Mandarin TTS using HTS toolkit

National Tsing Hua University
Chi-Yueh Lin
2010/01/25
Outline

• Blizzard Challenge 2009
• HTS overview and installation
• HTS Feature Extraction and Model Training
• HTS Context-Dependent Labels and Question Set
• Possible problems during the HTS training
Blizzard Challenge 2009
Blizzard Challenge 2009

- Three Mandarin TTS tasks in Blizzard Challenge 2009
  - MH:
    - building voices from the full Mandarin database.
    - about 6,000 utterances/130,000 Chinese characters.
  - MS1:
    - building voices from only few sentences.
    - 10, 50, and 100 utterances.
  - MS2:
    - Same as MH, but for telephone channel.
Blizzard Challenge 2009

• Evaluation (all judged by human listeners)
  o Naturalness
    ▪ A scale from 1 (completely non-natural) to 5 (completely natural)
  o Similarity
    ▪ A scale from 1 (sounds like a totally different person) to 5 (sounds like exactly the same person).
  o Intelligibility
    ▪ CER : Character Error Rate
    ▪ PTER : Pinyin-Tone Error Rate
    ▪ PER : Pinyin Error Rate
Blizzard Challenge 2009

- Evaluation Results
  - MH
    - no system is as natural as the natural speech.
    - only one system is significantly more natural than all other systems.
  - MS1
    - no system is as natural as the natural speech.
    - no system is as similar as the original speaker.
  - MS2
    - one system is significantly more natural than all other systems.
Blizzard Challenge 2009

- Mandarin TTS systems
  - Unit selection
    - I²R, PKU, Toshiba, CASIA, DFKI
  - HMM-based
    - NTUT, NIT, ITRI
- On average, **NIT's system** has the best synthesis quality in terms of
  - Naturalness
  - Intelligibility
HTS overview and installation
HTS - HMM-based speech synthesis system

- Current version: 2.1
  - Hidden semi-Markov model (HSMM)
    - Training/adaptation/synthesis
  - Multi-space probability distribution for F0
    - F0 value for each frame is either zero or non-zero.
  - Tree-based clustering based on the MDL criterion
    - MDL: Minimum Description Length
    - HTK employs ML criterion
  - Considering global variance (GV)
    - Alleviate a muffled, over-smoothed voice.
HTS Overview

Figure is adopted from H. Zen et al., "Recent development of the HMM-based speech synthesis system (HTS)"

2010/1/25 NGASR Workshop 2010
HTS - Installation

• Software requirements
  o HTK 3.4 & HDecode 3.4
    http://htk.eng.cam.ac.uk/
  o HTS 2.1 patch for HTK 3.4
    http://hts.sp.nitech.ac.jp/?Download
  o HTS Engine API 1.02
    http://hts-engine.sourceforge.net/
  o SPTK 3.3
    http://sp-tk.sourceforge.net/
  o HTS training demo (Japanese or English)
    http://hts.sp.nitech.ac.jp/?Download
HTS - Installation Flow

1. Unzip HTK and HDecode
2. Unzip HTS-2.1_for_HTK-3.4
   o copy HTS-2.1_for_HTK-3.4.patch to /htk
   o run >>patch -p1 -d . < HTS-2.1_for_HTK-3.4.patch
   o There must be no error during the patching.
3. Build HTS
   o >>./configure (HTS will be installed in /usr/local/HTS-2.1 by default) or
     >>./configure --prefix=/absolute_path (manually specify the installation directory)
   o >>make all and >>make install
HTS - Installation Flow

4. Unzip and Build HTS Engine
   o `>>./configure` (default is `/usr/local/`) or `>>./configure --prefix=/absolute_path` (manually specify the installation directory)
   o `>>make` and `>>make install`

5. Unzip and Build SPTK
   o `>>./configure` (default is `/usr/local/SPTK`) or `>>./configure --prefix=/absolute_path` (manually specify the installation directory)
   o `>>make` and `>>make install`
HTS - Installation Flow

