CSLT TECHNICAL REPORT-20160015 [Friday 6th January, 2017]

基于Kaldi的哈萨克语语音识别系统

Ying Shi[1]
, Zhiyuan Tang[1]
, Nurbolat [1]
and Dong Wang[1*]

[*]Correspondence:
wangd99@mails.tsinghua.edu.cn,
[1]Center for Speech and Language Technologies, Research Institute of Information Technology, Tsinghua University, ROOM 1-303, BLDG FIT, 100084 Beijing, China
Full list of author information is available at the end of the article

Abstract

近年来深度学习技术 [1]在人工智能领域得到广泛应用，语音识别作为人工智能的一个分支也得到了前所未有的发展，越来越多的科研机构及公司投身到了语音识别领域。尽管如此，哈萨克语作为我国众多小语种中较为重要的一种，依然没有一套成型的可供科学研究及工程应用的语音识别系统。本文将基于语音识别工具Kaldi，从数据准备阶段开始介绍如何从无到有构建哈萨克语语音识别系统。本文的受众对象为初入语音识别领域的读者，所以在这篇文章中没有过多的专业知识的介绍，旨在降低语音识别及Kaldi的入门门槛，让没有太多基础的读者能够快速使用自己的数据构建语音识别系统。

Keywords: Kaldi; 语音识别

1 简介

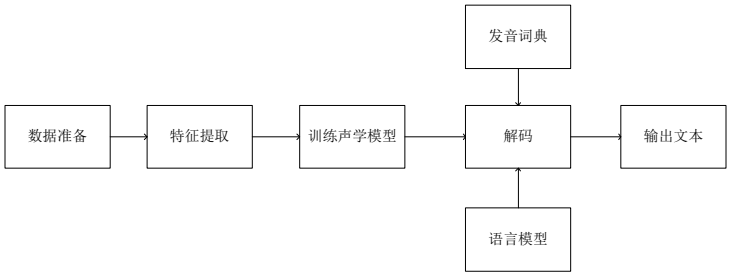

Figure 1: 语音识别流程

上图展示了Kaldi中语音识别的基本流程，其中特征提取、声学模型训练、解码是主要环节，并已被集成于Kaldi的recipe当中，可以通用于不同的数据集，这为新的语音识别系统的快速搭建提供了保证。针对一种新的语言（比如哈萨克语）或数据库，相较于Kaldi中已经提供的例子（比如WSJ），最大的区别在于语言自身的特性和数据的结构，因此，将Kaldi已有的recipe快速应用到新的语言上，必须将该语言的数据集处理成Kaldi所需的格式，这属于数据准备阶段（发音词典和语言模型包含在这一步中）的工作，数据准备完成后，将数据放在recipe指定的目录下，接下来的过程如同WSJ recipe描述的一样，运行上层脚本完成语音识别系统的训练。

可见，如果在Kaldi下快速搭建一个全新的语音识别系统，主要工作在于数据准备。本文以哈萨克语为例，先介绍哈语的特性，再基于上图的流程逐步介绍如何搭建哈萨克语语音识别体统，其中数据准备将是我们详细介绍的部分。

2 哈语的语言特性

2.1 哈语简介

ڭ	ن	م	ل	ق	ك	ي	ز	ج	ه	د	ع	گ	ۇ	ب	۴	ا	ء
N	n	m	l	q	k	y	z	j	E	d	G	g	V	b	A	a	v
	ى	ۇ	ى	ش	چ	ھ	ح	ف	ۋ	ۋ	ۋ	ت	س	ر	پ	ۋ	و
	i	e	x	c	H	h	f	U	u	w	t	s	r	p	O	o	

Figure 2: 哈文字母对照表

哈萨克语属阿尔泰语系突厥语族克普恰克语支，其形态结构为黏着语类型。哈萨克语共有33个音，其中9个是元音，24个是辅音，新疆哈萨克文是以阿拉伯字母为基础的拼音字母，字母是从右向左写，词和词之间必留一定的空隙。而且有些字母有两种书写形式，大部分字母有四种书写形式。用那个形式按照字母在词中出现的位置来定，都有一定的规则。

哈萨克文字中有些元音有附加部分，所以用拉丁字母小写“v”来表示其元音的附加成分，哈萨克文字里有九个元音，五个元音没有附加成分，四个元音有附加成分，如下表所示（第一行代表没有附加成分的元音，第二行则代表有附加成分的元音）：

a	o	u	e
va(A)	vo(O)	vu(U)	ve(i)

