MTTS Documentation

MTTS

2022 年 04 月 18 日

Contents

Mandarin/Chinese Text to Speech based on statistical parametric speech synthesis using merlin toolkit

Latest update time: 2018-03-01

Author: Jackiexiao

Wechat: explorerrr

CHAPTER 1

Merlin 简要使用手册

Merlin 的简要介绍 Merlin 不是一个完整的 TTS 系统，它只是提供了 TTS 核心的声学建模模块（声学和语音特征归一化，神经网络声学模型训练和生成）。

前端文本处理 (frontend) 和声码器（vocoder) 需要其他软件辅助。

frontend:

- festival
- festvox
- hts
- htk

vocoder:

- WORLD
- SPTK
- MagPhase

1.1 1 Merlin 的安装

安装

Merlin 只能在 unix 类系统下运行，使用 Python，并用 theano 作为后端

Merlin 的 Python 语言采用的是 Python2.7 编写（更新：merlin 已经支持 python2.7-3.6 的版本），所以我们需要在 Python2.7 的环境下运行 Merlin，为避免 python 不同版本之间的冲突，我们采用 Anaconda 对 Python 运行环境进行管理。

Anaconda **'国内镜像下载地址 <https://mirrors.tuna.tsinghua.edu.cn/help/anaconda/>'_** 下载完毕后

```
bash Anacoda......sh
```

使用 Anaconda 创建 Merlin 运行环境具体操作如下：

打开终端，使用下面命令查看一下现有 python 环境

```
conda env list
```

使用下面命令创建一个名为 merlin 的 python 环境

```
conda create --name merlin python=2.7
```

先进入 merlin 环境中

```
source activate merlin
```

在这个环境下安装 merlin

```
sudo apt-get install csh cmake realpath autotools-dev automake
pip install numpy scipy matplotlib lxml theano bandmat
git clone https://github.com/CSTR-Edinburgh/merlin.git
cd merlin/tools
./compile_tools.sh
```

留意程序的输出结果，一些程序如果没有成功安装会影响到后面的结果

如果一切顺利，此时你已经成功地安装了 Merlin，但要注意的是 Merlin 不是一个完整的 TTS 系统。它提供了核心的声学建模功能：语言特征矢量化，声学和语言特征归一化，神经网络声学模型训练和生成。但语音合成的前端（文本处理器）以及声码器需要另外配置安装。此外，Merlin 目前仅提供了英文的语音合成。

此外，上述安装默认只配置支持 CPU 的 theano，如果想要用 GPU 加速神经网络的训练，还需要进行其他的步骤。由于语料库的训练时间尚在笔者的接受范围之内（intel-i5，训练 slt_arctic_full data 需要大概 6 个小时），因此这里并没有使用 GPU 进行加速训练。

运行 Merlin demo

```
sudo bash  ~/merlin/egs/slt_arctic/s1/run_demo.sh
```

该脚本会使用 50 个音频样本进行声学模型和 durarion 模型的训练，并合成 5 个示例音频。在此略去详细的操作步骤，具体可参见：Getting started with the Merlin Speech Synthesis Toolkit installing-Merlin

1.2 2 Merlin 源码理解

1.2.1 0 文件含义

Folder	Contains
recordings	speech recordings, copied from the studio
wav	individual wav files for each utterance
pm	pitch marks
mfcc	MFCCs for use in automatic alignment mfcc tutorial
lab	label files from automatic alignment
utt	Festival utterance structures
f0	Pitch contours
coef	MFCCs + f0, for the join cost
coef2	coef2, but stripped of unnecessary frames to save space, for the join cost
lpc	LPCs and residuals, for waveform generation
bap	band aperiodicity

1.2.2 1 免费的语料库

Merlin 使用了网络上免费的语料库 slt_arctic，可以在以下网址进行下载：slt_arctic_full_data.zip

1.2.3 2 训练数据的处理

Merlin 自带的 demo（merlin/egs/slt_arctic/s1 ）已经事先完成了 label 文件以及声学参数 mgc lf0 bap 的提取，所以这里不需要前端 FrontEnd 和声码器对数据进行处理。

Merlin 通过脚本文件 setup.sh 在 ~/merlin/egs/slt_arctic/s1 目录下创建目录 experiments，在 experiments 目录下创建目录 slt_arctic_demo，完成数据的下载与解压，并将解压后的数据分别放到 slt_arctic_demo/acoustic_mode/data，slt_arctic_demo/duration_model/data 目录下，分别用于声学模型和持续时间模型的训练。

1.2.4 3 Demo 语料库的训练

run_demo.sh 文件会进行语音的训练以及合成。这里有许多的工程实现细节，在这里略去说明，其主要进行了如下步骤![img](/img/image5.png)

其中语料库包含了文本和音频文件，文本需要首先通过前端 FrontEnd 处理成神经网络可接受的数据，这一步比较繁琐，不同语言也各不相同，下面会着重讲解。音频文件则通过声码器（这里使用的是 STRAIGHT 声码器）转换成声码器参数（包括了 mfcc 梅谱倒谱系数，f0 基频，bap：band aperiodicity 等）再参与到神经网络的训练之中。

1.2.5 4 Demo 语料库的合成

Demo 中提供了简单的合成方法，使用 demo（merlin/egs/slt_arctic/s1 ）下的脚本文件：merlin_synthesis.sh 即可进行特定文本的语音合成。

同样的，由于 merlin 没有自带 frontend，所以其 demo 中直接使用了事先经过 frontend 转换的 label 文件作为输入数据来合成语音。如果想要直接输入 txt 文本来获得语音，需要安装 FrontEnd（下文会提及）并根据 merlin_synthesis.sh 文件的提示用 FrontEnd 来转换 txt 文本成 label 文件，再进行语音合成。

对于英文语音合成，merlin 中需要首先通过 Duration 模型确定音素的发音时间，然后根据声学模型合成完整的语音。

1.2.6 5.Merlin 的训练网络

Merlin 的训练网络可见 *Merlin: An Open Source Neural Network Speech Synthesis System *

Merlin 一共提供了 4 类神经网络用于 HMM 模型的训练，分别是

- 前馈神经网络
- 基于 LSTM 的 RNN 网络
- 双向 RNN 网络
- 其他变体（如 blstm）

1.3 3 Merlin 英文前端

Merlin 前端 FrontEnd

（1）Label 的分类

在 Merlin 中，Label 有两种类别，分别是

- **state align**（使用 HTK 来生成，以发音状态为单位的 label 文件，一个音素由几个发音状态组成）
- **phoneme align**（使用 Festvox 来生成，以音素为单位的 label 文件）

