Lecture 15: Waveform Synthesis in TTS

Original slides by Dan Jurafsky, Alan Black, & Richard Sproat
Predicting Intonation in TTS

**Prominence/Accent:** Decide which words are accented, which syllable has accent, what sort of accent

**Boundaries:** Decide where intonational boundaries are

**Duration:** Specify length of each segment

**F0:** Generate F0 contour from these
ToBI: Tones and Break Indices

- **Pitch accent tones**
  - H* “peak accent”
  - L* “low accent”
  - L+H* “rising peak accent” (contrastive)
  - L*+H ‘scooped accent’
  - H+!H* downstepped high

- **Boundary tones**
  - L-L% (final low; Am English Declarative contour)
  - L-H% (continuation rise)
  - H-H% (yes-no question)

- **Break indices**
  - 0: clitics, 1, word boundaries, 2 short pause
  - 3 intermediate intonation phrase
  - 4 full intonation phrase/final boundary
marianna  made  the  marmalade
marianna

made

the

marmalade

1 1 1 4

0 1.49
Examples of the TOBI system

• I don’t eat beef.
  L* L* L*L-L%

• Marianna made the marmalade.
  H* L-L%
  L* H-H%

• “I” means insert.
  H* H* H*L-L%
  1
  H*L- H*L-L%
  3

Slide from Lavoie and Podesva
ToBI

- [http://www.ling.ohio-state.edu/~tobi/](http://www.ling.ohio-state.edu/~tobi/)
- **TOBI for American English**
  - [http://www.ling.ohio-state.edu/~tobi/ame_tobi/](http://www.ling.ohio-state.edu/~tobi/ame_tobi/)


FO Generation

• By rule
• By linear regression
• Some constraints
  • By accents and boundaries
  • F0 declines gradually over an utterance (“declination”)

Declination

- F0 tends to decline throughout a sentence
F0 Generation by rule

Generate a list of target F0 points for each syllable

For example:

Generate simple H* “hat” accent (fixed speaker-specific F0 values) with 3 pitch points: [110, 140, 100]

Modified by

- gender,
- declination,
- end of sentence,
- etc.
FO generation by regression

- Supervised machine learning
- We predict: value of F0 at 3 places in each syllable
- Predictor features:
  - Accent of current word, next word, previous
  - Boundaries
  - Syllable type, phonetic information
  - Stress information
- Need training sets with pitch accents labeled
- F0 is generally defined relative to pitch range
  - Range between baseline and topline frequency in an utterance
Euclidean distance for F0

- Pure Euclidean distance not necessarily a good objective function for F0
- Perception maps more to “shape types” of the F0 curve than precise pitch values
- (See whiteboard example)
Output of festival generator

<table>
<thead>
<tr>
<th>do</th>
<th>uw</th>
<th>y</th>
<th>uw</th>
<th>H*</th>
<th>really</th>
<th>want</th>
<th>to</th>
<th>see</th>
<th>L*</th>
<th>L- H%</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>110</td>
<td>50</td>
<td>50</td>
<td>75</td>
<td>64</td>
<td>57</td>
<td>82</td>
<td>57</td>
<td>50</td>
<td>72</td>
</tr>
</tbody>
</table>
Output of festival
Part II: Waveform Synthesis

Given:

- String of phones
- Prosody
  - Desired F0 for entire utterance
  - Duration for each phone
  - Stress value for each phone, possibly accent value

Generate:

- Waveforms
The two stages of TTS

PG&E will file schedules on April 20.