2.2 元音和谐率

一般一个单词里虽然有好多个有附加成分的元音，但一个单词里只能出现一个附加成分，而且不能出现在单词中间，只能放在单词前面(单词以附加成分起首)。

哈萨克语的单词中，前音节对后音节的影响特别大，一个单词里的所有元音存在着明显的互相调谐、配合、匹配的现象。

(1) 单词里第一个元音是前元音，那么其单词里所有元音都是前元音(一个单词里第一元音是有附加成分的元音，那么后面的几个元音都是有附加成分的元音)。如果单词首有元音附加成分，那就代表这单词里的元音都是有附加成分的元音组成的。比如：vbare(=bAri) 这单词里第一个字母是小写“v”那么单词中的两个元音a和e 都是有附加成分的va(=A)和ve(=i)

(2) 单词里第一个元音是后元音，那么其单词里所有元音都是后元音(如果单词里第一个元音没有附加成分的元音，那么后面的几个元音都是没有附加成分的元音)，如果单词首没有附加成分，那就代表这单词都是没有附加成分的元音组成的。比如：qaren 这单词里没有附加成分，那么就元音a和e是没有附加成分的元音。

2.3 元音和谐率的特殊情况

元音的附加成分有时候会受到辅音的影响，如果只要单词里出现两个辅音“k”，“g”和元音“E”，那么单词就省略附加成分，但单词里的元音还是当做有附加成分的元音。也就是说虽然单词前面没有附加成分小写“v”，但单词里出现这三个字母，我们就认为单词前面有附加成分

比如：kareN 这单词中应该前面有元音的附加成分，可是出现了辅音“k”，那么省略附加成分，但还是当作有附加成分的单词。只要单词里“k” 则默认单词前有附加成分“v”

比如：elgEn 这个单词中应该前面有元音e的附加成分“v”，但辅音“g”出现了，所以省略元音附加成分，但还是当作有附加成分的单词。只要单词里“k” 则默认单词前有附加成分“v”。

比如：esE 这个单词中应该前面有附加成分，但元音“E”出现了，所以省略附加成分，但还是当作有附加成分的单词。只要单词里“E” 则默认单词前有附加成分“v”

像这样的细节还有很多，我们会继续探索，并对这部分章节进行补充与完善。

3 数据准备

本章我们将分别介绍以下三方面数据的准备：(1)原始数据，主要包含音频文件，及其对应的转录文本。(2)基本数据，主要包括语音特征，及其对应的转录文本、说话人信息，以及发音词典等。(3)文本语料，主要用于语言模型的训练。

3.1 原始数据

原始数据包含音频文件和转录文本两大部分。音频文件是由专业设备按照统一标准录制而成。每个音频文件的名称（ID）中可包含性别、说话人等相关信息，这种命名方式最大程度上包含了我们后边需要用到的信息，可降低数据准备所需要的工作量。比如，我们将某个哈语音频文件组织成以下形式：

/work3/shiying/kazak/kazak-spdb-src/lvcsr/data/train/F0101001.wav

F0101001 是这段音频的ID，其中第一个字母代表说话人的性别（F代表女性，M代表男性），这个字母与紧接着的4位数字0101共同构成了说话人的编号，最后的三位数字001代表这位说话人所说的第一句话。

转录文本与音频文件是一一对应的。我们将上例中的音频文件所对应的文本组织成一下形式：

psyhykAliq mAsElEniN jastanwGa bEtAlwinAn oqw aGartwdaGe...

这个文本文件所在的位置为：

/work3/shiying/kazak/kazak-spdb-src/lvcsr/speech/py/Group_01.txt

这样的文件有10个分别为Group01-10，其中包含了每条音频文件对应的文本信息，每个文本的第一条对应ID尾数为001的音频文件后边的依次类推，它的总数必须与音频文件的总数相同。

3.2 基本数据

基本数据，包含了 a)作为系统输入的语音特征及其对应的转录文本、说话人信息，以及 b)发音词典，它们分别放在了两个不同类型的文件夹中：

3.2.1 data/{train,test,dev}

这三个文件夹中存放的是作为系统输入的语音特征及其对应的转录文本、说话人信息，实际上这三个目录下的文件的名称是一致的，只不过各自的使用场景不一样，train 是声学模型的训练集，test是系统的测试集，dev是开发集或交叉验证集。nnet1中有用到dev，我们所做的哈语语音识别是基于Kaldi nnet3的，所以实验中并未用到dev。我们以train为例来介绍这类文件夹应该包括的内容。

