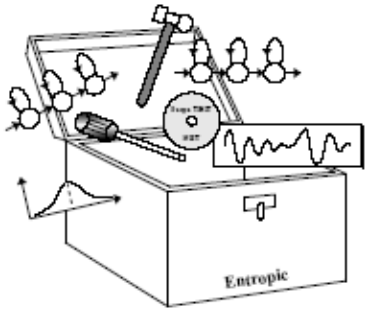
第二章　HTK 工具包概述



前一章节我们介绍了基于 HMM 识别的基本原理。已经涉及了 HTK 中的一些关键的工具。这一章节将要描述 HTK 工具的软件结构。介绍 HTK 工具的大纲以及如何将其结合起来构造和测试一个基于 HMM 的识别器。为了方便使用过 HTK 的读者，我们列举了最近版本的主要变化。下一章节通过建立一个简单的连续语音识别系统来阐述如何使用 HTK 工具。

2.1　　HTK 软件结构

HTK 中大部分的功能都被封装成库。这些模块保证每个对外的接口使用相同的方式。也提供了一些通用函数资源。Fig2.1 图解了一个典型的 HTK 工具的组成，并且展示其输入/输出接口。

用户的输入输出以及一些和操作系统的交互是由 HShell 模块来控制。HMem 负责内存管理。HMath 提供数学计算的支持。HSigP 提供信号分析中需要的信号处理。HTK 中的每种文件类型均有专门的接口模块。HLabel 提供 label 文件的接口。HLM 提供了语言模型文件接口。HNet 提供了网络和网格文件的接口。词典通过 HDict 接口，矢量量化的码本通过 HVQ 接口。HMM 定义则通过 HModel。

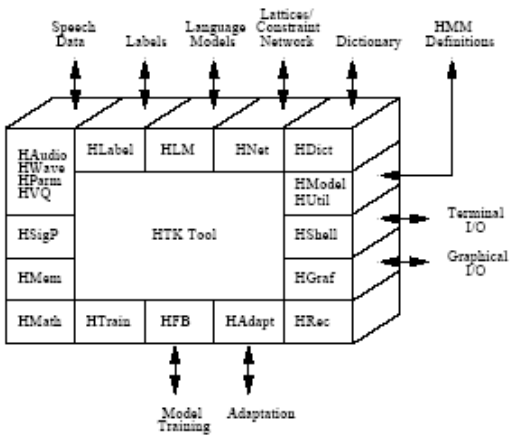


Fig. 2.1 Software Architecture

所有以 WAV 形式输入输出的语音通过 HWave 处理，以参数形式的输入通过 HParam 处理。为了提供一致的接口，Hwave 和 HLabel 提供多种文件格式的支持（可以从其他系统导入的数据）。HAudio 支持直接语音输入。HGraf 提供了简单的交互解码。HUtil 为操作不同的 HMM 模型提供了很多有用的函数，HTrain 和 HFB 支持不同的 HTK 训练工具。HAdapt 为不同的 HTK 自适用工具提供了支持。最后，HRec 包括了主要的识别处理功能。

下一部分将会讲到：通过设置一下配置项，来控制这些库模块的行为。具体这些库模块函数的详细描述将会在本手册的第二部分中描述，并介绍出现的相关的配置。另外在 18 章会给出完整的列表作为参考。

2.2 HTK 工具的通用属性

HTK 工具通过命令行来运行。每个工具都有很多必须参数和可选参数。可选参数使用‘-’前缀。下面的命令将调用虚构的叫做 HFoo 的 HTK 工具。如下：

```
HFoo -T 1 -f 34.3 -a -s myfile file1 file2
```

这个工具中有两个必须参数file1、file2和四个可选的参数。可选参数使用单字母，后面跟着可选的值。其中值和参数名用空格分开。-f后面是一个实数，-T后是一个整数，-s后是一个字符串，-a后没有参数，仅仅用来使能或者禁止工具中的某些特性。大写的选项在所有工具中的意义是一样的。-T是用来控制HTK输出Trace。

另外，参数可以储存在配置文件，我们可以通过这些文件里的参数来控制工具的操作。例如：

```
HFoo -C config -f 34.3 -a -s myfile file1 file2
```

