验证明，如果我们把所有的训练数据和测试数据都映射到这个第三空间中，然后在进行训练和测试，其识别率接近特定人的系统。而在实际的应用中，我们只能通过反变换把测试数据经第三空间再反映射到训练空间中。

6.5.2 一些实验结果比较

为了验证CCBC方法作为通用自适应技术的有效性，我们设计了五中不同类型的测试集与训练集不匹配的情况，包括(1)讲话者不同；(2)讲话者和讲话通道不同；(3)讲话者不同，并且在语音信号中加入了模拟白噪声；4)讲话者和讲话通道不同，并且在语音信号中加入了白噪声；5)不同的讲话者和通道，语音信号在80db商店背景下录制。在不同的实验中，我们还同处理有关不匹配情况的一些算法进行了比较。自适应词表为覆盖所有细化声母的108个词。结果如下：

实验一：说话人自适应：m10和 f10是保留的测试集(由数字录音机录制，经外接放大器、滤波器放大和滤波，TMS320C30卡12位采样得到)。 从表6.7 我们可以看到CCBC的方法比两个空间的直接变换方法(MMSE)误识率要低，比VQ方法略好。

表 6.7 说话人自适应实验

自适应方法	m10	f10
Baseline	9.4	10.0
CCBC	6.2	5.0
MMSE	7.8	6.2
VQ	6.0	5.5

实验二：说话人及通道自适应 :m11,m12,f11在安静办公室环境下，有声霸卡116位采集。 从表6.8可以看到，CMN方法对消除通道影响确实具有一定的作用，但由于CCBC同时作了说话人和通道的自适应，其误识率要比常规的CMN和RASTA方法要低得多。

表 6.8 说话人与通道混合自适应实验

自适应方法	m11	m12	f11
Baseline	22.0	20.2	27.4
CCBC	7.8	8.6	7.2
CMN	13.4	13.2	14.6
RASTA	16.4	14.6	17.6

实验三：说话人及噪声(伪噪声)自适应：m10加入高斯白噪声后的误识率
其中 O: Baseline, +: Lin-Log RASTA, *: CCBC. 从图6.4可以发现，常规的Lin-Log RASTA滤波方法在高信噪比的情况下，反而要起副作用，在信噪比降低情况下，LIN－LOG的假设才得以满足，其误识率逐步降低。而经CCBC处理后，其误识率始终比没有自适应的要低。

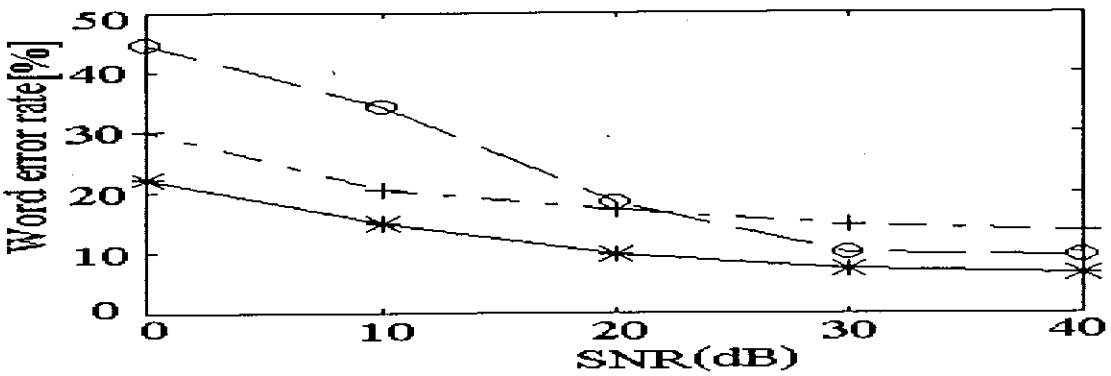


图6.4 m10加入白噪声后的误识率变化对照

实验四. 说话人、通道及噪声(伪)自适应: m11加入白噪声后的识别率情况 O: Baseline, +: Lin-Log RASTA, *: CCBC. 从图6.5可以得出同实验四相似的结论.

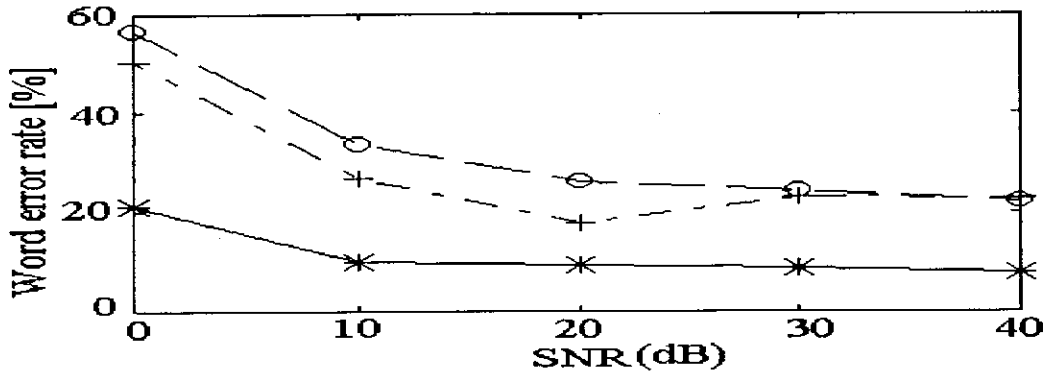


图 6.5 m11加入白噪声后的误识率变化情况

实验五: 说话人、通道及噪声(真实)自适应: m13,m14数据录自80db的环境噪声条件下, 采用声霸卡自带Microphone和声霸卡采样.

自适应方法	m13	m14
Baseline	23.6	33.2
CCBC	14.8	11.2
Lin-log RASTA	15.6	17.8

从上面一些实验结果可以看到, CCBC方法对不同来源的信号失真都有一定程度的补偿作用, 比较起一些针对不同失真源的特殊自适应方法, 其效果还有一定程度的提高. 本方法不但同信号源无关, 而且因为它是在特征空间中作的变换, 所以同采用的识别建模方法也无关. 从这两种意义上讲, 不失为一种通用的自适应方法.