首先从原始数据中梳理出相关信息（性别、说话人等），存在不同的文件中，放于data/train之下：

text	wav.scp	utt2spk	spk2gender	spk2utt

- text，文件内容如下：

F0101_001	psyhykaleq masElEneN jastanwGa bEtalwenan ...
F0102_002	bEyjyNtyanjynhebEy vux ozEn aterawe jwjyaN ...

在这个文件中包含了两列，其中第一列代表某句话的ID，可以看到与之前提到的命名方式相同下划线前的部分代表说话人的ID下划线后边的部分代表该说话人所说的第几句话。第二列则代表这句话的内容。对于出现在第二列的内容，最好不要包括标点符号，因为第二列的每个元素都要在后文我们将提到的Lexicon中有对应的发音，为了省去不必要的麻烦，我们去掉了所有的标点符号，同时我们也建议初学者这样做。另外我们得到的最原始的哈语语料并不是英文字母形式，而是阿拉伯文的形式，从阿拉伯字母到英文的字母的转换我们使用了新疆大学米吉提老师提供的CodeMap。用法如下：

```
在chars目录下
./program/codetransform.pl orgfilename newname
```

其中orgfilename代表需要转换的文件，转换后的文件将被写在newname中。

- wav.scp，文件内容如下：

F0101_001	/work3/shiying/kazak-spdb-src/.../train/F0101001.wav
F0101_002	/work3/shiying/kazak-spdb-src/.../train/F0101002.wav

在这个文件中，第一个元素与text中第一个文件相同，第二个文件则代表这句话的音频文件的具体位置，这里最好使用绝对路径。在wav.scp 中每个音频必须在text中有唯一的一条语句与其对应。

- utt2spk，文件内容如下：

F0101_001	F0101
F0102_002	F0102

utt2spk表示句子和说话人的对应关系，其中每一行的第一个元素代表句子的ID，第二个元素代表说话人的ID。

- spk2utt，文件内容如下:

F0101	F0101_001 F0101_002 F0101_003...
F0102	F0102_001 F0102_002 F0102_003...

与utt2spk相同spk2utt代表说话人与句子的对应关系，每一行的第一个元素代表说话人的ID，另一个元素则代表句子的ID。在Kaldi的众多脚本中使

用kaldi/egs/thchs30/utils/{utt2spk_to_spk2utt.pl ,spk2utt_to_utt2spk.pl} 这两个脚本可以实现utt2spk 与spk2utt 这两个文件之间的相互转换。所以读者只需要手动创建其中的一个文件，另一个可以通过脚本生成。

- spk2gender，文件内容如下:

F0101	f
M0102	m

很显然这个文件包括的是说话人以及他们的性别信息。所以第一个元素代表说话人的ID，另一个元素则代表该说话人的性别，其中f代表女性，m 代表男性（性别需要用小写）。以上所述的四个文件都需要进行排序，值得注意的是，Kaldi源码是使用C++ 语言实现的，读者最好在自己的环境变量里加上“export LC_ALL=C”，否则在排序时可能发生排序原则与C++ 不符合，导致在后边的脚本中出现错误。该文件在系统的训练当中是非必需的。