6. Unzip and Build HTS-Demo
   o Make sure Tcl/Tk 8.4 (with Snack toolkit) and SoX were installed.
     ▪ Snack Toolkit: extract F0 from speech waveform
     ▪ SoX (Sound eXchange): manipulate sound file
   o >>./configure (default is /usr/local/) or
     >>./configure --with-sptk-search-path=/absolute_path \ 
       --with-hts-search-path=/absolute_path \ 
       MGCORDER=24 MGCLSP=0 GAMMA=0 \ 
       FREQWARP=0.42
     (manually specify the search directory and Mel-cepstrum)
     Please see INSTALL for more various detail, especially when
     MGC-LSP is preferred.
   o >>make all and run the entire HTS-Demo training
HTS - Installation Suggestions

• Install HTS on Linux
  o Training scripts need some Linux commands
  o Running on a Windows platform needs Cygwin.
• Running DEMO program provided by HTS website first
  o Necessary directories and files are built automatically
  o Replace your own data and re-run the training
• Keep original HTK on disk if you need it to perform speech recognition.
  o HTS-patched HTK modified some codes, and some functions are not run as you expect to.
HTS - directory structures

- /configs
  - Configuration files used in training, most of them aim at flooring variances
- /edfiles
  - Editing files used by HHEd, which are in charge of model manipulations of HMMs
- /gen
  - Synthesized voices (check your results here!)
- /models
  - HTK-formatted models during the training
- /proto
  - Prototype HMM, only used in the very beginning of training
HTS - directory structures

- /stats
  - Statistic files about the training, e.g. number of occurrences and average occupancy count

- /tree
  - HTK-formatted clustering trees

- /voices
  - HTS-formatted models (HTS engine needs them!)

- /scripts
  - Two files controlling the training: Config.pm and Training.pl
HTS - directory structures

- Waveform and feature files
  - /data/raw: waveform files with no file header
  - /data/mgc (or /mcp): mel-spectrum features
  - /data/lf0: logarithmic fundamental frequency
  - /data/gv: global variances about MGC and LF0
  - /data/cmp: composited features (MGC+LF0)

- Other files
  - /data/scp: file lists for the training and synthesis
  - /data/win: parameters used in feature extraction
  - /data/scripts: scripts used in the data preparation
HTS - directory structures

- Files about labels
  - /data/labels
    - /data/labels/mono
      - mono-phone label files
      - Label needs to include time information because HRest needs it.
    - /data/labels/full:
      - context-dependent label files
      - Time information can be omitted.
    - /data/labels/gen:
      - Context-dependent label files for the synthesis
  - /data/questions: question set used in tree clustering
  - /data/lists: distinct labels
HTS Feature Extraction and Model Training
HTS - Feature Representation

• Spectral parameters
  o SPTK (Speech Signal Processing Toolkit)
    ▪ Cepstrum
    ▪ Mel-cepstrum
    ▪ Generalized cepstrum
    ▪ Mel-generalized cepstrum (MGC)
    ▪ Line spectral pair (LSP)
    ▪ Mel-LSP
    ▪ MGC-LSP

• Logarithmic fundamental frequency
  o “snack” toolkit
    ▪ “wavesurfer” is based on “snack” as well.
HTS - Feature Representation

Spectral parameters (MGC) x M
Spectral parameters (LF0) x 1

Window function (Δ- and ΔΔ-)

<table>
<thead>
<tr>
<th>MGC</th>
<th>Δ-MGC</th>
<th>ΔΔ-MGC</th>
<th>LF0</th>
<th>Δ-LF0</th>
<th>ΔΔ-LF0</th>
</tr>
</thead>
<tbody>
<tr>
<td>x 3M</td>
<td></td>
<td></td>
<td>x 1</td>
<td>x 1</td>
<td>x 1</td>
</tr>
</tbody>
</table>

Stream 1
One Gaussian per state

Stream 2
Two mixtures per state, “voiced” and “unvoiced”

Stream 3

Stream 4

(cmp)
HTS - Feature Representation

• Multi-Space Probability Distribution (MSD)

In HTS implementation, "voiced" space consists of only one Gaussian distribution, while "unvoiced" space is zero-dimensional.