（2）txt to utt

文本到文本规范标注文件是非常重要的一步，这涉及自然语言处理，对于英文来说，具体工程实现可使用 Festival，参见：Creating .utt Files for English

Festival 使用了英文词典，语言规范等文件，用最新的 EHMM alignment 工具将 txt 转换成包含了文本特征（如上下文，韵律等信息）的 utt 文件

（3）utt to label

在获得 utt 的基础上，需要对每个音素的上下文信息，韵律信息进行更为细致的整理，对于英文的工程实现可参见：Creating Label Files for Training Data

label 文件的格式请参见：lab_format.pdf

(4) label to training-data(Question file)

The questions in the question file will be used to convert the full-context labels into binary and/or numerical features for vectorization. It is suggested to do a manual selection of the questions, as the number of questions will affect the dimensionality of the vectorized input features.

在 Merlin 目录下，merlin/misc/questions 目录下，有两个不同的文件，分别是：

- questions-radio_dnn_416.hed
- questions-unilex_dnn_600.hed

查看这两个文件，我们不难发现，questions-radio_dnn_416.hed 定义了一个 416 维度的向量，向量各个维度上的值由 label 文件来确定，也即是说，从 label 文件上提取必要的信息，我们可以很轻易的按照定义确定 Merlin 训练数据 training-data；同理 questions-unilex_dnn_600.hed 确定了一个 600 维度的向量，各个维度上的值依旧是由 label 文件加以确定。

1.4 4 Merlin vocoder 声码器

Merlin 中自带的 vocoder 工具有以下三类：Straight，World，World_v2

这三类工具可以在 Merlin 的文件目录下找到，具体的路径如下 merlin/misc/scripts/vocoder

在介绍三类 vocoder 之前，首先说明几个概念：

MGC 特征 通过语音提取的 MFCC 特征由于维度太高，并不适合直接放到网络上进行训练，所以就出现了 MGC 特征，将提取到的 MFCC 特征降维（在这三个声码器中 MFCC 都被统一将低到 60 维），以这 60 维度的数据进行训练就形成了我们所说的 MGC 特征

BAP 特征 Band Aperiodicity 的缩写

LF0 LF0 是语音的基频特征

Straight

音频文件通过 Straight 声码器产生的是：60 维的 MGC 特征，25 维的 BAP 特征，以及 1 维的 LF0 特征。

通过 STRAIGHT 合成器提取的谱参数具有独特特征 (维数较高), 所以它不能直接用于 HTS 系统中, 需要使用 SPTK 工具将其特征参数降维, 转换为 HTS 训练中可用的 mgc(Mel-generalized cepstral) 参数, 即, 就是由 STRAIGHT 频谱计算得到 mgc 频谱参数, 最后利用原 STRAIGHT 合成器进行语音合成

World

音频文件通过 World 声码器产生的是：60 维的 MGC 特征，可变维度的 BAP 特征以及 1 维的 LF0 特征，对于 16kHz 采样的音频信号，BAP 的维度为 1，对于 48kHz 采样的音频信号，BAP 的维度为 5

网址为：github.com/mmorise/World

1.5 5 生成 Merlin 的英文 label 用于语音合成

注意到 merlin 是没有自带 frontend 的，对于英文，你需要安装 Festival 来将文本转换成 HTS label, 对于其他语言，你需要自行设计或者找到支持的 frontend，中文目前网络上还没有开源的工具，所以你需要自己设计

英文 FrontEnd 安装具体步骤如下参见：Create_your_own_label_Using_Festival.md

安装完毕之后，参考 merlin/tools/alignment 里面的文档生成自己的英文 label

CHAPTER 2

Merlin 源码详解

关于 merlin 的详细解读（强烈推荐），可参考 candlewill 的 [github gist](https://gist.github.com/candlewill/5584911728260904414b4a6679a93d53)

2.1 网络配置文件详解

训练时长模型需要一个配置文件（后续的声学模型也一样）。一般而言，在一个样例配置文件上做一些修改即可。例如，训练 DNN 模型所用的样例配置文件为 [duration_demo.conf](https://github.com/CSTR-Edinburgh/merlin/blob/master/misc/recipes/duration_demo.conf)。

Merlin 称这些不同的样例配置文件为 recipes，全部 recipes 可见：https://github.com/CSTR-Edinburgh/merlin/tree/master/misc/recipes 。

配置文件，主要包含路径信息、对齐方式、问题集名称、模型结构、数据划分、执行过程等信息。

2.1.1 run_merlin.py

程序执行入口，路径为：https://github.com/CSTR-Edinburgh/merlin/blob/master/src/run_merlin.py

2.1.2 执行过程

按照配置文件中不同的 sub-processes，将会有不同的执行方式。

上述各个参数默认取值都为 'False'，因此配置文件中只需要设置取值为 'True'的参数即可。

训练时长模型，训练声学模型，测试时长模型，测试声学模型对应的配置文件，指定的执行流程，分别如下所示：

训练时长模型

```
NORMLAB  : True
MAKEDUR  : True
MAKECMP  : True
NORMCMP  : True

TRAINDNN : True
DNNGEN   : True

CALMCD   : True
```

训练声学模型

```
NORMLAB  : True
MAKECMP  : True
NORMCMP  : True

TRAINDNN : True
DNNGEN   : True

GENWAV   : True
CALMCD   : True
```