接着，我们便可以进行语音特征（MFCC或Fbank）的提取，并将在data/train中增添以下文件。

cmvn.scp	**feats.scp**	wav.scp	text	utt2spk	spk2gender	spk2utt

特征提取的相关流程已写入Kaldi 的recipe，并不是本文的重点。特征提取部分的说明将放在Kaldi 训练脚本的讲解中。

3.2.2 data/dict

该文件夹包含音素、词典相关的说明文件，都需要人工准备。

lexicon.txt	nonsilence_phones.txt	silence_phones.txt	optional_silence.txt

- lexicon.txt，文件内容如下：

#	SIL
$< SPOKEN\_NOISE >$	SIL
SIL	SIL
Anen	A n e n
ENbEge	E N b E g e

lexicon.txt是dict目录下较为重要的一个文件，这个文件其实是一个字典，但与普通字典不同的是，这个字典包含的是完整的字以及这个字对应的发音，

所以在lexicon.txt中第一列代表字，第二列则代表这个字的发音序列，在哈语中每个字母都有自己的发音，所以在经过阿拉伯字母到英文字母的转换后，每个字的发音序列就是组成这个字的字母的序列。在这个文件中每个字都必须对应至少一个发音，可以存在多音字。理论上lexicon.txt 应该包含某种语言的所有字和所有的发音，但是由于哈语语料的限制，我们只使用了出现在哈语语料中的所有的字。在这个文件的前三行，是三个较为特殊的字符。#代表可能出现在语料中的某种字符，$< SPOKEN_NOISE >$代表噪音，SIL代表静音，我们将这三个特殊字符都对应到了SIL（silence）静音上。

- nonsilence_phones.txt，文件内容如下：

A
E
G
H
N
.
.
.

这个文件中必须包含所有非静音的音素。

- silence_phones.txt，文件内容如下：

SIL

这个文件仅包含一个元素，就是静音的音素的符号。在哈语中我们使用SIL。

- optional_silence.phones.txt，文件内容如下：

SIL

这个文件中包含被设为静音的音素，在哈语中也是只有SIL。

3.3 文本语料及其生成的语言模型

训练语言模型的文本语料可以是训练集的转录文本本身，但最好是额外提供的同领域大批量语料。语言模型的训练可以通过使用工具srilm完成。在这里我们要简略介绍一下n-gram [2] 模型的生成。n-gram 模型是一种基于统计的模型，即一个句子中的一个词出现的概率只与前N-1个词有关，与其他词无关，根据概率的知识很容易得到一句话出现的概率就是所有词出现概率的乘积，所以1-gram模型就是基本的词频统计，2-gram模型考虑词对，3-gram模型则考虑三元组。

制作语言模型只需要两个文件，第一个是语料，第二个则是词表。用于训练语言模型的语料，原则上要求足够大能够涵盖该语言在各个领域的表达，但是这么全面的语言模型用在较为简单的语音识别的任务上往往是非常浪费的，所以读者可以为自己的语音识别任务准备一个较小的语言模型，但是这个语言模型最好能与test集的语料处于相近或相同的领域，否则会对语音识别的效果有一定的影响，同时我们需要能够保证用于训练语言模型的语料每一行都具有实际意义，对于哈语我们的做法是按照句号问好感叹号换行，保证每一行是一个完整的句子，对于语料中出现的数字也有一些特别的要求，一般语料中的数字都是阿拉伯数字，对于这些阿拉伯数字最好的做法就是将数字转换成相应语言的特定格式，例如：“2016” 转换成汉语的“二零一六”，但是对于哈语将所有的阿拉伯数字转化成哈语形式的数字是非常困难的，所以我们的做法是将数字用空格隔开，即“2016”转换成“2 0 1 6”，再将0-9这10个阿拉伯数字加入到前边提到的lexicon.txt 中即可。训练语言模型所使用的词表就是组成语料的词表，但是我们往往不需要用全部的词表的去做语言模型，读者可以根据自己的需要对词表按词频进行裁剪，对于哈萨克语我们按词频从高到低排列取了前10 万的词。

我们所做的哈萨克语的语音识别所使用的语言模型是3-gram模型，制作语言模型的脚本为run_lm.sh （这个脚本由工程师张之勇提供）。

```
run_lm.sh  corous  order  name  vocab
```

其中corpus代表用于训练语言模型的语料，order代表使用几gram模型，name是制作出来的语言模型的命名，vocab是词表。这样制作的语言模型对于语音识别任务来说总是存在一些冗余，往往需要进行裁剪，幸运的是srilm提供了裁剪工具。如下：

```
ngram -prune parameter -lm LMname -write-lm newLMname
```

prune代表裁剪语言模型所用的参数，例如对于哈语语音模型的裁剪时的参数，我们使用了2e-7，代表词频概率取对数后小于$2 * 10^{-7}$ 的全部去掉。LMname 代表需要裁剪的语言模型，newLMname代表裁剪后的语言模型的命名。语言模型的准备工作到这里就基本全部结束了，剩下的工作就是就运行Kaldi中的脚本然后等待结果。