6.6 提高系统运行速度的一些措施

为提高系统的实时响应性，我们对其中的一些模块进行了速度上的优化，主要有以下三点:

6.6.1 实数快速FFT算法

在计算MFC倒谱的过程中需要利用FFT进行语音信号的功率谱的估计．由于FFT运算需要$N\log_2 N$次乘和加，因此，提取MFC倒谱时运算速度的瓶颈就在于FFT计算过程．我们采用了两种方法来提高FFT运算的效率．

- 因为语音信号是实信号，所以我们采用了实数FFT运算来代替常用的复数FFT，具体算法见[126]．实数分裂基FFT(4/2)的算法复杂度为：$(2/3)MN-(19/9)N+3+(1/9)(-1)^M$ 次乘法运算和 $(4/3)MN-(17/9)N+3-(1/9)(-1)^M$次加法运算．与响应的复数分裂基FFT相比较，乘法运算的数目大约减少一半，而加法运算的数目比一半还减少N-2次。
- 根据DFT运算的性质，我们用一次复数FFT运算过程实现两帧语音信号的实序列的FFT运算，其使FFT运算的运算时间减少一半。

6.6.2 快速VQ算法:

在HMM/VQ识别框架中，其后面输出概率的计算只需查表，而矢量量化过程则占相当大的计算量，尤其在我们采用多套码本的情况下．我们知道，设码本的大小为M，每一矢心为一P阶向量，则对某一帧作矢量量化计算欧氏距离共需2*P*M次乘法和2*P*M次减法，并作(M-1)次比较，计算量非常巨大．让我们观察下图:

设Fp与Fn分别表示信号相邻两帧的特征矢量，A为Fp的矢量量化矢心，H为Fn的矢量量化矢心．根据矢量量化的定义以及欧氏距离的三角型不等式，显然有:

$$d_{FnH} \le d_{FnA} \le d_{FpA} + d_{FpFn}$$

若计算P阶距离时出现$d_{FnH} \rangle d_{FpA} + d_{FpFn}$时则必不是我们所求的矢量量化中心。这样我们就有可能勿需计算到P阶时就可推出计算循环，而计算同下一个矢心的距离。

通过对整个数据的大量测试，平均的结束计算从P阶下降低到P/3阶，至少节省计算量一半之多而无任何精度损失。

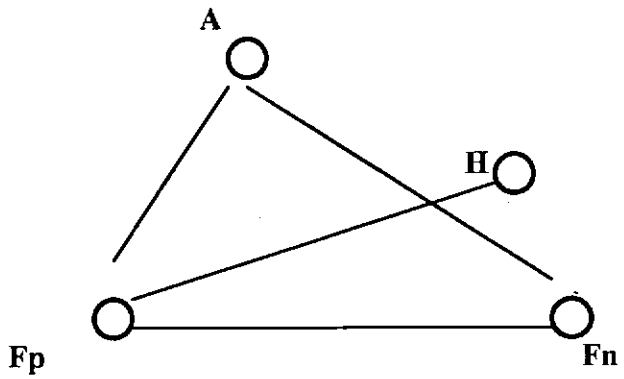


图6.4快速VQ示意图

6.6.3 搜索空间状态合并判断

为了在无任何加速系统的硬件环境底下，达到系统的实时响应，我们对识别算法进行了耗时分解。发现一词平均耗时1.90s,其中第一遍搜索耗时1.69s，第二遍搜索耗时0.21s.而第一遍搜索时主要耗时在于作状态的合并。我们知道，在某一时间达到节点n状态s的路径可能有多条，这些路径必须根据概率的大小进行合并，此时我们需要调用一个专用函数来检查是否已有路径达到该HMM节点中的该状态.由于第一遍搜索在全音节树空间上进行，其节点数及状态数非常有限，因而我们用了一个数组来表示是否有路径达到该节点和状态，若有，则记录该路径在当前状态堆栈中的位置，否则记为无．经上述处理后，搜索时间在486/66机器上就从1.59s搜索时间锐减到0.65s,达到了实时的要求。

6.7 基于Windows95环境下多线程实时听写系统

整个听写系统的流程框图如下所示。但对最后的DEMO系统来说，人们并不关心系统的实现，而系统听写的形式及界面至关重要。

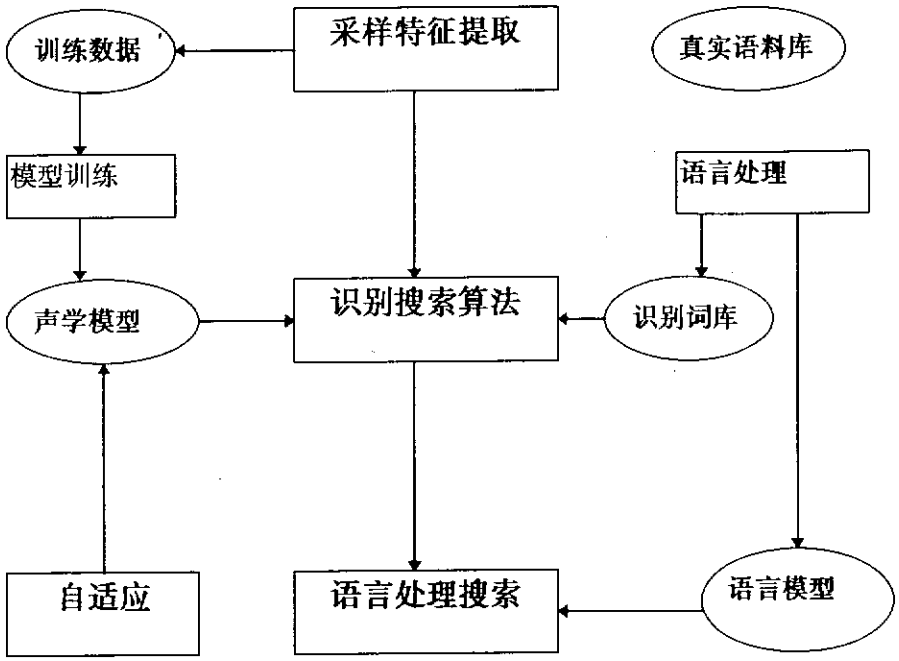


图6.5 识别系统框图

听写系统设计的的一个重要方面就是无等待性性. 包括无等待发音和计算机的缓冲接受. 无等待发音使得用户以快速而又随意的方式发音，也就是可以没有任何停顿的连续语音发音方式; 计算机的缓冲接受在于计算速度不能完全满足识别要求时有一定的缓冲机能. 其中第二个要求需要计算机的语音检测和识别成为两个独立的进程. 在DOS底下类似于语音检测要用中断实现。而在Windows3.1环境底下的驱动程序不能实现上述要求. 我们选择了Microsoft公司最新推出的Windows95系统作为我们的集成环境，系统基本框图如6.6所示。

在这儿语音的检测和特征提取由时间异步的I/O方式实现，数据的下载，端点检测和特征提取在一个回调函数里面实现; 而矢量量化和识别则在令一个线程中运行. 这样在词汇发音之间有一定的间隙条件下，可以以多进程的方式实现数据采集与搜索识别算法并发进行，充分利用机器的资源，提高系统的响应时间. 实际使用时，勿需等待前面识别结果显示，而可以一口气念下去. 这弥补了系统非实时响应的不足，在实际使用时非常重要.