测试时长模型

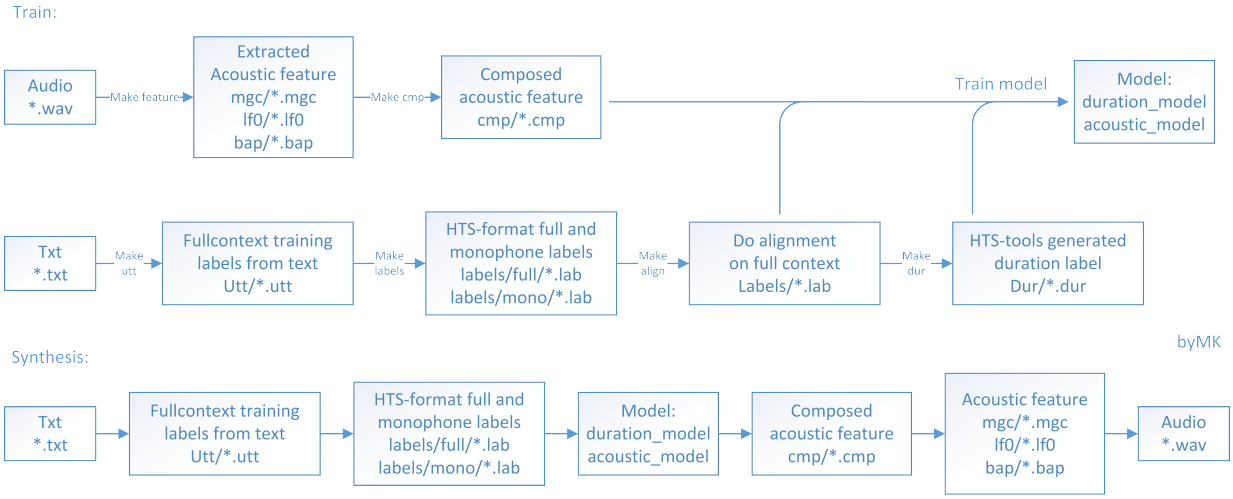
```
NORMLAB: True
DNNGEN: True
```

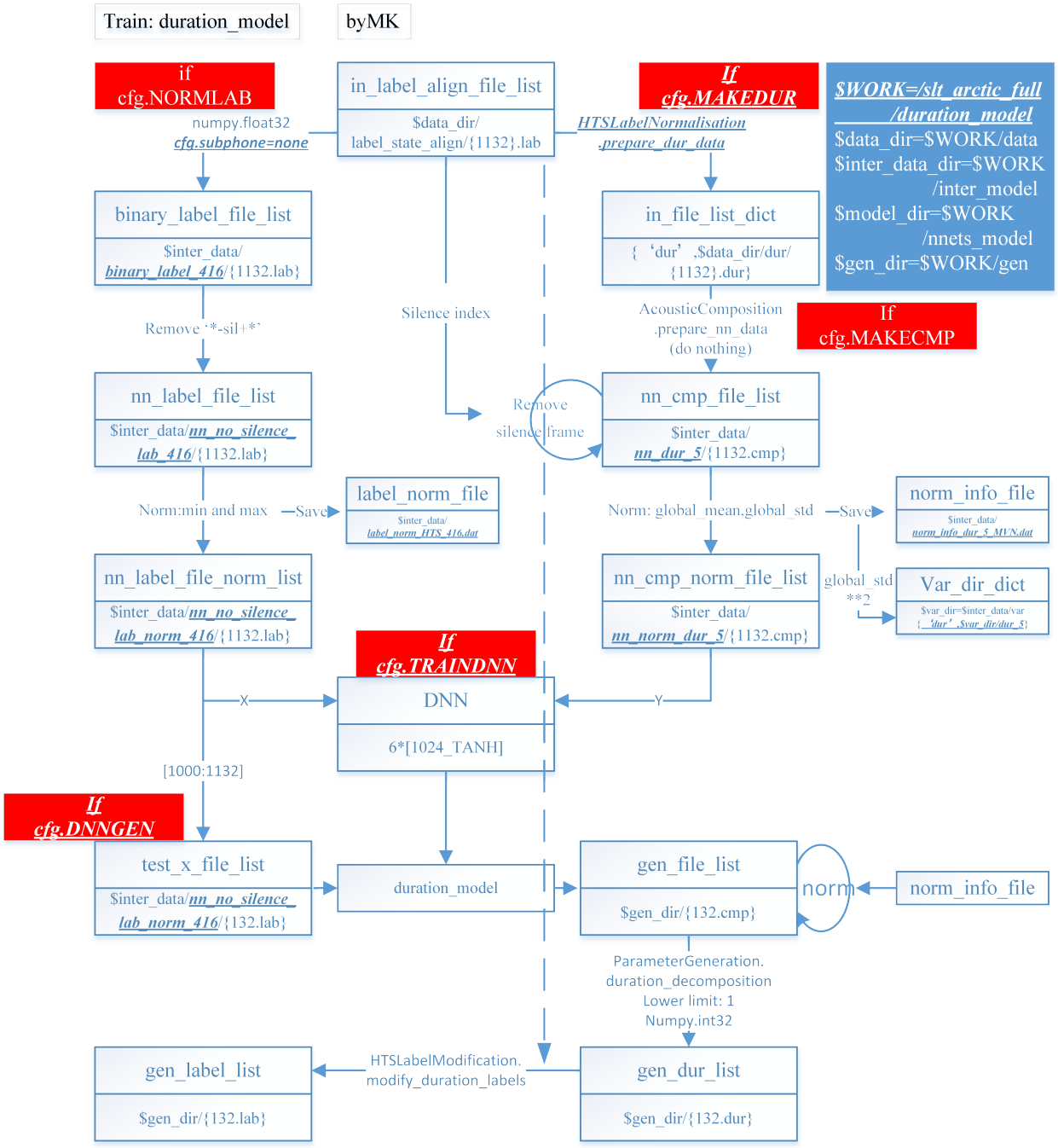
测试声学模型

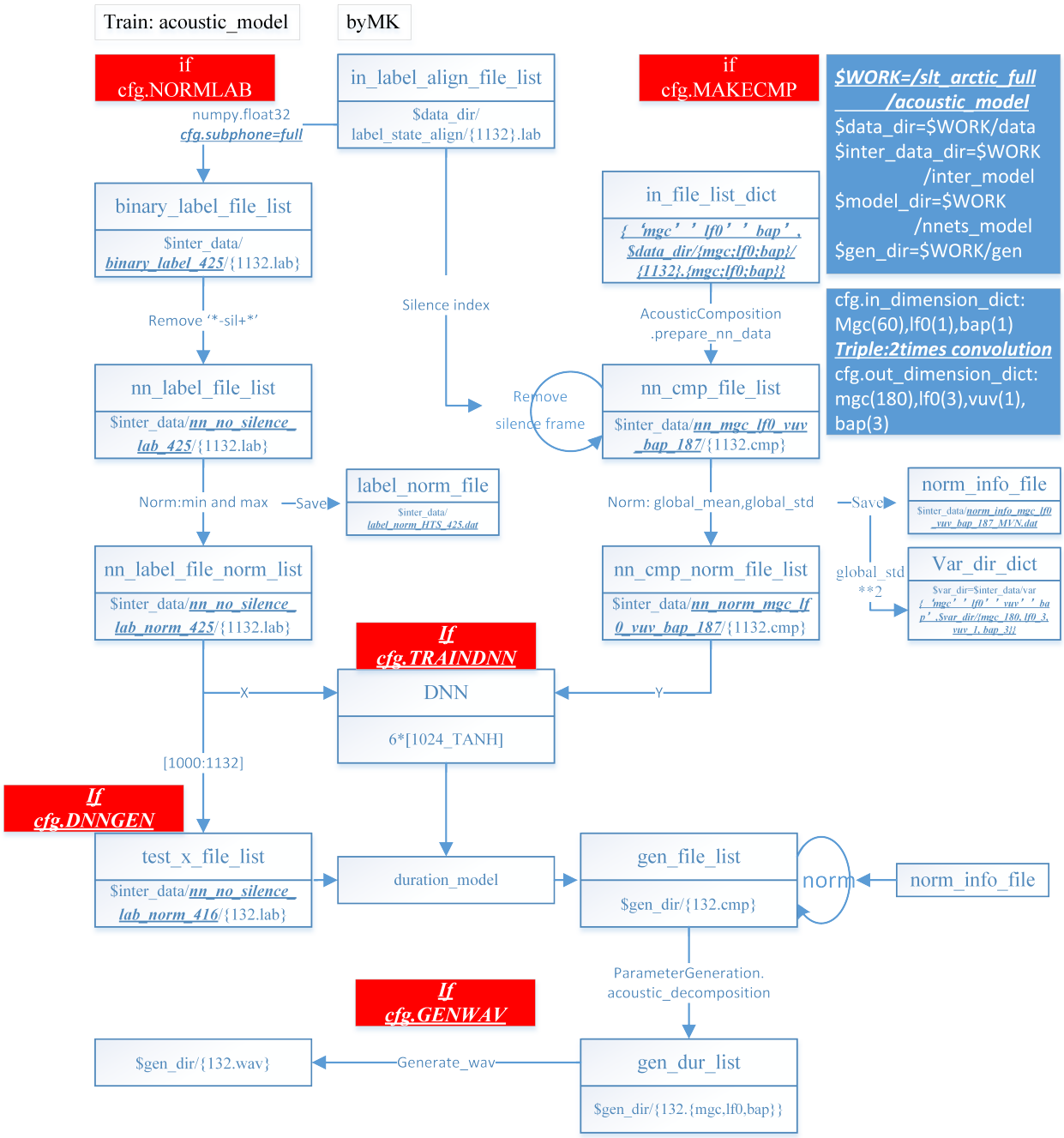
```
NORMLAB  : True
DNNGEN   : True

GENWAV   : True
```

2.2 Merlin 流程图

整体流程图和时长模型 & 声学模型


Train:
Audio
*.wav
Make feature
Extracted
Acoustic feature
mgc/*.mgc
lf0/*.lf0
bap/*.bap
Make cmp
Composed
acoustic feature
cmp/*.cmp
Train model
Model:
duration_model
acoustic_model
Txt
*.txt
Make
utt
Fullcontext training
labels from text
Utt/*.utt
Make
labels
HTS-format full and
monophone labels
labels/full/*.lab
labels/mono/*.lab
Make
align
Do alignment
on full context
Labels/*.lab
Make
dur
HTS-tools generated
duration label
Dur/*.dur
byMK
Synthesis:
Txt
*.txt
Fullcontext training
labels from text
Utt/*.utt
HTS-format full and
monophone labels
labels/full/*.lab
labels/mono/*.lab
Model:
duration_model
acoustic_model
Composed
acoustic feature
cmp/*.cmp
Acoustic feature
mgc/*.mgc
lf0/*.lf0
bap/*.bap
Audio
*.wav


Train: duration_model
byMK
if cfg.NORMLAB
numpy.float32
cfg.subphone=none
in_label_align_file_list
$data_dir/label_state_align/{1132}.lab
If cfg.MAKEDUR
HTSLabelNormalisation .prepare_dur_data
$WORK=/slt_arctic_full /duration_model