至此，一个语音识别系统所需的所有数据都已准备完毕，接下来便是将数据路径传入相关训练脚本，执行整个语音识别系统的流程。

4 语音识别系统流程简述

当所有数据准备好后，则可以按照Kaldi的recipe完成语音识别系统的搭建。本章以Kaldi/egs/thchs30/s5/run.sh为例（我们所做的哈萨克语语音识别系统也是以这个脚步作为参考），对训练脚本执行过程中的相关步骤进行简要说明，以增强初学者对Kaldi中语音识别系统的了解。

4.1 特征提取

语音识别所用的机器学习的算法是监督学习，经过上一步的数据准备，我们已经将监督学习所需要的输入以及输出准备完成了，但是Kaldi 对于用于训练模型的输入与输出有自己的格式要求，特征提取其实就是将输入转化为Kaldi所要求的格式，对于输入数据，Kaldi 提供了两种不同的特征提取的方式：MFCC以及FBANK。其中MFCC多用于训练GMM [3]模型，而FBANK 多用于DNN模型，这两种特征提取所用的脚本分别为“kaldi/egs/thchs30/s5/steps/make_mfcc.sh”和“kaldi/egs/thchs30/s5/make_fbank.sh”。这两个脚本所需要传入的参数都是一样的，所以我们以make_mfcc.sh为例介绍这两个脚本所需要的参数以及他们所生成的文件的含义。

make_mfcc.sh

```
steps/make_mfcc.sh –nj $n –cmd ”$train_cmd” data/mfcc/train
exp/make_mfcc/ mfcc/train
```

每个参数的含义：

- -nj	这个参数代表所使用的job 的个数，例如我们将这个参数设置为8，那么Kaldi 则会将输入数据分为8份然后投递到8个节点去运行
- -cmd	这个参数代表使用哪种q函数
data/mfcc/train	输入数据所在的目录
exp/make_mfcc/	脚本的log 文件将被写在这个目录下
mfcc/train	特征提取的结果将被写在这个目录下

这个脚本最终的产物是feats.scp 它的内容为:

F0101_001	/work3/shiying/kazak/mfcc/train/raw_mfcc_train.1.ark:10

Kaldi会通过特征提取将音频文件转换成矩阵格式，每一句在一个矩阵中，矩阵的每一行对应音频文件的一帧(一般为25ms)。feats.scp与前边数据准备时所用的wav.scp相对应，第一列是某句话的ID，第二列是这句话在Kaldi 标准格式的矩阵中所在的起始位置。以上述的列子来说“/work3/shiying/kazak/mfcc/train/raw_mfcc_train.1.ark:10”，代表F0101_001这句话在raw_train_1.ark中起始位置为第十个字节。

提取提取的第二步时计算cmvn

steps/compute_cmvn_stats.sh data/mfcc/train exp/mfcc_cmvn/train mfcc/train

每个参数的含义:

data/mfcc/train	输入数据所在的目录
exp/make_cmvn/	log文件将被写在这个目录下
mfcc/train	计算cmvn的结果被写在这个目录下

通过计算cmvn所得到的文件与feats.scp的格式是相同的:

F0101	/work3/shiying/kazak/mfcc/train/cmvn_train.ark:6

其中第一列元素代表说话人的ID，第二列元素也是一个矩阵，这个矩阵包含了说话人的统计倒谱均值和做过normalization的方差。

4.2 词典的FST转换

下边这个脚本主要作用是在data/lang下边生成一些文件。

utils/prepare_lang.sh - -position_dependent_phones false data/dict "$< SPOKEN\_NOISE >$" data/local/lang data/lang

每个参数的含义为:

- -position_dependent_phones false	是否使用词位信息，对于哈萨克语的识别我将它设为false
data/dict	这个脚本所需要的输入数据所在的目录
$< SPOKEN\_NOISE >$	oov词汇将被映射到这个符号上
data/local/lang	临时输出的目录
data/lang	最后的结果将被写在这个目录下。

这个脚本主要围绕数据准备阶段所准备的lexicon相关的文件做一些处理。在data/lang目录下会生成很多文件其中比较重要的两个是L.fst和L_disambig.fst这两个其实是lexicon的fst（finite state transducers 有限状态转换机）版本。对于初学者没有必要深究这其中的实现方式。

4.3 语言模型的FST转换

另外一个脚本与前边所讲述的作用一样，主要是对语言模型的处理:

```
utils/format_lm.sh data/lang data/graph/kazak.3.lm.gz
kazak/lm/lexicon.txt data/graph/lang
```