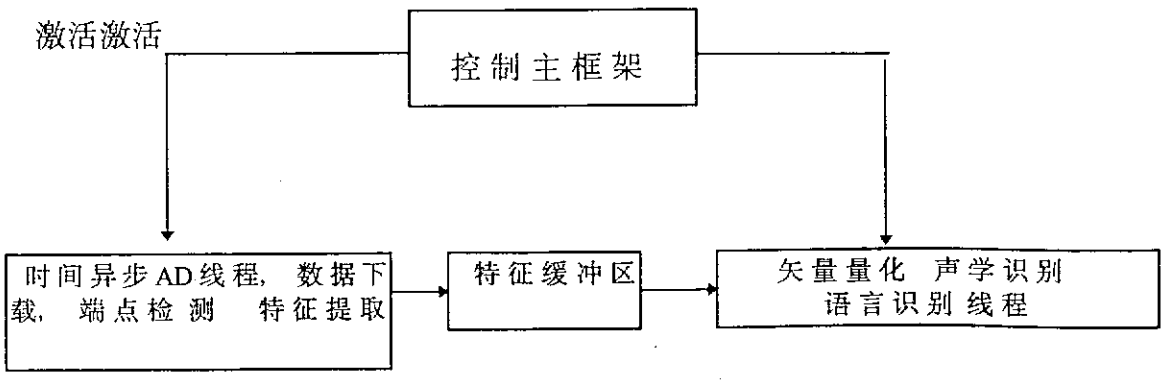


图6.6 多线程示意图

6.8 孤立词特定人实验结果

作为研究搜索算法的平台，我们最初建立了一个特定人系统，用户一般需要发3000词左右的音。实验句子文本来自科大提供的库，共120句，按孤立词方式发音，一个句子发音一个数据文件．实验结果如下:

表 6.7 孤立词音及字识别率

前一名音准确率	前五名音准确率	字正确率
87.5%	98.5%	98.0%

从上面可以看到，即使没有声调信息，通过三元语言模型的处理，也能把正确的词选择出来。在第一名准确率比较高的情形下，字正确率同前五名音的正确率相当，也就是说，只要识别在候选中，语言模型就能把它解模糊出来。

6.9 孤立词非特定人测试结果

采用同样的科大提供的测试语料和测试语音，进行端点检测和孤立词语音分割。声学模型采用了第三章提出的端点有关和音节内部声母右相关模型，共275个．语言 处理部分分别加入二元语言模型和三元语言模型，测试结果如下表6.8：

表 6.8 :非特定人字和词的识别率

语言知识	前1名音准确率	前10名音准确率	字识别率
二元模型	65%	94.2%	83%
三元模型	65%	94.2%	90%

从结果上我们可以发现，语言模型在识别中作用巨大。在前10名识别率达到94％的情况下，字的最后正确率可达90％，也就是说二者只有4－6％的差异。这说明了两个方面的内容：一个是前10名的包含率非常重要，同时我们在实验中，当单独利用Bigram知识时，Trigram知识的应用比只用Bigram知识准确率大大提高。这充分说明了基于词的统计模型特别是三元模型的强大威力，事实上笔者认为，它比任何基于词类的统计模型来得准确的多。

同时我们从大量组的实验发现，在第一名识别率维持在65％左右时，汉字的正确率约为词组前十名的识别率减去4％。而当第一名正确率达到85％左右时，字的正确率则同前10名的词正确率相当。所以对正确的汉字转换来说，第一名正确率和前十名包含率都至关重要。

6.10 系统总结

在本孤立词、非特定人听写系统中，诸多技术的采用和有效集成提高了最后的字的准确率。特别是作者作为系统的主要设计者和集成者，大量采用了本论文的研究成果：

- 采用了音节内上下文有关和端点有关的二种声学模型提高了建模的精确度；这种建模方式特别适合于本系统的宽紧式两遍树状搜索。
- 宽紧式两遍树状搜索机制不但加快了搜索速度，也大大减少了搜索误差，使得系统纯软件运行在标准的多媒体Pentium机上成为可能。
- 采用声学搜索中的Viterbi束搜索作为识别的后处理。这种算法不但能有效地利用词的二元和三元文法知识，而且使得整个系统成为一种无等待的听写模式。
- 通道归一化技术有效地减少了识别系统对输出通道变化的依赖。
- 完成了Windows95环境下语音信号的实时时间异步处理、特征抽取和识别等多线程工作方式。

这些主要技术加上有效的MFCC特征参数，强大的汉语词三元统计模型等使得系统的字正确率达到90％。

第七章　汉语连续语音搜索算法研究

书面语识别的最终目标是大词汇量非特定人连续语音的识别. 连续语音的识别建立在孤立词搜索算法基础之上，但不完全等同于孤立词识别，因为后者是在一个封闭的词库中作理论上的有限搜索，而前者则对词的个数长度都是不知道的. 在第四章讨论的孤立词听写系统中，我们还可以发现，声学匹配和语言搜索是分开的. 而在连续语音识别中，这样做会带来非常大的混淆度. 因而连续语音搜索算法的第二个特点是它必须把语言模型知识和声学模型搜索融合在一起，不能完全分离，这样才能最大限度地发挥语言知识的引导作用，提高识别率. 它可以说是一个语音识别系统的骨架，只有骨架合理，才能最快地得到语音学上和语言学上最合理的结果出来.