$data_dir=$WORK/data
$inter_data_dir=$WORK /inter_model
$model_dir=$WORK /nnets_model
$gen_dir=$WORK/gen
binary_label_file_list
$inter_data/binary_label_416/{1132.lab}
in_file_list_dict
{'dur',$data_dir/dur/{1132}.dur}
Remove '*-sil+*'
Silence index
AcousticComposition .prepare_nn_data (do nothing)
If cfg.MAKECMP
nn_label_file_list
$inter_data/nn_no_silence_lab_416/{1132.lab}
Remove silence frame
nn_cmp_file_list
$inter_data/nn_dur_5/{1132.cmp}
Norm:min and max
Save
label_norm_file
$inter_data/label_norm_HTS_416.dat
Norm: global_mean,global_std
Save
norm_info_file
$inter_data/norm_info_dur_5_MVN.dat
nn_label_file_norm_list
$inter_data/nn_no_silence_lab_norm_416/{1132.lab}
nn_cmp_norm_file_list
$inter_data/nn_norm_dur_5/{1132.cmp}
global_std **2
Var_dir_dict
$var_dir=$inter_data/var
{'dur',$var_dir/dur_5}
If cfg.TRAINDNN
X
Y
DNN
6*[1024_TANH]
[1000:1132]
If cfg.DNNGEN
test_x_file_list
$inter_data/nn_no_silence_lab_norm_416/{132.lab}
duration_model
gen_file_list
$gen_dir/{132.cmp}
norm
norm_info_file
ParameterGeneration. duration_decomposition
Lower limit: 1
Numpy.int32
gen_label_list
$gen_dir/{132.lab}
HTSLabelModification. modify_duration_labels
gen_dur_list
$gen_dir/{132.dur}