每个参数的含义为:

data/lang	L.fst 所在的目录
data/graph/kazak.3.lm.gz	所准备的语言模型（注意语言模型需要被压缩为.gz格式）
kazak/lm/lexicon.txt	lexicon.txt
data/graph/lang	最后的输出将被写在这个目录下

这个脚本的主要产物为G.fst，与前文提到的L.fst 相同它是语言模型的fst 版本。

4.4 GMM模型训练

Kaldi对于GMM模型的训练以及解码都遵循同样的流程即：训练模型（本节）-解码测试（4.5节）-数据标注（4.6节）- 使用标准数据训练下一个模型。我们以训练monophone 为例讲解相关脚本的作用以及参数的意义。

训练monophone所使用的脚本为：

```
steps/train_mono.sh - -boost-silence 1.25 - -nj $n - -cmd "$train_cmd"
data/mfcc/train  data/lang exp/mono
```

每个参数的含义为:

- -boost-silence 1.25	提升静音似然度的因子。
- -cmd "$train_cmd"	与前文所介绍的cmd意义相同
- -nj $n	与前文所介绍的nj意义相同
data/mfcc/train	输入数据的目录
data/lang	L.fst 所在的目录
exp/mono	最终的模型将被写在这个目录下

模型训练结束后，在exp/mono目录下我们会得到一个名为fina.mdl （gmm 模型）的模型。这个文件是最终GMM模型的二进制拓扑结构，我们可以使用“copy-transition-model –binary=false final.mdl final.txt”将这个二进制文件转换成txt格式以便查看其中的内容。

4.5 GMM-HMM解码

在解码的时候，我们需要使用前边的结果准备一个新名为：HCLG.fst 的文件,这个文件是由H C L G通过特定方式合成的。其中L与G分别代表前边提到的L.fst和G.fst。H 则代表HMM相关的内容，C代表音素级别的上下文，如果读者对这个文件的内容感兴趣可以通过查看Kaldi 的官方文档以获取关于这个文件的细节描述性 [4]。

用于生成HCLG.fst的脚本为

```
utils/mkgraph.sh - - mono data/graph/lang exp/mono mono/graph
```

每个参数的意义为:

- - mono	是否使用mono生成解码图
data/graph/lang	G.fst与L.fst所在的目录
exp/mono	tree所在的目录
mono/graph	HCLG.fst 将被写在这个目录下

decode的脚本为:

```
steps/decode.sh - - cmd "$decode_cmd" mono/graph data/mfcc/test
exp/mono/decode
```

每个参数的意义为：

- - cmd	与前文介绍的cmd 的意义相同
mono/graph	HCLG.fst 所在的目录
data/mfcc/test	测试数据所在的目录
exp/mono/decode	最后的结果将被写在这个目录下

解码结束后，在exp/mono/decode 目录下将会生成很多类似wer_10.0.0这样的文件，这些文件代表识别的错误率（wer word error rate），我们可以使用“grep wer* | kaldi/egs/thchs30/utils/best_wer.sh” 查看最低的错误率可以达到多少。

4.6 GMM标注

标注的任务是使用当前的模型对数据进行标注，得到每一帧对应的音素（实际上Kaldi是将数据标注为帧-transition id,并不是音素为了便于初学者理解，我们将它写为帧-音素），标注的脚本为

```
steps/align_si.sh - -boost-silence 1.25 –nj $n - -cmd ”$train_cmd”
data/mfcc/train  data/lang  exp/mono exp/mono_align
```

每个参数的意义为（前文介绍过的参数没有在这里重复列出）:

data/mfcc/train	这个脚本所需要的输入数据所在的目录
data/lang	L.fst所在的目录
exp/mono	final.mdl 所在的目录
exp/mono_align	标注结果将被写在这个目录下

标注结束后在exp/mono_align目录下会生成很多类似ali.1.gz的文件。对于这些标注文件Kaldi 提供一些系列工具查看每一帧对应的音素和pdf 等 [4]：

ali-to-phones final.mdl ark:1.ali ark,t:file	查看每一帧对应的音素序号
ali-to-pdf final.mdl ark:1.ali ark,t:file	查看每一帧对应的pdf
copy-int-vector ark:1.ali ark,t:file	查看每一帧对应的transition-id