一个句子的最优搜索问题可以用一颗无限大的树来描述(而孤立词的搜索则是在有限状态树中进行)，该树中每一节点为一词，从根节点到叶节点为一完整的句子: 语音识别可以看作是树状网络的搜索问题. 当从根节点向叶节点前进时，每一节点的扩展表示了该节点(词)可以后接的节点(词). 每个扩展都带有一个概率，它从观测到的声学特征序列和词之间的语言连接概率共同产生. 识别就是要找到一个从根节点（句子的开始）到叶节点（句子的结束）最有可能的路径（词序列，W)，其概率有节点之间的连接概率（随机统计语言模型P(W). 和声学匹配得分概率(P(O|W))共同概率: 即有:

$$\hat{W} = \arg\max_{W} P(W|O)$$

根据Bayes's定理,

$$P(W|O) = P(O|W)P(W) / P(O),$$

因为在同一语音序列中，P(O)是一样的，所以有

$$W^* = \arg\max_{\{W\}} P(O|W)P(W)$$

这儿O是声学特征序列，W是词序列. 搜索的问题是就是要在现有的声学模型、语言模型和观测序列的条件下，找到具有最大概率的词序列，如果该序列不是正确结果，则该错误是由模型引起，而非搜索引起[127].

在本章中我们首先介绍基于多遍搜索机制的连续语音识别的二个典型框架(N-Best, 词图)；然后我们讨论在m元文法尤其是在二元文法条件下最优序列的搜索问题和在词图基础上后续词三元文法的加入问题. 针对汉语中存在大量同音词问题，我们提出了基于汉语同音词特点的邻词－时间双属性树动态扩展算法. 最后讨论一下我们在非特定人连续语音识别的一些初步结果.

7.1 连续语音识别的典型框架

在口语或听写机系统中，识别面临着巨大的搜索问题．我们要充分利用所有的知识源如语音声学特征，N元统计模型、文法、语义、语用、主题和韵律等知识。一种搜索策略是把所有这些知识都一次性地自顶向下加以利用，在每词处利用所有知识判断后接词的可能性及概率，如果我们能在这样的搜索空间里穷举，我们就能得到最优的句子．但是，由于许多知识源是远距离相关的(特别是基于规则的知识，韵律知识等)，即使采用裁剪或Best-first的搜索策略,这个搜索空间也会非常之巨大，另外，**自顶向下的搜索算法要求所有的知识源都是能预测的，且是自左向右的结构，这大大限制了知识源的类型和利用**．因而对高效搜索的需求变得十分迫切．

在上述任务需求的推动下，研究人员在基本搜索算法(主要有时间同步的Viterbi-beam搜索和人工智能中的深度优先A*搜索算法)的基础上，先后出现了N-Best框架，Forward-Backward和Multi-Pass等用于连续语音识别的搜索框架。这些框架的一个基本思想就是逐渐但是充分加入高级的声学模型知识和语言模型知识，利用前一遍的搜索结果来启发加快后一遍的搜索过程这就是递进式多遍搜索解决方案．这种方案的本质就是把知识源按照适当的顺序加以利用，使得我们在允许的搜索误差范围内，牺牲知识源提供的熵来减少计算的复杂度和降低计算的消耗。有多种针对上述问题的搜索框架及其变种，但总体上从中间产生的搜索结果看可以分为两类。第一类一般在第一遍采用最有效的但又需较少计算量的知识源来搜索出N个句子[2]，而第二类则会用之产生一个中间词图作为下一遍搜索的基础或语法[4,5,6]．而N-Best的句子本身也可从词图中产生，因而事实上词图框架可以认为是N-Best算法的一个中间结果，它们的关系如下图。

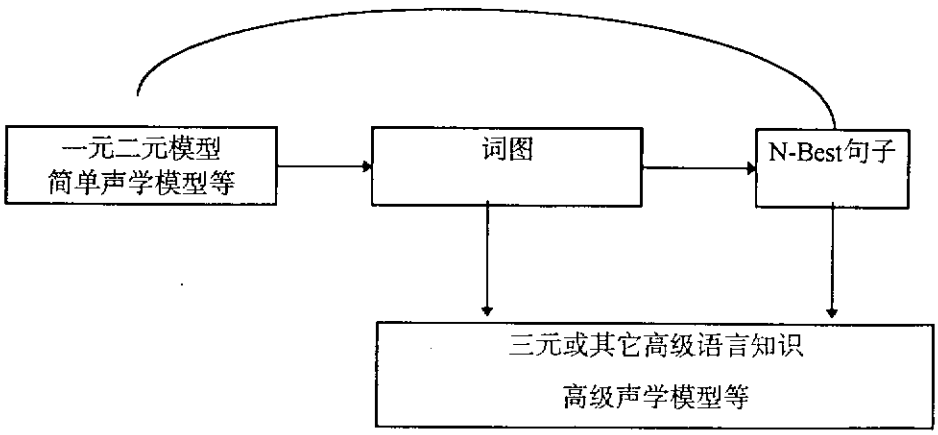


图7.1 N－BEST与词图的关系及连续语音识别框架

下面我们将简单介绍一下N－BEST框架和词图框架，

7.1.1 N－Best框架

常规的N-Best框架指的是在一定的知识源条件下，产生N个句子候选，然后再利用更加高级的语言或语音知识进行重新排序的过程．从其产生N个句子候选的算法来讲，它可以是时间同步的Viterbi－Beam算法，也可以是Tree-trellisA*算法或其它的近似算法，即使采用同样的算法，它们在具体实现和所要计算达到最优的目标也有所不同．

如在文献[2]指出，严格的N-Best算法需要计算词序列的概率而非状态序列的概率，而且它必须保证能找出所有在约束条件内的所有序列．在这情况下，它必须在每个时间点对所有的路径区分前面所有的词．当两条或两条以上的路径到达某个状态时，检查是否已经存在同样的路径，如果有，我们把它们的概率相加，如果没有，则创建一条新的路径．当对某一状态的所有路径创建后，在该状态比较所有的路径并作裁剪，使得到达该状态的路径个数控制在一定个数内．由于所有不同的词序列当作不同的路径保存，因而容易看出所有到达句子结束的词序列其得分是精确的得分．当然，因为可能的词序列呈指数扩展，算法需要在状态一级(局部)和词序列一级(全局)作两层裁剪，前者的裁剪门限比后者要小．一个有趣的问题是能否证明所有大于门限的词序列是否会在N个最后句子候选中，并有正确的得分．这个问题的证明依赖于实际的裁剪门限，详见[3]．

用串(W1,W2,...,Wn,Wn+1,N,S)表示一个局部路径，路径的前面词为W1,W2,...,Wn，现处在Wn+1词中，其节点号为N，状态号为S，只有所有这些项皆相同时的路径才能预以合并；然后对所有到达状态S的路径比较其概率大小，只保留一定个数的路径；然后我们同时对所有的路径进行比较，裁剪某些大大小于全局最优的路径．