HTS - training procedure

1. Training CI-HMMs
   - **HCompV**: computing the global variances
     - In fact, only the variance floors (x 0.01) are saved.
   - **HInit**: parameter estimation by Viterbi algorithm
   - **HRest**: parameter estimation by Baum-Welch algorithm
   - **HHEd**: flooring small variances if necessary
     - require variance floors computed in HCompV
     - all stream weights are tied
   - **HERest**: five iterations of embedded re-estimation
2. Expanding CI-HMMs to CD-HMMs
   - **HHEd**: generating CD-HMMs by duplicating CI-HMMs.
     - The number of CD-HMMs depends on the number of distinct context-dependent labels. (very large)
   - Transition probabilities of HMMs having the same CI phone are tied together.
   - **HERest**: one iteration of embedded re-estimation
     - output statistics for MGC and LF0
       - the number of occurrences
       - average number of frames per state
     - output DUR statistics based on the MGC+LF0 one
HTS - training procedure

3. Tree-based clustering 1
   - **HHEd**: loading the question set and constructing a decision tree based on MDL criterion
     - Three independent trees, one for MGC, one for LF0, and one for DUR.
     - On average, less than 1% of models remain
   - **HERest**: five iterations of embedded re-estimation

4. Untie CD-HMMs
   - **HHEd**: untie model parameters
     - MGC, LF0, and DUR are all untied
   - **HERest**: one iteration of embedded re-estimation
     - output statistics for MGC, LF0, and DUR
HTS - training procedure

5. Tree-based clustering 2
   - **HHEd**: loading the question set and constructing a decision tree based on MDL criterion
     - Three independent trees, one for MGC, one for LF0, and one for DUR.
     - On average, less than 1% of models remain
   - **HERest**: five iterations of embedded re-estimation

6. Converting HTK MMF format to HTS model format
   - model training completed in this step
   - HTS engine synthesizes voices
HTS Context-Dependent Labels and Question Set
Text Analysis - Hierarchy
Context-Dependent Label

- Trigram or five-gram of phonemic or sub-syllabic units. Units can be:
  - Initials and finals
  - Phones (dividing one final into several parts)
- Trigram of syllabic tones:
  - Current, preceding, and following tones
  - Five tones in Mandarin (輕、一、二、三、四)
- Relative positions (forward and backward positions) and the number of units in a specific level:
  - phonemic/sub-syllabic unit in a character, a word, a phrase, or a sentence.
  - character in a word, a phrase, or a sentence.
  - word in a phrase, or a sentence.
Context-Dependent Label - Blizzard Challenge

0  5058750  XX-sil+j/S:3
5058750  5713750  sil-j+iou/S:3/A:XX_4@2/B:XX_p@ngp
   /C:1_1@1$1#1&1|1/D:1_1@1$1#1&1|1
   /E:2_8@18$3#8&3|5/F:2_8@18$3#8&3|5/G:4_0@0
5713750  6738750  j-iou+l/S:5/A:XX_4@2/B:XX_p@ngp
   /C:1_1@1$1#1&1|1/D:1_1@1$1#1&1|1
   /E:2_8@18$3#8&3|5/F:2_8@18$3#8&3|5/G:4_0@0
6738750  7211875  iou-l+ian/S:3/A:4_2@2/B:XX_p@ngp
   /C:3_2@2$1#1&1|1/D:2_2@2$1#1&1|1
   /E:1_7@17$3#8&3|5/F:2_8@18$3#8&3|5/G:0_1@0
7211875  9064375  l-ian+uagn/S:5/A:4_2@2/B:XX_p@ngp
   /C:3_2@2$1#1&1|1/D:2_2@2$1#1&1|1
   /E:1_7@17$3#8&3|5/F:2_8@18$3#8&3|5/G:0_1@0
Context-Dependent Label - Blizzard Challenge