Train: acoustic_model
byMK
if cfg.NORMLAB
in_label_align_file_list
$data_dir/ label_state_align/{1132}.lab
if cfg.MAKECMP
$WORK=/slt_arctic_full /acoustic_model
$data_dir=$WORK/data
$inter_data_dir=$WORK /inter_model
$model_dir=$WORK /nnets_model
$gen_dir=$WORK/gen
numpy.float32
cfg.subphone=full
binary_label_file_list
$inter_data/ binary_label_425/{1132.lab}
in_file_list_dict
{'mgc' 'lf0' 'bap', $data_dir/{mgc:lf0:bap}/ {1132}.{mgc:lf0:bap}}
cfg.in_dimension_dict:
Mgc(60),lf0(1),bap(1)
Triple:2times convolution
cfg.out_dimension_dict:
mgc(180),lf0(3),vuv(1),
bap(3)
Silence index
Remove '*-sil+*'
AcousticComposition .prepare_nn_data
nn_label_file_list
$inter_data/nn_no_silence_ lab_425/{1132.lab}
Remove silence frame
nn_cmp_file_list
$inter_data/nn_mgc_lf0_vuv _bap_187/{1132.cmp}
Norm:min and max
Save
label_norm_file
$inter_data/ label_norm_HTS_425.dat
Norm: global_mean,global_std
Save
norm_info_file
$inter_data/norm_info_mgc_lf0 _vuv_bap_187_MVN.dat
nn_label_file_norm_list
$inter_data/nn_no_silence_ lab_norm_425/{1132.lab}
nn_cmp_norm_file_list
$inter_data/nn_norm_mgc_lf 0_vuv_bap_187/{1132.cmp}
global_std
**2
Var_dir_dict
$var_dir=$inter_data/var {'mgc' 'lf0' 'vuv' 'bap',$var_dir/{mgc_180, lf0_3, vuv_1, bap_3}}
If cfg.TRAINDNN
DNN
6*[1024_TANH]
X
Y
[1000:1132]
If cfg.DNNGEN
test_x_file_list
$inter_data/nn_no_silence_ lab_norm_416/{132.lab}
duration_model
gen_file_list
$gen_dir/{132.cmp}
norm
norm_info_file
ParameterGeneration. acoustic_decomposition
If cfg.GENWAV
$gen_dir/{132.wav}
Generate_wav
gen_dur_list
$gen_dir/{132.{mgc,lf0,bap}}

CHAPTER 3

语料库

3.1 1 King_tts_003 语料库

3.1.1 1.1 文本与标注

语料库使用购买自海天瑞声公司的 King_tts_003 语料库，该语料库由专业的标准女播音员录制，总时长为 15 小时，近 2 万个句子。其中标注包括了拼音、韵律（韵律边界与重音）、音素发音时长、声韵母标注。语音文件以 44.1 KHz，16bit，双音道，windows 的无压缩 PCM 格式存储。除了此外，该语料库还提供了记录 EGG（electroglottography）信号的音频。

3.1.2 声调的标记格式

采用数字 1、2、3、4、5, 代替《汉语拼音方案》中声调阴平（ ），阳平（ ），上声（ˇ），去声（ ），轻声（不标调）这几个标调符号

3.1.3 韵律的标记格式

韵律分成四级，分别用 #4，#3，#2，#1 表示。

#4： （1）一个完整语意的句子，切除前后可以独立成为一个句子，从听感上调形是完全降下来的，有明显的停顿。(2）如果是以二声词结尾的短句，这个二声的词被拖长音，且与后面是转折的关系的，有明显的停顿。

#3： 通常标在一个韵律短语后面，有时会是一个词，从听感上调形是降下来的，但不够完全，不能独立成为一个语意完整的句子。

#2： （1）表示被‘重读’的词或单个字 (为了强调后面)，有停顿，调形上有小的变化，有‘骤停’的感觉。(对于单音节词如果是被‘拖长音’，给 #1；如果是‘骤停’要给 #2 ）(2）并列关系的词如果被强调重读，给 #2；如果是很平滑的，给 #1。

#1： 只是韵律词的边界，通常没有停顿

3.1.4 声韵母与停顿的标记格式

标注符号采用 a，b，d，s 四种标记符号进行标注，标注符号的意思如下：

- a 表示中文汉字的声母。
- b 表示中文汉字的韵母。
- d 表示句中的静音长度小于 100ms 的停顿。
- s 表示句子的起始点和结束点以及句中大于 100ms 的停顿。

3.1.5 声韵标注的具体规则

1. 中文汉字拼音的声母用 a 表示，韵母用 b 表示。
2. 其中有一些汉字音节以元音开头，称为零声母音节，如 a/o/e/ang/eng/en/ai/ei/ao/ou/an/er/，我们用标记点 a 来进行标注。
3. 其中有一些汉字是特殊读音，仅仅表示鼻子发出的气流，如 m/n/ng/，分别对应汉字（㐄，嗯，嗯），我们用标记点 b 来进行标注。
4. 汉字发音为 yu/yi/wu/的为整体认读音节，但我们此次把以 w，y 为声母加韵母的拼音按照声韵进行切分。

举一个例子

```
我#1 就怕#2 自己的#1 俗气#3 亵渎了#2 普者黑的#1 风景

wo3 jiu4 pa4 zi4 ji3 de5 su2 qi4 xie4 du2 le5 pu2 zhe3 hei1 de5 feng1 jing3
```

3.2 2 语料库的获取

3.2.1 中文语料库

目前网络上尚未有免费的语料库，需要自行构建语料库或者是购买公司或者大学的语料库。对于自行构建语料库来说，除了韵律部分需要人工标记之外，其他部分实际上可以通过工具实现标注，例如字音转换，分词，

词性标注，音素发音时长。

3.2.2 英文语料库

ARCTIC 数据库是由卡内基梅隆大学（CMU）语言技术研究所开发的英文朗读语音数据库 [51]。最初是用来训练基于单元挑选（Unit selection）的语音合成系统，后来成为基于 HSS 英文系统训练的通用数据库。该库包含 1132 句文本，选自文化作品数字化项目“古腾堡计划”中的两个英文短篇故事 [52]。共有 7 名说话人参与录音，其中 BDL、SLT、CLB 和 RMS 四位说话人使用美式英语，其余三位是有加拿大、苏格兰及印度口音的英语母语说话人。为便于评测，本论文中的使用四位美式英语说话人的数据训练英文的声音模型。ARCTIC 中另外一位说话人 Roger 也是美式英语，与其他人的数据独立发行，大量用于 Blizzard Challenge 测试中。ARCTIC 数据库的上下文标注数据由开源语音合成系统 Festival[53] 的前端文本分析模块得到，使用由名古屋工业大学 Tokuda 实验室发布的以音素为单位的标准英文上下文标注格式。

CMU_ARCTIC speech synthesis system [OL] .http://festvox.org/cmu_arctic/index.html .2012