上述意义上N个最佳候选句子的搜索计算量是巨大的．因而在文献[4]中又作了一定的简化，提出了Word-dependent N-Best算法和Word-Lattice N－Best算法．Word-dependent算法只要(Wn,Wn+1,N,S)相同，即可合并路径；而Word-Lattice N-Best算法更简单，只要(Wn+1,N,S)相同即可合并．实际上，Word-Lattice N-Bests算法它不考虑前面的词是什么．笔者在硕士论文中采用了与此类似的算法来作没有词库约束的孤立词识别[95]．实际使用的N-Best算法还包括从最佳词序列到最佳状态序列的简化．

7.1.2 词图框架(Word-Lattice)

N－Best 框架无疑是非常吸引人的，因为只有有限的N个句子的重新排序意味着我们能非常容易地加入任何的知识，包括句子全局范围内的语用、语意和节奏韵律等超音段信息．但是当句子长度很长时，N即使取很大的值也往往难以

覆盖正确的句子[7]. 很容易近似地估算，一个即使只有10个词的句子，即使每个词只有3个候选就能覆盖100％的准确率，就需要组合出近60000条句子.

N－Best算法另一个弱点在于在还未充分利用知识的条件下武断地选出最后的N个句子出来，如果我们能为后续搜索提供更加大的余地，则最后的识别可靠性就会好得多，短语图或词图(Word-Graph)就是一种比较好的形式. 一个短语图或词图可以定义为一个有向图，有向图的每个连接表示一个词或短语，每个节点表示一个时间点。有向图只有一个起始节点，时间为0,只有一个终止节点，时间为整个语音帧长. 从起始节点经有向图后到达终止节点构成一条完整的句子. 每个有向图的边还给出在所定时间区间内该词的概率得分.

上述短语图或词图为后续语音处理提供了一个非常简练和有效的语法约束，而它所包含的可能的句子路径则大大超过了N-Best算法所能提供的数目。在该框架底下，通过在词图或短语图中加入更加高级的语言学知识，可大大降低由于Top-N不包含带来的缺漏. 在统计语言模型下，对现有搜索算法作一定修正后，就能产生短语图（在二元模型条件下，即为词图）. 下面我们将用统一的观点来阐述它们的概念.

7.2 m 阶语言模型下短语图生成算法描述

统计模型已经被证明是一种最简单却又最有效的语言模型，已被广泛地应用到语音识别系统中. 如果在我们的搜索过程中，采用m阶的统计语言模型 $P(u_m|u_1^{m-1})$,则我们勿需对整个路径进行记录，只需记录后(m－1)个词及当前词Wn+1,当前节点N和当前状态S，并对记录完全相同的路径进行合并，此步从m元统计模型最优的角度出发没有简化，也未损失任何精度. 但因为此时记录的局部路径已经不是完整的一个句子的局部路径，路径的合并就需要我们记录一些回溯信息，记录的单位是(m-1)个词的组合，这就形成了所谓的短语图(Phrase Graph). 同孤立词识别一样，我们对词序列的搜索首先近似化成最佳状态序列的搜索. 为了描述算法，我们定义下述变量:

$h(w;\tau;t)=\Pr(x_{\tau+1}^{t}|w)$ =词w产生特征量$x_{\tau+1}\ldots x_t$的概率值

$G(w_1^n;t)=\Pr(w_1^n)\cdot\Pr(x_1^t|w_1^n)$=词串$w_1\ldots w_n$产生$x_1\ldots x_t$，时间结束点为t，其中包含了语言概率。

采用上述定义，我们就能够 把某一特定的词序列的声学得分和语言得分加以分开，这个分解可从下面表示出来:

$x_1,\cdots,x_\tau,x_{\tau+1},\cdots,x_t,x_{t+1},\cdots,x_T$

$G(w_1^{n-1};\tau)h(w_n;\tau,t)\cdots$

采用m阶语言模型后，我们可以把词序列中最后(m-1)词相同的句子序列在短语级予以合并，　也就是说我们记录的路径只需区分后接（m-1)个词u_2^m．我们把相应的概率记为$H(u_2^m;t)$:

$$H(u_2^m;t)=\max_{w_1^n}[\Pr(w_1^n)\Pr(x_1^t|w_1^n):w_{n-m+2}^n=u_2^m]=\text{产生声学特征矢量}x_1\cdots x_t$$

并且其词串的最后(m-1)个为u_2^m,时间结束点为t;该式实际上定义了句子即路径的合并过程．采用上述定义后，我们可以把词层次上的动态规划搜索过程写为:

$$H(u_2^m;t)=\max_{u_1}[p(u_m|u_1^{m-1})H(u_1^{m-1};\tau(t;u_1^m))h(u_m;\tau(t;u_1^m),t)]$$

这儿我们用函数$\tau(t;u_1^m)$来定义词　u_{m-1}与词u_m的分割点．其本身可以定义为一个最大化过程:

$$\tau(t;u_1^m)=\arg\max_t[H(u_1^{m-1};\tau)h(u_m;\tau,t)]$$

上述这两个式子给出了短语图中最主要的两个参数(概率及时间分割点)的计算描述．在m阶语言模型条件下，对于词U_{m-1}与词U_m之间的分割点，我们只需关心前m-1词的历史．

这一点是理解算法的关键，需要对这个问题进行进一步的说明．我们说U_{m-1}与词U_m之间的分割点只依赖于前m-1词的历史，并非表明对于两个词串:　$w_1w_2\ldots w_iu_1u_2\ldots u_{m-1}u_m$　和$w'_1w_2'\ldots w_j'u_1u_2\ldots u_{m-1}u_m$　如果其时间结束点皆是t,那么u_m与u_{m-1}之间的最佳分割点一样．而是在我们搜索N个最佳候选过程中，这两条路径在以后的时间点中，它们对应路径的概率只有固定的差异．换句话说，若路径$w_1w_2\ldots w_iu_1u_2\ldots u_{m-1}u_m$扩展成一个固定的句子，概率为P，则$w'_1w_2'\ldots w_j'u_1u_2\ldots u_{m-1}u_m$也可扩展成后面完全一样的句子,其概率为$P'=P-\Delta$。其中，若第一、二条路径在t时跃迁到$u_m$时的概率分别为:
$H(w_i,u_1,\ldots,u_{m-1})$和$H(w_j',u_1,\ldots,u_{m-1})$,则

$$\Delta=H(w_i,u_1,\ldots,u_{m-1})\text{-}H(w_j',u_1,\ldots,u_{m-1})$$

并且必定可以在该短语图中，这两条路径通过回溯最后可以得到。正是从这种意义上，我们作这样的假设是合理的和完全精确的，也没有作任何近似。显然，如果不考虑裁剪带来的因素，满足m阶语言模型条件下的前N名最佳状态序列必定在该算法生成的图中。

