# 语音识别

## 基础

### 分帧：

一段语音往往持续数秒到几个小时。但是人说一句话的时候，一个词持续的时间往往比较短，仅仅有零点秒，词由音素组成。因此每个音素持续的时间更短。

分帧的目标是：获取短时平稳特征。在这个很短的时间内，音素是一个音素，而不能包含多个音素。一般设置为25ms

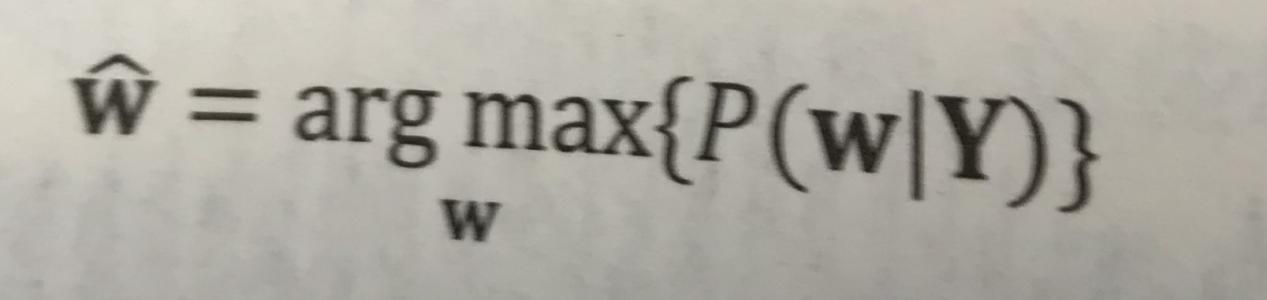
### 帧移：

音素总有会转移的时候，因此总有一个25ms的帧会包含从一个音素到另外一个音素的转移。所以每隔10ms取一帧。

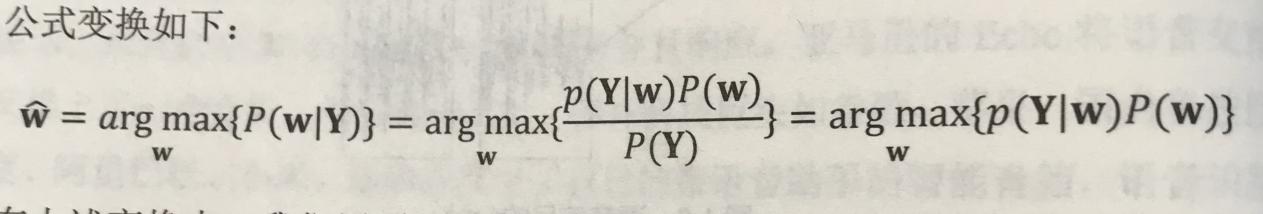
### 目标函数

Y是输入的音频信号

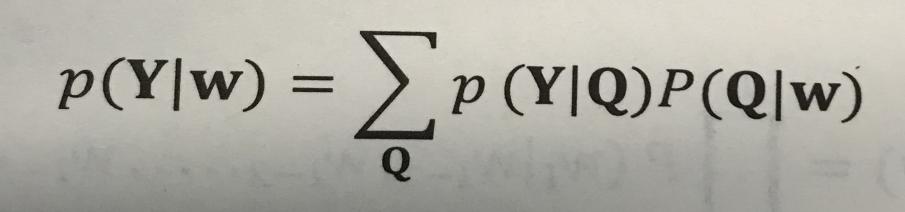
w是输出的词序列



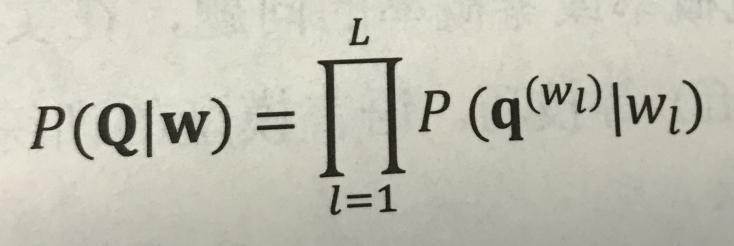
使用贝叶斯公式



w是一系列的词，词由音素序列Q构成



单词wl发音为q(wl)的序列的概率