• **sil-b+i an/S:3**
  b-ian+ch/S:5
  o preceding unit identity
  o current unit identity
  o following unit identity
  o number of states for the unit
  • **/A:XX_1@2**
      o The tone of the preceding character
      o The tone of the current character
      o The tone of the following character
  • **/B:XX_n@v**
      o The POS of the preceding word
      o The POS of the current word
      o The POS of the next word
Context-Dependent Label - Blizzard Challenge

- /D:1_1@1$1#1&1|1
  - Forward position of a character in a word
  - Forward position of a character in a phrase
  - Forward position of a character in a sentence
  - Forward position of a word in a phrase
  - Forward position of a word in a sentence
  - Forward position of a phrase in a sentence
  - Forward position of a sentence in a paragraph

- /E:2_2@8$1#4&2|4
  - same as /D: features, but denotes backward position
Context-Dependent Label - Blizzard Challenge

- /F:2_2@8$1#4&2|4
  - The number of characters in a word
  - The number of characters in a phrase
  - The number of characters in a sentence
  - The number of words in a phrase
  - The number of words in a sentence
  - The number of phrases in a sentence
  - The number of sentence in a paragraph
- /C:1_1@1$0#1&1|1 and /G:1_0@0
  - Oops, I’m not able to decipher them, but might be relevant to characteristics about prosody.
Question Set - Format

- Each question is defined by three fields: HHEd command 'QS', question description, and a set of matching patterns
  - \textbf{QS} \textbf{L_SM==b} \{ b-* \}
    - \textbf{QS} : HHEd command
    - \textbf{L_SM==b} : "Is left initial 'b'?", do NOT contain any space in this string
    - \{ b-* \} : find any label with b- as a sub-string
- Multiple matching patterns are separated by commas.
  - \textbf{QS} \textbf{C_YM=LT3} \{ *-e+*, *-ie+*, *-ve+* \}
- The matching pattern is NOT Perl-compatible
  - Asterisk '*' matches any character
  - Question mark '?' matches one character
- Check DoMatch(), RMatch() in HShell.c for detail
Question Set - Phonetic Questions

- Questions about N-gram of phonemic/sub-syllabic units
  - $\text{QS } L\_\text{SM}==b \{ \text{b-*} \}$ % left unit is initial 'b'
  - $\text{QS } C\_\text{YM}==u \{ \text{*-u+*} \}$ % current unit is final 'u'
  - $\text{QS } R\_\text{SM}==m \{ \text{*+m/S:*} \}$ % following unit is initial 'm'

- Questions about units in some broad phonetic sense
  - $\text{QS } R\_\text{SM}=LT3$
    - $\{ \text{*+zh/S:*}, \text{*+ch/S:*}, \text{*+sh/S:*}, \text{*+r/S:*} \}$ % following unit belongs to 'retroflex'
  - $\text{QS } C\_\text{YM}=LT9 \{ \text{*+ai/S:*}, \text{*+uai/S:*} \}$ % current unit belongs to final and ending with an 'ai'
Question Set - Tonal Questions

• Questions about surrounding tones
  o **QS L_Tone==1**  { */A:1_/*/B:* }  
    % the tone of the preceding character is '1' (一聲)
  o **QS C_Tone==2**  { */A:*2@*/B:* }  
    % the tone of the current character is '2' (二聲)
  o **QS R_Tone==3**  { */A:*@3/B:* }  
    % the tone of the following character is '3' (三聲)
  o **QS C_Tone==14**  
    { */A:*1@*/B:* , */A:*4@*/B:* }  
    % the tone of the current character is '1' or '4' (一聲 or 四聲)
  o **QS R_Tone<=1**  
    { */A:*@0/B:* , */A:*@1/B:* }  
    % the tone of the following character is '0' or '1' (輕聲 or 一聲)
Question Set - Part-of-Speech Questions

• **QS C_POS==a** { */B:*_a@*/C:* }
  % the POS of the current word is 'a'

• The question set provided by iFlyTek only considers POS of current word, neither preceding word nor following word. Of course we can append such questions in the set...