如果这条命令执行，HFoo工具将会在初始化的时候装载这些存储在配置文件里的参数。多个配置文件可以使用重复使用-C选项。例如：

```
HFoo -C config1 -C config2 -f 34.3 -a -s myfile file1 file2
```

配置文件通常作为命令行参数的另一种配置方式。例如，trace选项一般总是在配置文件里设置。然而，配置文件主要是用来控制所有HTK工具依靠的库模块的详细行为。

虽然这种命令行的风格比起现在的图形界面显得很老套。但是实际中，可以通过书写一些脚本即可以控制HTK工具的执行。对于大型的系统构建和试验这是非常重要的。而且，使用文本方式定义所有操作，可以使得系统构建和试验时的细节被录制和记录下来。

最后，不带参数执行这个工具，可以得到其中命令行的解释以及所有HTK工具的选项。

2.3 工具包

我们通过建立一个基于子词（sub-word）模型的连续语音识别系统并介绍这些过程，将使我们很好的了解 HTK 工具。如 Fig2.2 所示：总共有 4 个主要过程：数据准备，训练，测试和分析。

2.3.1 数据准备工具

为了建立一个 HMM 模型集合，语音数据文件和他们的相关的标注是必需的。语音数据将会从数据库（一般存储在光盘中）获得。在训练之前，这些数据都将会被转换成适当的参数形式。相关的标注将被转化成正确的格式，使用需要的音素或者词的 lable。如果需要，HSLab 可以用来录制语音，然后对相关的内容进行手工标注。

虽然所有的 HTK 工具能够在过程中提取语音数据的参数，但实际上通常我们最好只对数据进行一次特征提取。HCopy 就是为了做这个。通过字面意思，HCopy 用来复制一个或者更多的源文件到输出文件。一般，HCopy 将会复制整个文件，但是为了分段和连接文件，还要提供多种机制。通过设置一些合适的配置变量，所有的输入文件在被读入时将会转为参数形式。因而，通过这种简单的复制每个文件的机制，对每个文件完成必须的特征提取过程。HList 工具用于检查语音文件的内容，当输入文件被转化后，可以使用 HList 在处理大量的数据之前检查转化结果。标注文件也需要准备。原始的标注不需要特别精确，但是确实必需的。例如：因为不同的音素集的差异，HMM 训练需要上下文无关的标注。HLEd 工具是一个脚本驱动的标签编辑器。用于生成必要的标注文件。HLEd 能够输出到单个 MLF 文件（Master Label file）通常这个文件可以使接下来的处理变得更加方便。最后在数据准备阶段，HLState 能够收集和显示在标签文件里的统计信息。在建立一个离散 HMM 系统准备阶段，**HQuant 被用来建立一个矢量量化的码本。**

2.3.2 训练工具

系统构建的第二步就是为每个HMM模型定义其拓扑结构。HTK可以给HMM模型构建任何拓扑结构。HMM的定义可以存储在外部的文本文件中。因而可以使用任何文本编辑器来编辑。HTK的标准发布版本包括很多HMM模型的例子和自动生成通用拓扑结构的脚本以供选择。除了转移概率，在原型定义中给定的HMM模型的参数并不重要。原型定义的目的是阐述总体的特性和HMM模型的拓扑结构。实际具体的参数将会在后面的训练过程中计算出来。需要给定有特定意义的转移概率的值[译者注：转移概率能表示HMM的拓扑结构]，这些初始值并不会影响训练过程。选择这些概率的策略是，使所有状态跳转的转移概率基本相似。