## 单音子模型

一个音素的实际发音，与左右相邻或相近的音素无关。缺点是：这个假设不符合实际，实际发音往往会有协同发音的现象

## 三音子模型

speak 这个英文单词在词典中对应的发音序列为 [s p i: k]，而第二个音素[p]的发音因为其左临音素[s]而发生浊化，实际应被读为[b]；

get down 发音序列为 [g e t d au n],实际发音时，[t]因为其后续音素为[d] 所以产生吞音，出现不发音的情况。

oh my god. [d]因为处于句子末尾而被吞音

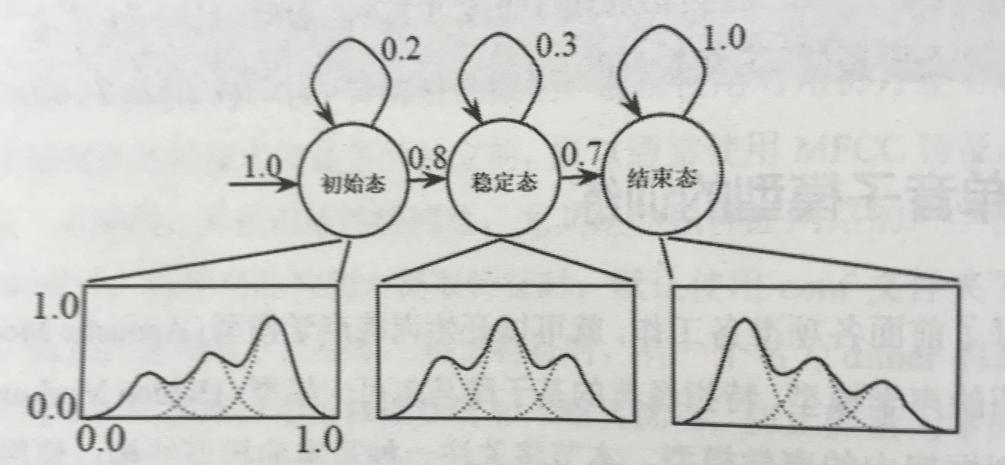
## hmm三状态

一个单音素，或者一个三音子，就是一个hmm。一个hmm一般会设3个隐状态，但是也有会设2个隐状态和5个隐状态的，但是常用的就是这3个隐状态。

这三个隐状态如何理解：

按照传统的hmm来说，观测序列已知，来做解码的时候，往往使用维特比算法，求得最优的隐状态序列。对语音来说输入的是帧序列，而一个音素往往是由多个帧才能组合完成。因此就有了某些帧被解码为初始隐状态，有些帧被解码为稳定态隐状态，有些是结束态隐状态。