• **QS L_POS==v** { *@v/C:* }
  % the POS of the preceding word is 'v'
• **QS R_POS==n** { */B:n_* }
  % the POS of the following word is 'n'
Question Set - Relative Position Questions

• Questions about the relative position of unit in a higher level.
  - QS C_AHP_L0inL1<=02 {*/D:1_*/E:*,*/D:2_*/E:*}
    % forward position of a character in a word.
  - QS C_ATP_L0inL1<=02 {*/E:1_*/F:*,*/E:2_*/F:*}
    % backward position of a character in a word.

• Similar questions in the set
  - C_AHP_L0inL1 % forward position of a character in a word
  - C_AHP_L0inL3 % forward position of a character in a phrase
  - C_AHP_L0inL4 % forward position of a character in a sentence
  - C_AHP_L1inL3 % forward position of a word in a phrase
  - C_AHP_L1inL4 % forward position of a word in a sentence
  - C_AHP_L3inL4 % forward position of a phrase in a sentence

• Questions about backward position are those with C_ATP_ as question description.
Question Set - Count Questions

• Questions about the number of units in some higher level unit
  o \( QS \ C\_AN\_L0inL3 \leq 02 \)
    \{ */F:*_1@*/G:*, */F:*_2@*/G:* \}
    % the number of words in a phrase is less than or equal to 2
• Similar questions in the set
  o C\_AN\_L0inL3 % the number of characters in a phrase
  o C\_AN\_L1inL4 % the number of words in a sentence
  o C\_AN\_L3inL4 % the number of phrases in a sentence
Question Set - Design Principle

- Put as many questions as you can according to the context-dependent labels. Let the decision tree and MDL criterion choose the proper ones.
- Which kinds of questions are necessary and which are additional?
  - Necessary (generate average quality voices)
    - tone
    - relative position
    - number of occurrences
  - Additional (improve the prosody of synthesized voices)
    - Prosody word, prosody boundary
  - Additional (improve the quality marginally)
    - Part-of-Speech (POS)
Scrutinize the HTS model and trace files
HTS model files

- Model files needed in HTS engine can be found in /voices directory
  - .pdf (probability density function file)
    - Binary-formatted
    - Statistics of parameters
  - .inf (decision tree file)
    - Text-formatted
    - Selected questions
    - Tree topology
  - .win (window function)
    - Text-formatted
    - Not interested here
HTS - Model File Format

- TREE files (.inf as extension) : text format
  - Selected questions for the parameter kind
  - Tree configuration

```plaintext
{ }[2]
{
  0 PhonePositionSyl>=2  -1  -3
  -1 PhonePositionSyl==1  -2  -6
  -2 L_pau_sil  -8  -7
  -3 C_CFang2  -4  45
  -4 C_CFang2  -5  -29
  . . .
  -910 R_CFen1
  -911 L_r
  -912 Phr_ForwardPosition_Sentence<=3
  -913 L_SylTone==3
  -914 R_CFy10
  -915 Syl_Number_Sentence<=28
}
```

"lf0_s2_909" : the 909th LF0 PDF in 1st state ("2" is the first physical state in HTK terminology)
HTS - Model File Format (MGC PDF)