4.7 DNN模型训练及解码

在这一章节中，我们将以TDNN（time delay neural network,kaldi/egs/wsj/s5/local/nnet3/run_tdnn.sh）为模板向读者介绍Kaldi 如何实现一个DNN的声学模型模型。

```
steps/nnet3/train_tdnn.sh - - parameter data/mfcc/train data/lang
exp/tri4b_ali  exp/nnet3/tdnn
```

每个参数的意义为：

- -stage	训练状态数
- -num-epochs	epochs的数量，iteration将会由这个参数生成
- - splice-indexes	TDNN的结构由这个参数生成
- -feat-type	feat的种类
- -initial-effective-lrate	起始时的学习率
- -final-effective-lrage	结束时的学习率
- -cmd	与前文介绍的cmd意义相同
- -pnorm-input-dim	pnorm 的输入维度（pnorm是一种激活函数，Kaldi还提供了其他种类的激活函数比如：Tanh，Sigmoid，Relu等）
- -pnorm-output-dim	pnorm 的输出维度
data/train	训练数据所在的目录
data/lang	L.fst所在的目录
exp/tri4b_ali/	标注数据所在的目录
exp/nnet3/tdnn	最终模型将被写在这个目录下

对于初学者来说参数列表中的参数基本不需要自己手动修改，直接运行这个脚本然后等待结果即可，但是很多时候我们的程序会因为一些非脚本内部逻辑错误而中断，比如说硬件死机等等，这时候通过修改- -stage这个参数可以迅速的从程序中断的状态恢复运行。Kaldi在训练模型的时候每训练一轮就会在存放结果的目录下保存一个当前的模型，一般被命名为iteration.mdl。iteration 就代表当前模型训练到了多少轮，所以在我们的程序因为一些因素中断后，我们可以将- -stage的设为iteration的值，程序就可以从中断的状态继续训练，而不是从头开始，可以节约很多时间成本。

DNN的解码和普通GMM-HMM模型的解码过程类似，我们所讲述的TDNN是kaldi中nnet3实现的模型，所以解码的时候需要调用：

```
steps/nnet3/decode.sh - -nj $n  - -cmd "$decode_cmd"  $graph_dir
data/fbank/test  exp/nnet3/nnet_tdnn/decode
```

每个参数的意义为：

- -nj	与前文介绍的nj意义相同
- -cmd	与前文介绍的cmd意义相同
graph_dir	HCLG.fst 所在的目录
data/fbank/test	测试集所在的目录
exp/nnet3/nnet_tdnn/decode	解码生成的文件将被写在这个目录下。

5 总结

本文所介绍的哈语的语音识别的模型，只是最初步的语音识别模型，在数据处理方面其实还存在一些不是很合理地方，未来还会为提高哈语识别率做其他改进工作，另外这篇文章的受众群体为刚开始接触语音识别和Kaldi 的读者，对于语音识别和Kaldi更细节的知识与技术，还需要读者在熟悉Kaldi 的流程后多做一些探索与实践。

6 推荐读物

http://kaldi-asr.org/doc/
https://www.inf.ed.ac.uk/teaching/courses/asr/
http://deeplearning.net/reading-list/
Automatic Speech Recognition A Deep Learning Approcach By Dong Yu and Li Deng
Deep Learning by Yoshua Bengio,Ian Goodfellow,Aaron Courville

Author details

[1]Center for Speech and Language Technologies, Research Institute of Information Technology, Tsinghua University, ROOM 1-303, BLDG FIT, 100084 Beijing, China. [2]Center for Speech and Language Technologies, Division of Technical Innovation and Development, Tsinghua National Laboratory for Information Science and Technology, ROOM 1-303, BLDG FIT, 100084 Beijing, China. [3]Chengdu Institute of Computer Applications, Chinese Academy of Sciences, 610041 Chengdu, China.

References

1. Yoshua Bengio, Aaron Courville, and Pascal Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.
2. wiki, "ngram," `https://en.wikipedia.org/wiki/N-gram`.
3. Hiroshi Shimodaira and Steve Renals, "Gmm," `https://www.inf.ed.ac.uk/teaching/courses/asr/2015-16/asr03-hmmgmm-handout-nup.pdf`.
4. kaldi, "Kaldi," `http://kaldi-asr.org/doc/`.