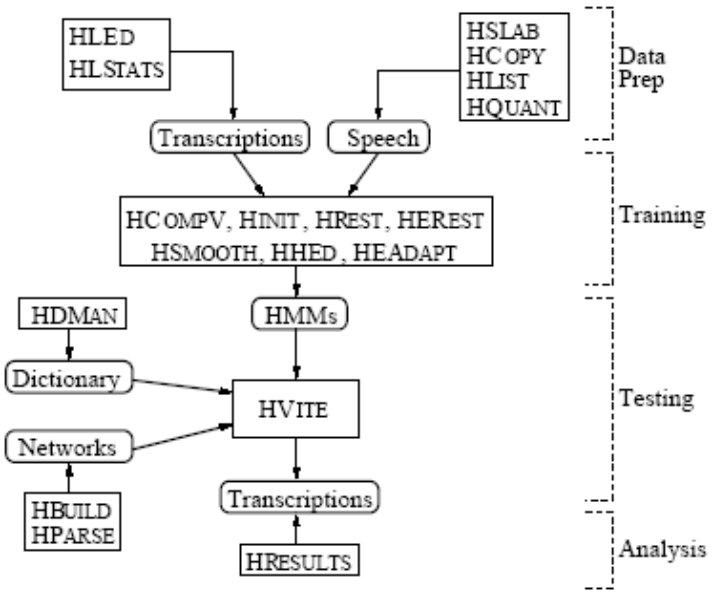


Fig. 2.2 HTK Processing Stages

下面开始介绍实际的训练过程，Fig2.3详细阐述了其细节。首先，创建初始模型集。如果有一些已经标注了子词（如：音素）的边界的语音数据，则这些数据已经可以用来作为初始化（bootstrap）数据。在这个例子中，HInit和HRest使用完全被标注过的初始化数据按照孤立词风格来训练，单独生成每个HMM模型。**HInit读入所有的初始化训练数据并切出所有的音素，然后使用K均值分类程序反复计算初始参数值。在第一重循环，训练数据被均一的分类，模型的每个状态将会匹配到相应的数据部分，然后估算均值和方差。如果按照*混合高斯模型*来训练，将使用修正形式的K均值聚类算法。**第二步和接下来的循环中均一分类部分将用Viterbi对齐代替。HInit计算出来的参数值将会通过HRest被重估。完全被标注的初始化数据再次被使用，但是此时的K均值分类算法将被取代，使用的是前章提到的Baum-Welch重估算法。当没有初始化数据的时候，使用一种所谓的“统一开始（flat start）”方式。这种方式下，所有的音素模型将被初始化为相同的参数，所有状态均值和方差等于全局语音的均值和方差。全局的均值和方差将由HcompV得到。

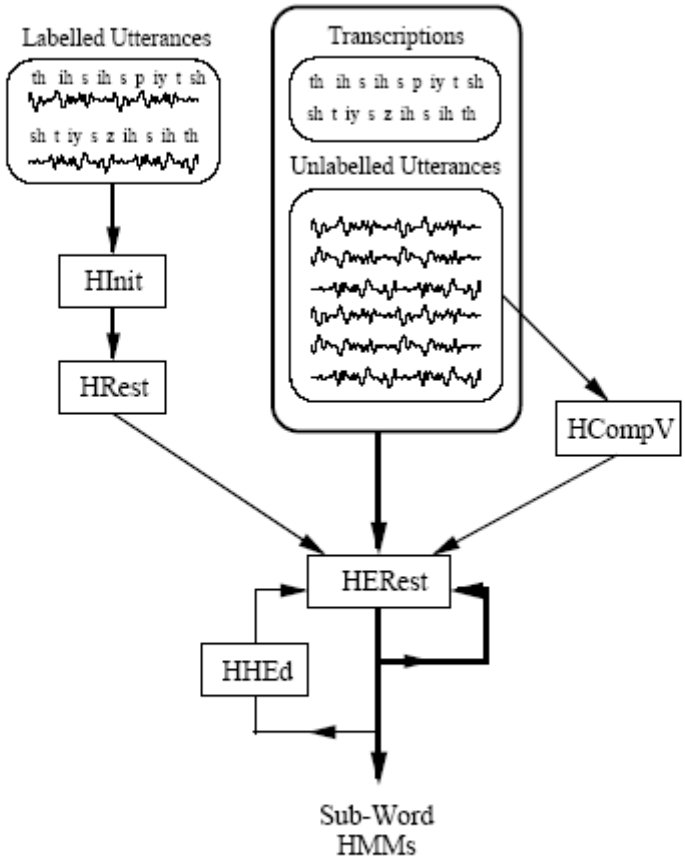


Fig. 2.3 Training Sub-word HMMs

一旦初始的模型集被创建后，HERest使用整个训练集来执行“嵌入式训练（embedded training）”，HERest将对全部HMM音素集模型执行一次Baum-Welch，同时重估这些模型的参数。对于每条训练语句，此训练数据包含的相应的音素模型将会被连接起来。前向-后向算法将对序列中的每个HMM模型累加状态占有率、均值、方差等统计信息。当处理完所有的训练数据后，累加的统计量将被用来计算模型的参数重估值。**HERest是HTK训练工具的核心。其被设计用来处理大规模的数据库。**也可以很方便地使用裁减策略来减少运算量。能够在网络中并行执行（译者注：可将训练数据分成多份，同时并行计算，然后使用累计的统计量重估模型）。