- **MGC**
  - `[VecDim]`: 4-byte integer
  - `[NumPDF]`: 4-byte integer x `NumState`
  - `[[[mean] x VecDim, [var] x VecDim] x NumPDF] x NumState`

```
0000004B 000000B8 000000BC 0000009F 0000009E 000000D9 4005E46E 400360DA 3F2F4AB1
3EAFD1AB BD8A5545 BEA8E055 3E7A0A55 3E47209F 3E981CB3 3E92CB1D 3DEE9137 3D431AC5
3E02D1CE 3D87C5FA 3D03E8DD 3E2757EE 3D9FBF56 BC95FD5D BD79B862 BD831588 BD8312FA
BC42C22E 3CBDF54F 3B4D5D20 3D46643A BDF0E8E3 BCD9CC62 BC725F2C BCD9E618 BCDAD10B
3D05AE72 BC2B5125 3C48ECE4 3C5DB3D8 BA81DE15 BC4ACDBD 3CAE04A3 3B034C3C 3ACE54F6
3C37CE52 3AC0414C BBCA1EED 3BD067AF 3B188FFA 3B9B933C 3CA4792E 3C77C871 3BE4D64F
3BFF3565 BAA8C595 3BC89F89 BC7197EB 3BF7A0B6 3C225792 3C73F5CF 3C366C74 BB6012EC
BB5D4073 BBD092D7 BBC02DC5 392E308B 3A69E309 BB0E2825 3BA20F1B 3AB81BCB BB597DA7
3AA96D62 3B261044 3B6339FE 3B70C80F 3B0D90D9 BB123F9F BA90C314 BB5BF196 BB38186C
3E999B20 3DCF5D85 3D4FBD8D 3D031AA4 3CEF7540 3D4C65AF 3CEF9284 3CBEB890 3C8DB6DB
```

- **vector size is '75', the numbers of PDFs for 5 states, mean vector of 1st PDF in 1st state, variance vector of 1st PDF in 1st state, mean vector of 2nd PDF in 1st state, and so on.**
HTS - Model File Format (LF0 PDF)

- LF0
  - [VecDim] : 4-byte integer
  - [NumPDFs] : 4-byte integer x NumState
  - [[[mean], [var], [w_v], [w_uv]] x VecDim] x NumPDFs] x NumState

- Vector size is '3', the numbers of PDFs for 5 states, 1st PDF for three independent LF0 streams, 2nd PDF for three independent LF0 streams
HTS - Model File Format (LF0 PDF)

- means of LF0, \( \Delta LF0 \), and \( \Delta \Delta LF0 \) of 1st PDF in 1st state,
- variances of LF0, \( \Delta LF0 \), and \( \Delta \Delta LF0 \) of 1st PDF in 1st state,
- MSD weighting factors for “voiced” of LF0, \( \Delta LF0 \), \( \Delta \Delta LF0 \) of 1st PDF in 1st state,
- MSD weighting factors for “unvoiced of LF0, \( \Delta LF0 \), \( \Delta \Delta LF0 \) of 1st PDF in 1st state.

- The following block represents the mentioned information of 2nd PDF in 1st state, and so on.
- Only the first weighting factor for “voiced” is used in parameter generation.
HTS - Model File Format (DUR PDF)

- **DUR**
  - `[VecDim] : 4-byte integer`
  - `[NumPDF] : 4-byte integer`
  - `[[mean] x VecDim, [var] x VecDim] x NumPDF`

- **vector size is '5'**, DUR has 259 PDFs, mean vector of 1st PDF, variance vector of 1st PDF, mean vector of 2nd PDF, variance vector of 2nd PDF, and so on.
HTS - Model File Format (GV-MGC PDF)

• GV-MGC
  o [VecDim] : 4-byte integer
  o [NumPDF] : 4-byte integer
  o [[mean] x VecDim, [var] x VecDim]

```
00000019 00000001 40664E6E 3F491740 3E281248
3E62894F 3DCA8467 3DE2EC01 3D25B757 3DA12665
3D137C7B 3D2796EF 3D57D843 3D0FB7AB 3CE889AA
3CE26F0A 3CA53009 3C96024E 3C95D7B5 3C9F5DEF
3C698BED 3C8C0E61 3C58B92E 3C40DB0C 3C2A2DEC
3C2CF7B5 3C079D33 3ECFCBF6 3CEB0FA9 3A28220E
3B0B2180 39B29B31 39FBBDD6 38484D2D 3932288A
382C3B30 38427E76 38B67DC5 37D2100B 37D9E728
37C1EC5D 3716EAC5 37269083 37193D1F 3735132D
368C6990 3704F367 366B8619 3652E6C0 3600C587
363A9209 35C0096
```