初始态、稳定态、结束态 分别对应一个混合高斯模型，混合高斯模型里面的超参数可以自定义



第t帧语音特征ot在第i个状态si上的声学得分

## IMG_1829

## IMG_1830

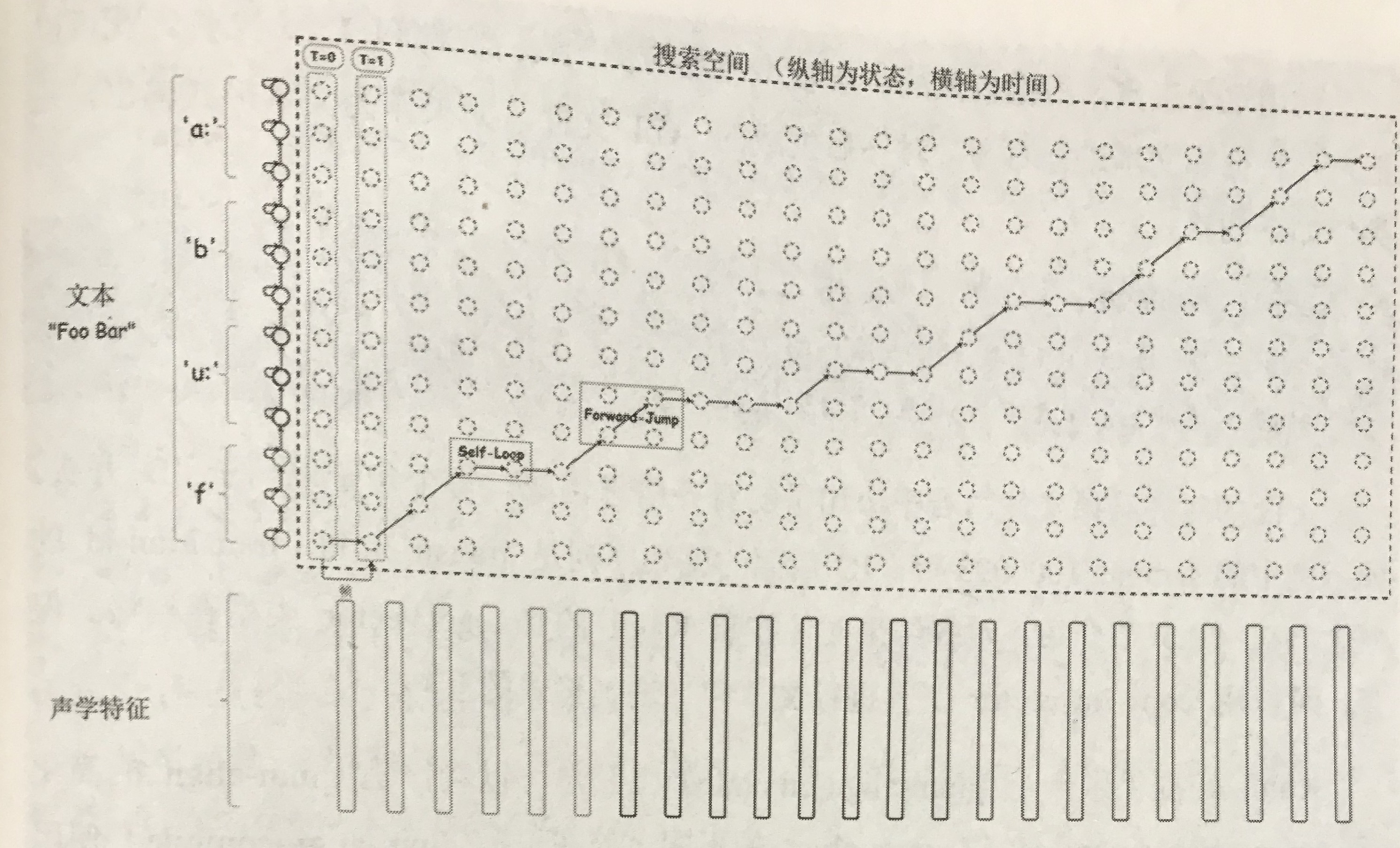
为了降低模型参数量，通常协方差矩阵为对角阵

## IMG_1831

## 单音子gmm-hmm解码

思考：假如训练好了一个gmm-hmm模型来做语音识别，最终的识别结果应该是什么样子的？

如下图所示：



输入的是每一帧的特征，最终的解码是一条路径。

每一个音素都是由一个三个状态的hmm模型表示，每一帧都会被解码成某个音素对应的某个隐状态。

self-loop:一个hmm中的三个隐状态，自己往自己跳转。

例子：

o1 o2 o3帧序列被解码为：

f-稳定态 f-稳定态 f-稳定态

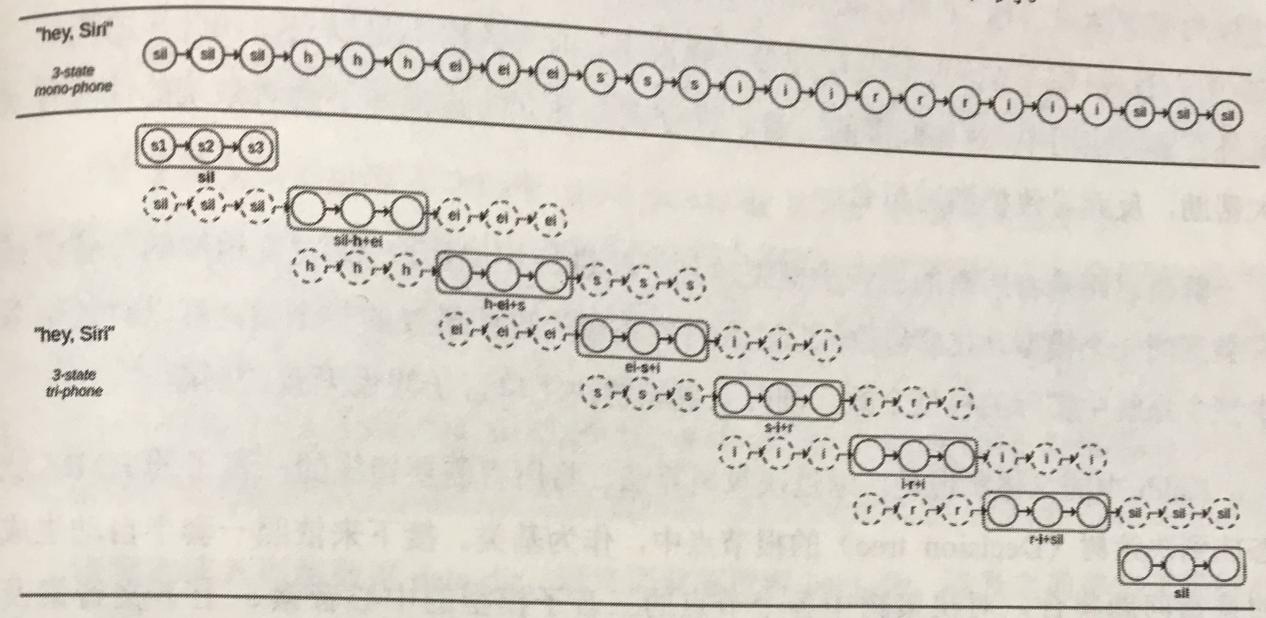
这就是self-loop

forward-loop:由一个音素的结束态转移到另外一个音素的开始态

## 三音子gmm-hmm解码

相对于单音子，多音子更符合现实中的特点

在解码的时候仅仅是把三音子当成单音子就可以了，没有其他区别



### 参数爆炸问题：

对英文来说，有40个单音子，如果做成3音子，则会产生40\*40\*40=64000个三音子。模型训练起来速度慢，并且需要大量的数据。因此会采用对三音子进行聚类，同一类的三音子共享参数。

## gmm-hmm训练

传统的gmm-hmm训练通过EM算法求解

而kaldi里面的gmm-hmm采用维特比训练进行训练（仅仅是提速了，效果没有打折扣，待更新）

## 三音子聚类裁剪

一个三音子hmm模型可以表示为： {L}-{C}+{R}.{S}

其中L: 三音子模型的左上文

C: 三音子模型的中音素

R: 三音子模型的右下文

S: hmm模型的三个状态 {S}={1,2,3}

40\*40\*40\*3个hmm状态需要建模，需要构建这么多个高斯混合模型。40\*40\*40个hmm模型需要建模。

## 解码器