在HTK的体系下， HMM模型的参数训练是逐步精确细化的。因此，一个典型的处理方式是从一个单高斯的上下文独立的音素模型集开始，通过扩展到上下文相关来逐步精确，然后使用多维混合高斯分布。**HHEd是编辑HMM模型的工具，将原模型克隆成上下文相关的模型，采用不同的参数绑定，在一些发布版中增加了混合高斯分布的数目。**一般的处理使用HHEd来修改HMM集合。使用HERest重估修正后的参数。为了提高特定发音人的表现，HEAdapt和HVite能调整HMM集合，目的是用少量的训练或自适应数据来更好地模拟特定说话人的特征。最终的结果即是说话人自适应的系统。

现在建立一个上下文相关的系统唯一的最大的问题就是数据不足。模型集合越复杂，就需要更多的数据来训练，保证参数重估的鲁棒性。因为数据总是有限的，我们就必需在系统复杂度和可用的数据找到一个折中的方法。对于连续密度系统通过将参数绑定来平衡。参数绑定（Parameter tying）允许将数据聚积加以共享，这些共享参数可以使参数估计更具鲁棒

性。除了连续概率分布的系统，HTK将支持连接混合系统和离散概率分布的系统。在这些情况下，不充足的数据通过平滑这些分布来解决。HSmooth即为此而设计。

2.3.3 识别工具

HTK为识别单独提供了工具HVite，采用前一章提到的token passing算法来实现基于Viterbi的语音识别。HVite需要以下输入：一个描述待识别单词序列的网络、定义单词如何发音的词典以及一个HMM模型集合。HVite会将词网络转化成音素网络，并为每个音素实例绑定适当的HMM定义。而后，识别器即可处理已存储的语音数据文件或者直接的音频输入。在最后一章结尾将会提到，HVite 支持cross-word triphones，并可实现用多token生成多个输出路径的lattice，同时也可以被配置用于Rescore lattice或和Force Alignment。

驱动HVite的语法网络通常是简单“词环”（*word loop*）（任何词可以跟在任何其他词后面），或者是代表有限状态的语法图。第一种情况下，词的转移一般绑定了*bigram*概率。词网络使用HTK标准的网格格式存储，是文本形式的，因而可使用文本编辑器直接创建词网络。这是一个很繁琐的工作，因此HTK提供了两个工具来辅助创建词网络。第一个，HBulid允许创建子网络并能在上层网络中使用。因此，虽然可能会用到处于相同层次的符号，但是可以避免多次复制。HBuild也能够被用来创建词loop，也能够读入backed-off bi-gram语言模型，通过合并bigram概率修正词loop的转移概率。前面提到过，标签统计工具HLState能够用来产生backed-off bigram语言模型。

另一种可选的方法，是使用更上层的符号直接指定词网络。这种符号是基于用在编译器规范中的EBNF（the Extended Backus Naur Form），并且和HTK早期版本中使用的语法规范兼容。工具HParse用于来将这种符号转化为等价的词网络。

无论采取何种方式来产生词网络，如果能看到它定义的语言例子，都将会很有用。HSGen工具就是这种用途。它的输入是一个网络，然后随机穿越网络并输出词串。可以通过查看这些串来确保他们是否符合需要。HSGen也能计算出任务的复杂度。

最后，大词典的构建能够包括合并几个源输入并对每个输入进行多种变化，词典管理工具HDMan用来辅助这个过程。.