Vector size is 25, only 1 model here, including mean vector and variance vector
HTS - Model File Format (GV-LF0 PDF)

• GV-LF0
  o [VecDim] : 4-byte integer
  o [NumPDF] : 4-byte integer
  o [[mean] x VecDim, [var] x VecDim] : both [mean] and [var] are 8-byte floats

  00000001 00000001 3CDA7823 383F0501

Vector size is 1, only 1 model here, including mean and variance.
HTS - Trace Synthesized Voices

- Each time HTS engine synthesizes a voice, there is an accompanying .trace file.
HTS - Trace Synthesized Voices

- By combining the tree model (.inf), trace file (.trace), and a little programming, one can visualize the path of a selected model and the questions being asked.
HTS model files

• File formats may change version by version, check them carefully when you need to extract parameters directly.
  – Format of LF0 PDFs in HTS 2.1 is totally different from that in HTS 2.0.
  – Trace file contents during the synthesis are also different.
Possible Problems during the HTS training
Beta probability error?

- One possible error during the training. Check the log file to find which files go wrong.
  - It may or may not terminate the training procedure.
  - Feature extraction goes wrong with those files. Remove them from the training list if only a few.
No text parser at hand?

- There is one in Windows XP!
  - \Program Files\Windows NT\Accessories\Lctool
  - Execute and export to .TXT format
  - 44,577 words in total
Discontinuous Pitch Contour

• The Snack toolkit is not a very accurate F0 estimator.
  • Try STRAIGHT or Praat
• After the feature extraction, but before the training
  o interpolate or extrapolate a pitch contour
    ▪ basic polynomial fitting
    ▪ orthonormal basis decomposition
• After the parameter generation, but before the synthesis
  o Altering a sequence of voiced/unvoiced status.
Discontinuous Pitch Contour

- Degrade the quality of synthesized speech drastically, especially misjudge a voiced state as an unvoiced one.
- No excitation signal in an unvoiced state.

Figure is from “Voiced/Unvoiced Decision Algorithm for HMM-based Speech Synthesis”, INTERSPEECH 2009

Figure 2: Mandarin syllable “Hui4” with a bad “ui”.
• V/UV change point in a Mandarin syllable.

\[ x = \frac{S_{\text{unvoiced}}}{n_{\text{end}} - n_{\text{start}} + 1} \]

\[ p(x | \lambda) = \sum_{k=1}^{K} \omega_k N(x | \mu_k, \Sigma_k) \]

Figure is from “Voiced/Unvoiced Decision Algorithm for HMM-based Speech Synthesis”, INTERSPEECH 2009
Discontinuous Pitch Contour

Figure 1: Five examples of voiced phones with erroneous v/u predictions.

Figure is from “A Minimum V/U Error Approach to F0 Generation in HMM-based TTS”, INTERSPEECH 2009
Discontinuous Pitch Contour

Figure is from “A Minimum V/U Error Approach to F0 Generation in HMM-based TTS”, INTERSPEECH 2009
Discontinuous Pitch Contour

\[ e_j^{uv} = V_j^{uv} + U_j \]

\[ V_j^{uv} = V_{j-1}^{uv} + \gamma(j, g = v), \quad V_0^{uv} = 0, \quad j = 1, \ldots, N \]

\[ U_j^{uv} = U_{j+1}^{uv} + \gamma(j, g = u), \quad V_{N+1}^{uv} = 0, \quad j = N, \ldots \]

\[ e_j^{vi} = V_j^{vi} + U_j \]

\[ V_j^{vi} = V_{j+1}^{vi} + \gamma(j, g = v), \quad V_{M+1}^{vi} = 0, \quad j = M, \ldots \]

\[ U_j^{vi} = U_{j-1}^{vi} + \gamma(j, g = u), \quad V_0^{vi} = 0, \quad j = 1, \ldots, M \]

Figure is from “A Minimum V/U Error Approach to F0 Generation in HMM-based TTS”, INTERSPEECH 2009
Thank You!