7.3 在二元统计语言模型下词图的描述及其搜索算法

当我们在这一遍搜索中只考虑二元语言模型约束时，作为7.2的特例，某词的起始时间可以假设只同前一词有关，即有:

$$\tau(t;u_1^m)=\text{const}(u_1^{m-2})$$

该结论对二元模型是准确的，但对于高阶统计或一般通用的语言模性来说，则称为词对近似(Word-Pair Approximation)[3][4]. 文献[3]中将用此近似方法得到的N-best句子成为Word-Dependent N-Best算法.

在该条件下，我们给出词图的一般描述:

(1)在每一时间点t,我们考虑所有词对$u_{m-1}^m=(v,w)$,采用束搜索，我们只考虑那些最有可能的词对.

(2)对每一三元对(t;v,w),我们保留如下信息:

a.词对的时间分割点$\tau(t;v,w)$

b.词w的得分$h(w;t(t;v,w),t)$

7.3.1 邻词有关词树动态扩展搜索

在上面的论述中，我们尚未牵涉到词对的时间分割点$\tau(t;v,w)$的计算，也未对词库的组织形式作说明。原则上，它可以用Two-level Building算法或者One-pass算法。因为路径必须根据前词加以区分。在词库组织成树状的情形下，严格二元概率连接条件下的最优搜索称之为"邻词有关词法树搜索".

假设 $Q_v=(t,s;w)$ = 邻词v,现词w,在时间t时处在状态s的概率,

$B_v(t,s;w)$=该路径中词w的起始时间

对$Q_v=(t,s;w)$采用动态规划（DP）有下式

$$Q_v(t,s;w)=\max_{\partial}[q(x_t,s|\partial;w)Q_v(t-1,\partial;w)]$$

其中$q(x_t,s|\partial,w)$为HMM中跃迁和输入概率的乘积，$B_v(t,s;w)$根据DP结果不断传播，通过选取合适的初始值$\partial=0$,我们就能找出最好的分割点。在s=S(w)时，我们有

$$H(w;t)=\max_{\partial}[p(w|v\}Q_v(t,S(w);w)]$$

这儿我们已将二元连接概率加入到里面。根据上式，我们对从词V过来的连接词w有如下的初始化:

$$Q_v(t-1,0;w) = H(v,t-1)$$
$$B_v(t-1,0;w) = t-1$$

把这个一遍搜索的邻词属性有关的算法推广到词图的构造中，只需加入时间分割点和词得分的计算:

$$\tau(t;v,w) = B_v(t,S(w),w)$$
$$h(w;\tau,t) := Q_v(t,S(w);w)/H(v;\tau)$$

这个算法区别于一般的Single-best算法在于，后者只需记录一个最优的词对及其分割点，而词图算法则需记录所有的词对及其分割点，并对所有的v与固定的w连接路径合并成一条向下传播（当然不同的w需要启动不同的路径）。这样就能保证我们搜索出二元概率连接意义上的最优句子来。由此我们可以推断，在某一时间时，必须保留一串词的候选向下扩展，而对一元概率连接最优来说，则只需合并成一个最优词即可扩展即可。

对上述Word-pair的进一步简化就是就是[3]中的Word-Lattice N-Best算法。

7.3.2 时间有关词树动态扩展搜索

在7.3.1论述中，我们对树的扩展是根据前面的词来进行的，该算法称之为"邻词有关词法树搜索"，相应的生成的词图实际上是词与词之间的连接，在后续的搜索中就以这种已经生成的连接作为接下来搜索语法。[5]中提出了另外一种树扩展算法，称为Word Hypotheses Generator(WHG)。其WHG产生过程不一定要采用词的二元连接概率知识，它是根据每个时间点来扩展树，所以其在每个时间点产生一颗树。也就是扩展的词根据起始时间来标定，而不是用前面的词来标定。这样生成的词图就比较简单，其每个节点可以用四个参量来描述: Word ID=Leaf WordID,Start Time=Tree Start Time,End Time=Current Time,Score=Leaf Score,每时间点把叶节点中最大概率往下传题，合并则根据(S,T)进行,其中S为树中节点编号，T为该树起始时间，所以实际上不同的时间标设的词树之间没有进行任何合并，每一个时间树是独立进行的。但文章提出了根据全局最大值进行裁剪的方法，树起始点的时间也可以进行一些限定(例如每隔三个时间点产生一颗新树)，这样可以使生成的树的个数变得稀疏一些。在这种定义下，（Wp,S,T)等同于(S,T)。因为知道了时间也就知道了该时间点所有结束的词。通过上述策略，该WHG产生算法其计算量等同于二元概率连接产生 Best-Sentence 的搜索量。

这种算法的实质是词与词之间只需要时间上相连就是一条合法的路径。显然其后续搜索量比邻词有关扩展要大，同时这种算法不能保证二元连接概率意义上的N－Best结果最优。

7.3.3 基于汉语同音词特点的邻词－时间双属性树动态扩展

让我们来分析一下上面两个算法的特点及缺陷。从二元连接概率意义上要产生N-Best的角度看，我们要采用“词有关的词法树搜索”。但在每个时间点上，为了保证词的包含率，我们需要取尽量多的词候选，而由于汉语中存在大量的同音词，如果完全按照词来扩展，则在每个时间点扩展的树个数会相当大；另一方面，从同一时间扩展的那些词来看，在到达下一个词的叶节点前，它们的得分的增值又是一样的，因而我们完全可以按照概率最大的一个词扩展，在到达叶节点时，再单独取前面所有词与之匹配，因而纯粹“词属性的词法树搜索”是很不经济的一个算法。但该算法则可以在不同起始点的词树之间进行合并，只要保证它们的邻词一样即可。

“时间有关词法树扩展”则与上面不同。它在每一时间点扩展一颗树，最大限度地减少了搜索的冗余度。但它每个时间点之间是独立的，每个时间点又按最大概率一条路径扩展下去，因而无法达到Bigram上的最优，尤其无法保证前N名候选。