EMA 数据库 [54] 是由南加州大学 SAIL 实验室开发的使用电磁设备记录发声器官数据的英文情感语音数据库。共包含 2 名女性（JN、LS）和 1 名男性（AB）美式英语母语说话人。录音文本共包含 10 句英文语料，每名说话人用每种情感将一句语料重复录制 5 次。然后通过一批测听人员对每条录音情感表现的打分，根据平均意见，决定哪些录音是有效的情感语音数据。这个数据库包含语音波形和同步录制的舌位动态信号，本身是用于研究情感语音的声学表现以及发声器官协同建模的小规模音库。在本文中，将用于学习中性语音到情感语音转换的声学参数变换规则。由于 EMA 只是用于进行声学分析，没有提供标注。因此，使用 Festival 的前端文本分析模块得到了与 ARCTIC 相同格式的英文上下文标注。

对于端到端的语料库，可具体参见 github tacotron 的复现项目，各自使用了不同的语料库

CHAPTER 4

文本分析

4.1 1 拼音标注风格

读此章节时读者有必要回顾拼音的基础知识

- 中华人民共和国教育部发布的 汉语拼音方案
- 整体认读音节

整体认读音节 16 个整体认读音节分别是：zhi 、chi、shi、ri、zi、ci、si、yi、wu、yu、ye、yue、yuan、yin 、yun、ying，但是要注意没有 yan，因为 yan 并不发作 an 音

声母 21 个声母没有什么争议，如果说有 22 个声母，一般指多加一个零声母，yw 都属于零声母。如果用 23 个声母，则是 21 声母 +yw 两个零声母，如果用 27 个声母，则是将不同情况下的 yw 零声母分成 6 种情况，标注成 aa, ee, ii, oo, uu, vv，即 21+6=27 个声母（具体见 hmm 训练，合成基元的选择一节）

韵母 国家汉语拼音方案中韵母数量为 35 个，但另一说为 39 个（如百度百科），在原国家汉语拼音方案上增加了-i（前）、-i（后）、er、ê。

下面新加的 4 个元音做简要解释

- ê[ϵ] 在普通话中，ê 只在语气词 “🈂” 中单用【因此一些项目忽略了这个单韵母，即 38 个韵母】。ê 不与任何辅音声母相拼，只构成复韵母 ie、üe，并在书写时省去上面的附加符号 “^”。
- er[] 是在 [ ] 的基础上加上卷舌动作而成。发音例词：而且 érqiě 儿歌 érgē 耳朵 ěrduō 二胡 èrhú 二十 èrshí 儿童 értóng
- -i(前) 指 zi/ci/si 中的 i 发音例词：私自 sīzì 此次 cǐcì 次子 cìzǐ 字词 zìcí 自私 zìsī 孜孜 zīzī

- -i(后) 指 zhi/chi/shi/ri 中的 i 发音例词：实施 shíshī 支持 zhīchí 知识 zhīshi 制止 zhìzhǐ 值日 zhírì 试制 shìzhì

拼音标注风格分成两类，

1. 第一类是国家规定的方案，也就是日常生活中用到的风格，规定了声母 21 个，其韵母表中列出 35 个韵母，具体参见中华人民共和国教育部发布的 汉语拼音方案

2. 第二类是方便系统处理的拼音标注风格，不同项目有不同的注音风格，区别主要在于

- 对 y w 的处理，有的项目为了方便处理，也将 yw 视为声母，有的则会将对应的 yw 传换成实际发音，如 ye,yan,yang（整体认读音节）等改成 ie,ian,iang，而不适用 yw
- 是否将整体认读音节还原成单个韵母或声母
- ju qu xu 的标注是否转为实际发音标注，即 jv qv xv
- yuan yue yun 的标注是否转成 yvan yve yvn
- 注意到 iou, uei, uen 前面加声母时，写成 iu ui un，例如牛 (niu), 归 (gui)，论 (lun)，标注时是否还原成 niou, guei, luen 的问题
- 儿化音是否简化标注，例如’花儿’，汉语拼音方案中标注为’huar’，一般我们将其转为’hua er’

本项目使用的风格

- 将 yw 视作声母，但同时将 ya 还原成 yia, ye 还原成 yie, 其余类似
- 标注为 jv qv xv
- 标注为 yvan yve yvn
- 将 iou, uei, uen 标注还原
- ê 标注为 ee, er(包括儿化音中的 r) 标注为 er, i(前) 标注为 ic, i(后) 标记为 ih
- 声调标注，轻声标注为 5，其他标注为 1234

最终使用的声韵母表如下

声母（23 个） b p m f d t n l g k h j q x zh ch sh r z c s y w

韵母（39 个）

- 单韵母 a、o、e、ê、i、u、ü、-i（前）、-i（后）、er
- 复韵母 ai、ei、ao、ou、ia、ie、ua、uo、üe、iao 、iou、uai、uei
- 鼻韵母 an、ian、uan、üan 、en、in、uen、ün 、ang、iang、uang、eng、ing、ueng、ong、iong

韵母（39 个）（转换标注后）

- 单韵母 a、o、e、ea、i、u、v、ic、ih、er
- 复韵母 ai、ei、ao、ou、ia、ie、ua、uo、ve、iao 、iou、uai、uei
- 鼻韵母 an、ian、uan、van 、en、in、uen、vn 、ang、iang、uang、eng、ing、ueng、ong、iong

注意：* pypinyin 中使用的是 yuan ju lun * 本文语料库使用的是 yvan jv lun，语料库中音素标注将 yw 视作声母

另外一种推荐的方案是使用 27 个声母，即去掉 yw

声母（27 个） b p m f d t n l g k h j q x zh ch sh r z c s aa ee ii oo uu vv

4.2 2 多音字的处理

本项目使用了 pypinyin

4.3 3 文本规范化

本项目暂时没有实现此功能

对文本进行预处理，主要是去掉无用字符，全半角字符转化等

有时候普通话文本中会出现简略词、日期、公式、号码等文本信息，这就需要通过文本规范化，对这些文本块进行处理以正

- “小明体重是 128 斤”中的“128”应该规范为“一百二十八”，而“G128 次列车”中的“128”应该规范为“一二八”；
- “2016-05-15”、“2016 年 5 月 15 号”、“2016/05/15”可以统一为一致的发音