2.3.4 分析工具

一旦基于HMM的识别器建立好，评测其效果是必不可少的。通常通过使用转化一些预录制的测试样句，与正确的参考转化数据来匹配识别器的输出。HResults这个比较工具利用动态规划来对齐标准答案和识别结果。然后计算替代，删除，插入错误。提供的选项保证被HResults使用的算法和输出格式与NIST使用的是兼容的。和全局表现测试一样，HResults也能够提供speaker-by-speaker breakdowns，混淆矩阵(confusion matrices)和时间对齐标注。对于关键词检出应用，它也能计算*Figure of Merit* (FOM)分和*Receiver Operating Curve* (ROC)信息。

2.4 V3.2 的新特性

这个部分将会列举出 HTKV3.2 同先前的 V3.1 版本相比较的新特性：

1 HLM 工具包被包括在 HTK 中，支持训练和测试 word 或者基于类的 n-gram 语言模型。
2 HPARM 支持全局特性空间转化。
3 HPARM现在支持三级微分。
4 HLRescore这个新工具为大量网格的后向操作（例如网格裁减）提供了支持。找到网格中的最优路径和网格的语言模型扩展。
5 HERest 支持两模型的重估，实现方式是在Baum-Welch重估算法中，使用其中一个模型的alignment结果重估另一个模型的参数。
6 决策树状态聚类的初始化在HHEd中被提升。
7 HHEd支持一些与方差下限和减少混合数目相关的新命令。
8 修正了块对角矩阵的MLLR转化估计中的主要bug。
9 加进去很多小的修正和bug修复。

2.4.1 V3.1 的新特性

这部分将要列举出 V3.1 版本相对于 V3.0 版本（功能等同于 V2.2）的新特性

1 HPARM 支持 PLP（Perceptual Linear Prediction）特征提取。
2 HPARM 通过在 filterbank 分析中变化频率轴来实现 VTLN（Vocal Tract Length Normalisation)。
3 HPARM 支持方差缩放。
4 HPARM 支持 倒谱均值聚类和方差规整。
5 所有的工具支持扩展文件名语法，可以很方便的处理未分段的数据。

2.4.2 V2.2 新特性

这部分将要列举出 V2.2 版本与 V2.1 版本相比的新特性

1 说话人自适应现在可以通过 HEAdapt 和 HVite 支持,根据新的发音人或者环境得到的模型进行自适应。
   - HEAdapt 采用 MLLR 和或者 MAP 来实现离线有监督的自适应。
   - HVite 用 MLLR 来实现无监督的自适应

   这两种工具在静态模式下使用，所有的数据在任何自适应之前呈现或者使用增加的方式。
2 增加了对 PC WAV 文件的支持，除了 16bit PCM 线性，HTK 还能支持：
   - 8-bit CCITT mu-law
   - 8-bit CCITT a-law
   - 8-bit CCITT a-law

2.4.3 V2.1 增加的特征

为方便用过 HTK 早期版本的用户，这一部分列出了 V2.1 与 V2.0 相比的主要改变。

1 对语音输入处理的部分进行了重新设计，在HParm中结合了一个新的基于能量的语音/静音检测。这一检测具有鲁棒性并可以用一些配置变量来配置。语音/静音检测现在可以在WAV文件上运行。现在可以通过要求用户讲出任意一句话来完成语音/静音检测参数的测定。

2 现在通过配置参数ADDDITHER HParm允许WAV数据中带有随机噪音信号。这样避免了数字溢漏，溢漏是由于在一些编码方式下人为地形成了WAV数据。

3 当用HVite对说话人进行强制校正时，HNet可以进行更有效的操作。对基于biphone/triphone的音素识别结合了更深的网络优化。

4 即使当没有token到达网络的结尾时，通过设置HRec配置参数*FORCEOUT*为真，HVite现在可以产生部分识别假设。

5 词典可以扩展到允许一个单词具有不同发音概率。同时，现在HVite允许在识别器中用一个发音刻度因子。

6 通过结合自动位交换的不同机器，HTK现在支持HTK二进制文件的读写（WAV形式，二进制MMF，二进制SLF，HERest累加器）。用这些工具，所有预先设定的二进制数据文件可以读/写成大头(NONVAX)字节序列。通过配置参数*NATURALREADORDER*和*NATURALWRITEORDER*能改变设定好的操作。

7 HWave支持Microsoft WAV文件形式。

8 HAudio允许用按键控制来自动输入。