基于上述两个算法的互补性，这儿提出了基于汉语同音词特点的“词－时间双属性词法树扩展”搜索算法。

在该算法中，搜索扩展堆栈中每个扩展状态标设为(t_p, N, S),其中t_p表示进入词树的时间，N为词树的节点编号，S为节点中HMM的状态，每个扩展状态还有一个独立的指针，指向在时间t_p－1时所有可能结束的词标号。假设在时间点t_p－1产生候选词$w_1 w_2 \ldots w_n$,其概率分别为$p_1 p_2 \ldots p_n$,最大词$w_{\max}$的概率为$p_{\max}$,到时间点t时其增加概率记为Δp，此时从标设(t_p, N, S)指针所指向的所有词的概率为$p_1 + \Delta p, p_2 + \Delta p, \ldots, p_n + \Delta p$,对于时间t时扩展堆栈中所有标设我们按照以下规则进行状态的合并，合并包括标设所指的词汇的删除和扩展堆栈中标设的删除:

a.比较标设中节点号N和状态号S相同的扩展项，观测其各自所指的词汇表，若有词汇相同，删除概率较小的，只保留概率最大的项;

b.若某个扩展项所有所指向的词汇皆被删除，则删除该扩展项。

c.比较所有扩展项所指词汇的概率大小，设定一门限后，删除概率较小的词汇。

d.按照扩展项扩展路径，同时拷贝其所指的词汇

这个算法是严格的二元概率连接意义上的N-Best算法，同时充分利用了时间有关的词法树搜索的特点。它主流上以时间属性进行扩展，合并了同一时间结束点所有词扩展的计算冗余，另一方面又通过该时间点所有词的索引在全局范围内根据词属性进行合并，并根据合并结果删除某些时间点。它在精度上保证与“词属性词法树扩展一致”,而在速度上大致与”时间属性词法树扩展相当“，保证识别精度的前提底下加快了搜索速度。

7.4 非特定人连续语音识别初步实验

连续语音的识别同孤立词的识别有联系，也有不同．其区别主要在于在词树搜索中，孤立词只要到了叶节点搜索就结束了，而连续语音则需要不断地作树的动态扩展．我们在第五章中已经讨论了孤立词的宽松式两遍搜索机制，也在前节中针对连续语音问题讨论了词树的几种动态扩展方法．我们的连续语音识别就是基于上述的这样框架．第一遍搜索同孤立词的第一遍搜索一样，在第二遍搜索中引入了词的二元概率，并采用邻词－时间双属性树扩展方法产生词图．一个简单的加入词的二元概率连接知识搜索出后的词图7.2所示：在每个节点表示一个时间分割，每个跃迁表示一个词及其在前一遍搜索后的得分概率．我们希望在该图的约束范围内搜索出一条在三元连接概率条件下的最优路径．

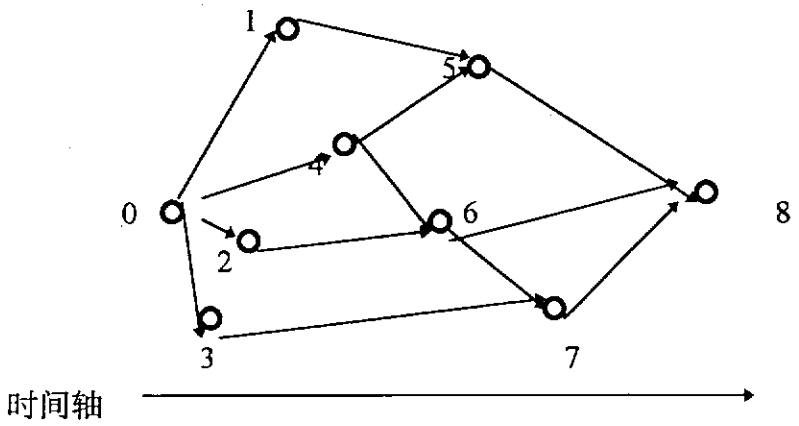


图7.2 词图示意

此时的搜索算法不能沿用孤立词时的语言模型搜索算法，因为此时每前进一步词的占用时间不一样，无法作同一测度下的概率比较；每条路径所经过的词汇个数也不相同．一种做法是把概率归一化成每一时间帧的值，把语言模型的概率归一化成单个词的概率．如下式：

$$p_{avg} = \alpha * \frac{p_a}{t} + (1-\alpha) * \frac{p_l}{n}$$

但实验结果表明这种测度并不非常理想，尤其是对语言模型概率得分按照词数平均造成路径间的得分差异减少，影响裁剪的判决．针对这个问题，我们设计了在词图上搜索的A*启发式算法．

在产生词图的搜索中，我们已经把每个跃迁的概率按路径进行累加，包括声学概率得分和语言概率得分，并在路径的结合点记录到达该节点的最大概率，并把最大概率往下传递．这些节点记录的概率构成了语言层面上启发式搜索的基础。

正如第四章所描述的一样，基本的A*算法扩展所有连接当前节点的后续节点，如上图所示，此时第一步就扩展路径8－>5,8->6,8->7；具有预测功能的A*的算法则只扩展P5,P6,P7中最大的一个，并修改节点8的启发函数为剩下来两个概率中的最大值；依次类推．采用后者可大大加快搜索速度．

同声学匹配中A*算法的低效率相比，在词层次上路径的寻找，A*算法效率要高得多，更重要的是它能确保搜寻出的路径概率是最大的．在前向中，概率为一元、二元概率的加权累加值，而在反向中，概率则为一元、二元和三元联接的概率累加值，因而前向概率Pf始终大于后向概率Pb，满足可采纳的条件．

对非特定人连续语音识别进行了初步的实验．模型采用连续密度的HMM，共测试女声60句子，语音测试库来自科大95年底“863”测试句子．训练集为孤立词的发音，即COSDIC库中的所有语音．系统共138个模型．最后字的准确率及识别时间如下:

表7.1 连续语音识别初步结果

最后字正确率	平均每句耗时
70.2%	20s

在实验中，我们发现高频句子的正确率相当的高，而某些偏僻的句子的准确率要低得多．说明系统对语言模型的依赖性较大，而声学匹配的准确率还不够高．这一方面是由于该模型的训练来自孤立词的库，如果改成连续语音库作训练后相信识别率会进一步提高；另一方面是138个模型对连续语音识别来说是不够的．这期待我们不断地加入在声学模型研究中的一些结果，不断更新．

对上述识别结果，我们还分析了其准确句子的音节准确率和所有识别句子的音节准确率．结果如下表:

表7.2 连续语音音节识别准确率分析:

音节识别率	Top-1	Top-5	Top-15
所有句子统计	48.43%	79.85%	95.47%
正确句子统计	70%	90%	100%

从中可以大致看到，在连续语音中音节的准确率如果能达到70%左右的水平，而前10名大致达到100％，则其最后的结果将是相当理想的.