对于英文而言，如：

- 2011 NYER twenty eleven
- £100 MONEY one hundred pounds
- IKEA ASWD apply letter-to-sound
- 100 NUM one hundred
- DVD LSEQ D. V. D. dee vee dee

4.4 4 词性标注

本项目使用结巴工具进行词性标注。结巴分词工具包采用和 ictclas 兼容的标记法。由于结巴分词的标准较为简单，本项目使用结巴的词性标注规范，在上下文标注和问题集中只取大类标注，即字母 a-z 所代表的词性，具体见下方列表中给出的结巴词性标注表

词性标注规范

- 结巴使用的词性标注表
- 中科院 ictclas 规范

- 斯坦福 Stanford coreNLP 宾州树库的词性标注规范
- ICTPOS3.0 词性标记集 链接中还包括了 ICTCLAS 汉语词性标注集、jieba 字典中出现的词性、simhash 中可以忽略的部分词性
- 北大标注集

4.5 5 句子语气类型

[todo] 找到能自动标识句子语气类型的工具

句子语气的类型	陈述句	疑问句	祈使句	感叹句
标识符	d	e	i	q

4.6 6 中文分词

本项目使用了结巴分词器，读者可以按自己的需要选择其他分词器，可见 github 项目：中文分词器分词效果评估对比

CHAPTER 5

本项目的设计规则

5.1 1 合成基元

这里选取声韵母作为基元，同时为了模拟发音中的停顿，可以将短时停顿和长时停顿看做是合成基元，此外，将句子开始前和结束时的静音 sil 也当做合成基元

5.1.1 合成基元的列表

本项目选用的合成基元为

- 声母 | 21 个声母 +wy（共 23 个）
- 韵母 | 39 个韵母
- 静音 | sil pau sp

sil(silence) 表示句首和句尾的静音，pau(pause) 表示由逗号，顿号造成的停顿，句中其他的短停顿为 sp(short pause)

声母（23 个） b p m f d t n l g k h j q x zh ch sh r z c s y w

韵母（39 个）

- 单韵母 a、o、e、ê、i、u、ü、-i（前）、-i（后）、er
- 复韵母 ai、ei、ao、ou、ia、ie、ua、uo、üe、iao 、iou、uai、uei
- 鼻韵母 an、ian、uan、üan 、en、in、uen、ün 、ang、iang、uang、eng、ing、ueng、ong、iong

韵母（39 个）（转换标注后）

- 单韵母 a、o、e、ea、i、u、v、ic、ih、er
- 复韵母 ai、ei、ao、ou、ia、ie、ua、uo、ve、iao 、iou、uai、uei
- 鼻韵母 an、ian、uan、van 、en、in、uen、vn 、ang、iang、uang、eng、ing、ueng、ong、iong

5.1.2 其他项目的方法-引入零声母，这里没有采用

6 个零声母的引入是为了减少上下文相关的 tri-IF 数目，这样就可以使得每个音节都是由声母和韵母组成，原先一些只有韵母音节可以被视作是声母和韵母的结构，这样一来，基元就只有声母-韵母-声母以及韵母-声母-韵母两种结构，而不会出现两个韵母相邻的情况，进而明显减少了上下文相关的基元。

如果这么做的话就是 21+6=27 个声母，可以将零声母标记成 aa, ee, ii, oo, uu, vv，一是将 yw 替换，二是将一个韵母

- ye,yan,yang（整体认读音节）标注成——ii ie, ii ian, ii iang （ie, ian, iang 是真实发音的韵母）
- ao an ou 熬安欧, 标记成 aa ao, aa an, oo ou

5.2 2 上下文相关标注与问题集

上下文相关标注的规则要综合考虑有哪些上下文对当前音素发音的影响，总的来说，需要考虑发音基元及其前后基元的信息，以及发音基元所在的音节、词、韵律词、韵律短语、语句相关的信息。

本项目的设计规则参考了 面向汉语统计参数语音合成的标注生成方法

具体规则与示例 * 上下文相关标注 * 问题集设计规则和示例 * 完整问题集文件

问题集 (Question Set) 即是决策树中条件判断的设计。问题集通常很大，由几百个判断条件组成。一个典型的英文问题集文件 (merlin)

问题集的设计依赖于不同语言的语言学知识，而且与上下文标注文件相匹配，改变上下文标注方法也需要相应地改变问题集，对于中文语音合成而言，问题集的设计的规则有:

- 前前个，前个，当前，下个，下下个声韵母分别是某个合成基元吗，合成基元共有 65 个 (23 声母 +39 韵母 +3 静音)，例如判断是否是元音 a QS “LL-a” QS “L-a” QS “C-a” QS “R-a” QS “RR-a”
- 声母特征划分，例如声母可以划分成塞音，擦音，鼻音，唇音等，声母特征划分 24 个
- 韵母特征划分，例如韵母可以划分成单韵母，复合韵母，分别包含 aeiouv 的韵母，韵母特征划分 8 个
- 其他信息划分，词性划分，26 个词性; 声调类型，5 个; 是否是声母或者韵母或者静音，3 个
- 韵律特征划分，如是否是重音，重音和韵律词/短语的位置数量
- 位置和数量特征划分

对于三音素模型而言，对于每个划分的特征，都会产生 3 个判断条件，该音素是否满足条件，它的左音素（声韵母）和右音素（声韵母）是否满足条件，有时会扩展到左左音素和右右音素的情况，这样就有 5 个问

题。其中，每个问题都是以 QS 命令开头，问题集的答案可以有多个，中间以逗号隔开，答案是一个包含通配符的字符串。当问题表达式为真时，该字符串成功匹配标注文件中的某一行标注。格式如：

QS 问题表达式 {答案 1，答案 2，答案 3，……}

QS "LL==Fricative" {f^*,s^*,sh^*,x^*,h^*,lh^*,hy^*,hh^*}

对于 3 音素上下文相关的基元模型的 3 个问题，例如：* 判断当前，前接，后接音素/单元是否为擦音 * QS 'C_Fricative' * QS 'L_Fricative' * QS 'R_Fricative'

更多示例：

Question	含义
QS "C_a"	当前单元是否为韵母 a
QS "L_Fricative"	前接单元是否为擦音
QS "R_Fricative"	后接单元是否为擦音
QS "C_Fricative"	当前单元是否为擦音
QS "C_Stop"	当前单元是否为塞音
QS "C_Nasal"	当前单元是否为鼻音
QS "C_Labial"	当前单元是否为唇音
QS "C_Apieal"	当前单元是否为顶音
QS "C_TypeA"	含有 a 的韵母
QS "C_TypeE"	含有 e 的韵母
QS "C_TypeI"	含有 i 的韵母
QS "C_POS==a"	当前单元是否为形容词
QS "C_Toner==1"	当前单元音调是否为一声

值得注意的是，merlin 中使用的问题集和 HTS 中有所不同，Merlin 中新增加了 CQS 问题，Merlin 处理 Questions Set 的模块在 merlin/src/frontend/label_normalisation 中的 Class HTSLabelNormalisation

Question Set 的格式是 QS + 一个空格 + "question_name" + 任意空格 + {Answer1, answer2, answer3 …} # 无论是 QS 还是 CQS 的 answer 中，前后的 ** 不用加，加了也会被去掉 CQS + 一个空格 + "question_name" + 任意空格 + {Answer} # 对于 CQS, 这里只能有一个 answer 比如 CQS C-Syl-Tone {_(d+)+} merlin 也支持浮点数类型，只需改为 CQS C-Syl-Tone {_([d.]+)+}

CHAPTER 6

术语表

- Front end 前端
- vocoder 声音合成机（声码器）
- MFCC
- 受限波尔曼兹机
- bap band aperiodicity
- ASR：Automatic Speech Recognition 自动语音识别
- AM：声学模型
- LM：语言模型
- HMM：Hiden Markov Model 输出序列用于描述语音的特征向量，状态序列表示相应的文字
- HTS：HMM-based Speech Synthesis System 语音合成工具包
- HTK：Hidden Markov Model Toolkit 语音识别的工具包
- 自编码器
- SPTK：speech signal precessing toolkit
- SPSS：统计参数语音合成 statistical parametric speech synthesis
- pitch 音高：表示声音 (基本) 频率的高低
- Timbre 音色
- Zero Crossing Rate 过零率

- Volume 音量
- sil silence
- syllable 音节
- intonation 声调，语调，抑扬顿挫
- POS part of speech
- mgc
- mcep Mel-Generalized Cepstral Reprfesentation
- mcc mel cepstral coefficents
- mfcc Mel Frequency Cepstral Coefficents
- LSP: Line Spectral Pair 线谱对参数
- **多个音素的 命名规则**
  - monophone 单音素
  - biphone diphone 两音素
  - triphone 三音素
  - quadphone 四音素
- utterance 语音，发声
- 英语韵律符号系统 ToBI(Tone and Break Index)
- CD-DNN-HMM（Context-Dependent DNN-HMM）
- frontend :The part of a TTS system that transforms plain text into a linguistic representation is called a frontend
- .wpa word to phonetic alphabet
- .cmp Composed acoustic features
- .scp system control program
- .mlf master label file
- .pam phonetic alphabets to model
- .mgc mel generalized cepstral feature
- .lf0 log f0 a representation of pitch（音高）音高用基频表示
- .mgc
- .utt .utt files are the linguistic representation of the text that Festival outputs（full context training labels)
- .cfg

- initial && final 声母和韵母

缩略语表 (摘自文献 [5])

- AM Acoustic Model，声学模型
- ACR Absolute Category Rating，绝对等级评定
- ASR Automatic Speech Recognition，自动语音识别
- CART Classification and Regression Tree，分类回归树
- CCR Comparison Category Rating，比较等级评定
- CFHMM Continuous F0，连续基频模型
- CMLLR Constrained Maximum Likelihood Linear Regression，受限最大似然线性回归
- CMOS Comparison Mean Opinion Score，比较平均意见分
- CORC Correlation Coefficient，相关系数
- CR Command-Response，命令响应
- CSMAPLR Constrained Structural Maximum A Posterior Linear Regression，受限结构化最大后验概率线性回归
- DBN Dynamic Bayesian Network，动态贝叶斯网络
- DCR Degradation Category Rating，损伤等级评定
- DCT Discrete Cosine Transform，离散余弦变换
- DMOS Degradation Mean Opinion Score，损伤平均意见分
- ED Emotion Dependent，特定情感
- EM Expectation Maximization，期望最大化
- F0 Fundamental Frequency，基音频率
- GMM Gaussian Mixture Model，高斯混合模型
- GTD Global Tied Distribution，全局绑定分布
- HMM Hidden Markov Model，隐马尔科夫模型
- HNR Harmony Noise Ratio，谐波噪声比
- HSS HMM-based Speech Synthesis，基于 HMM 的语音合成
- HSMM Hidden Semi-Markov Model，隐半马尔科夫模型
- HTK HMM Tool Kit，HMM 工具包
- HTS HMM-based Speech Synthesis System，基于 HMM 的语音合成系统
- LPC Linear Prediction Coefficient，线性预测系数

- MAP Maximum A Posterior，最大后验概率
- MCD Mel-Cepstral Distortion，倒谱系数失真
- MDL Minimum Description Length，最小描述长度
- MDS Multi-Dimensional Scaling，多维标度
- MGCC Mel-Generalized Cepstral Coefficient，梅尔广义倒谱系数
- MLI Maximum Likelihood Increase，最大似然增量
- MLSA Mel Log Spectral Approximation，梅尔对数谱近似
- MLLR Maximum Likelihood Linear Regression，最大似然线性回归
- MLPG Maximum Likelihood Parameter Generation，最大似然参数生成
- MOS Mean Opinion Score，平均意见分
- MSD Multi-Space Distribution，多空间分布
- PiTAR Pitch Target Realisation，基频目标实现
- PM Prosodic Model，韵律模型
- RMSE Root-Mean-Square-Error，根均方误差
- SA Speaker Adaptation，说话人自适应
- SI Speaker Independent，说话人无关
- SMAP Structural Maximum A Posterior，结构化最大后验概率
- SMAPLR Structural Maximum A Posterior Linear Regression，结构化最大后验概率线性回归
- SPTK Speech Processing Tool Kit，语音处理工具包
- SSM Supra-Segmental Model，超音段模型
- SSML Speech Synthesis Markup Language，语音合成标记语言
- TA Target Approximation，目标逼近
- ToBI Tone and Break Index，调式与停顿标记
- TTS Text-To-Speech，文语转换
- VC Voice Conversion，声音转换
- VFS Vector Field Smoothing，矢量场平滑
- VPR Voice Print Recognition，声纹识别
- VTLN Vocal Tract Length Normalization，声道长度规整