7.5 总结

在本章中作者首先分析了现有一些连续语音识别的框架，并指出了它们之间的相互关系。对m元尤其是二元统计模型条件下搜索的最优化问题进行阐述，并介绍了两种动态树扩展方法。在此基础上，作者完成了以下工作:

. 提出了一种针对汉语同音词特点的邻词－时间双属性动态扩展树算法，这种算法有效地减少了计算的冗余度，提高了搜索效率.

. 在提出的孤立词和连续语音搜索算法基础上，实现了一个基于Viterbi和A*混合搜索的连续语音三遍搜索识别框架。并利用现有模型，对汉语非特定人连续语音进行了完整的测试，达到了70%的字准确率.

. 分析了连续语音的初步识别结果，对音节识别率进行了统计. 得出了在连续语音中音节识别准确率的大致奋斗目标. 这对以后的工作将产生非常有益的指导.

总结和结论

汉语非特定人、大词汇量听写系统的建立任务，工作量浩瀚。其中包括硬软件环境选择、建立，通用声卡数据采集工具编制及实际录制，特征提取、优化及鲁棒性预处理，声学模型研究，训练算法研究，多种搜索框架设计及实现乃至最后的语言模型使用等方方面面。作者论文期间在每一方面几乎都有所涉及，当然有时也难免会浮光掠影，不够深入。但作为博士论文的内容，是不可能面面俱到的。因而突出了几个重点，而这几点作者恰恰认为也是我们国内语音识别研究比较少的、与国际水平有一定距离的。

对于语音识别中一些通用算法和具体实现如 Forward-Backward 训练算法和它的 Scaling,输出状态捆绑以后的参数估算，时间同步的 Viterbi 解码算法，特别是涉及到搜索算法等，需要非常高的实现技巧，某些技巧甚至对识别效率乃至识别率都会产生举足轻重的影响。这虽然花费了作者非常大的精力，但为了保持论文的清晰，未作详细论述。

另外本论文研究时间跨度较大，其间无论是课题任务、实验用的数据库乃至识别方法等时有变迁，因而各实验条件及结果会略有差异，但论文试图反映出我们研究工作的思路和得出的一些行之有效的结论和方法。

主要工作及贡献

概括地，本论文的主要工作可以归纳如下:

1. 根据汉语的特点，从音节信息熵的角度分析了各种不同知识源条件下的音节困惑度，从理论上阐述了汉语语音的识别无论从语音层面还是从语言层面考虑词汇表约束的重要性，从而为孤立词和连续语音的识别提供了有价值的指导。
2. 指出了词典的树状表示和动态扩展是一种高效和可行的数据组织形式；在树状结构下，为提高语言知识的利用效率研究了语言概率的分布式表示，实现了汉语音节的一元和二元连接概率的分布式表示，在连续语音中提高了音节的识别准确率和包含率。
3. 从不同角度研究比较了声韵母和音素这两种典型的汉语语音建模单元。从可训练性、一致性特别是最后识别率的分析得出了声韵母是一种比较理想的汉语语音基本建模单元的结论。
4. 在孤立字、词(特定人和非特定人)的平台上全面深入地研究了基于上下文有关的声学模型的建立。这些模型包括音节内上下文有关模型，音节间上下文有关模型，声调有关声学模型及端点有关声学模型。指出了在汉语语音识别中除了考虑音节内上下文有关模型以外，必须考虑音节间的协同发音现象和声调对发音的影响。
5. 研究分析了语音信号变体的不同来源。明确提出了两种提高建模精度的途径即模型分类法和提高模型的自身描述能力，阐述了它们各自不同的使用特

点，从而明确了建立上下文有关模型的重要性。

6. 全面分析了利用参数共享及平滑概念，建立精确而又鲁棒的声学模型的两个方法－即基于全数据驱动的自下到上的聚类算法和基于语音学规则和数据驱动相结合的自顶到下的决策树算法，并分别比较了这两者的优缺点。根据汉语语音学特点，提出了一套用于决策的 64 个控制分裂问题集，并首次将决策树方法用于汉语语音的声学模型建模。
7. 提出了在连续语音中融合声调信息的模型建立方法，把韵母本体和其左右声调作为上下文信息建立了同声调有关的声学模型，一定程度上解决了汉语声调识别与音识别的同步问题，避免了声调的硬判决。
8. 在孤立词搜索算法上，结合汉语特点提出了宽紧式高效启发式树状搜索算法，该算法几乎使孤立词识别的搜索误差降为零，同时提高了识别速度。在大量实际问题中，这个算法还可用于小词表识别系统中的集外词的拒识。
9. 基于本论文核心模型和搜索算法，优化吸收听写机系统所需的其它各模块包括前端处理、自适应、语言模型与利用、实时实现等，建立了一个基于 Windows95 下多线程的无等待的孤立词听写系统。该系统在正式评测中其速度和精度成绩优异。
10. 提出了一种基于汉语同音词特点的邻词－时间双属性词树动态扩展方法。该算法加上孤立词的第一遍宽搜索和词图上的 A*搜索，构成了连续语音三遍搜索的框架。初步实验取得了比较满意的结果，为进一步发展连续语音的识别打下了基础。

下一步工作

由于基频提取技术及平滑方法的限制，本论文未将该信息加入特征维中。可以相信，基频及其差分信息的可靠加入将进一步改进本论文提出的声调软判决框架。

基于语音学知识以及数据驱动的决策树声学模型建立过程，其效果同设计的控制问题集以及语音数据库的规模有着直接的关系。我们将在更大的数据库上进行实验，期望取得更明显的效果。

另外声学模型的改变，特别是音节间上下文有关模型的建立大大提高了连续语音识别搜索的难度，提出了新的挑战。如何在连续语音识别中按照我们上述的思路建立新的模型特别是有效地同搜索算法相结合是我们下一步工作的重点。