1. **Text Analysis**: Text into intermediate representation:

2. **Waveform Synthesis**: From the intermediate representation into waveform
PG&E will file schedules on April 20.
Outline: Waveform Synthesis in TTS

- Diphone Synthesis
- Unit Selection Synthesis
  - Target cost
  - Unit cost
- Joining
  - Dumb
  - PSOLA
- HMM Synthesis
Internal Representation:
Input to Waveform Synthesis
Diphone TTS architecture

- **Training:**
  - Choose units (kinds of diphones)
  - Record 1 speaker saying 1 example of each diphone
  - Mark the boundaries of each diphones,
    - cut each diphone out and create a diphone database

- **Synthesizing an utterance,**
  - grab relevant sequence of diphones from database
  - Concatenate the diphones, doing slight signal processing at boundaries
  - use signal processing to change the prosody (F0, energy, duration) of diphone sequence
Diphones

- Mid-phone is more stable than edge:
Diphones

- mid-phone is more stable than edge
- Need $\sim |\text{phones}|^2$ number of units
  - Some combinations don’t exist (hopefully)
- ATT (Olive et al. 1998) system had 43 phones
  - 1849 possible diphones
  - Phonotactics ([h] only occurs before vowels), don’t need to keep diphones across silence
  - Only 1172 actual diphones
- May include stress, consonant clusters
  - So could have more
- Lots of phonetic knowledge in design
- Database relatively small (by today’s standards)
  - Around 8 megabytes for English (16 KHz 16 bit)
    
    Slide from Richard Sproat
Voice

- Human speaker
  - Called a voice talent
- Diphone database
  - Called a voice
Designing a diphone inventory: Nonsense words

- Build set of carrier words:
  - pau t aa b aa b aa pau
  - pau t aa m aa m aa pau
  - pau t aa m iy m aa pau
  - pau t aa m iy m aa pau
  - pau t aa m ih m aa pau

- Advantages:
  - Easy to get all diphones
  - Likely to be pronounced consistently
    - No lexical interference

- Disadvantages:
  - (possibly) bigger database
  - Speaker becomes bored
Designing a diphone inventory: Natural words

• Greedily select sentences/words:
  • Quebecois arguments
  • Brouhaha abstractions
  • Arkansas arranging

• Advantages:
  • Will be pronounced naturally
  • Easier for speaker to pronounce
  • Smaller database? (505 pairs vs. 1345 words)

• Disadvantages:
  • May not be pronounced correctly
Making recordings consistent:

- Diphone should come from mid-word
  - Help ensure full articulation
- Performed consistently
  - Constant pitch (monotone), power, duration
- Use (synthesized) prompts:
  - Helps avoid pronunciation problems
  - Keeps speaker consistent
  - Used for alignment in labeling
Building diphone schemata

- Find list of phones in language:
  - Plus interesting allophones
  - Stress, tons, clusters, onset/coda, etc
  - Foreign (rare) phones.

- Build carriers for:
  - Consonant-vowel, vowel-consonant
  - Vowel-vowel, consonant-consonant
  - Silence-phone, phone-silence
  - Other special cases

- Check the output:
  - List all diphones and justify missing ones
  - Every diphone list has mistakes

Slide from Richard Sproat
Recording conditions

- **Ideal:**
  - Anechoic chamber
  - Studio quality recording
  - EGG signal

- **More likely:**
  - Quiet room
  - Cheap microphone/sound blaster
  - No EGG
  - Headmounted microphone

- **What we can do:**
  - Repeatable conditions
  - Careful setting on audio levels
Labeling Diphones

- Run a speech recognizer in forced alignment mode
  - Forced alignment:
    - Given: A trained ASR system, a wavfile, a transcriptions
    - Returns: an alignment of the phones to the wavfile

- Much easier than phonetic labeling:
  - Words and phone sequence are defined
  - They are clearly articulated
  - But sometimes speaker still pronounces wrong, so need to check.

- Phone boundaries less important
  - +- 10 ms is okay

- Midphone boundaries important
  - Where is the stable part
  - Can it be automatically found?
Diphone auto-alignment

- Given
  - synthesized prompts
  - Human speech of same prompts
- Do a dynamic time warping alignment of the two
  - Using Euclidean distance
- Works very well 95%+
  - Errors are typically large (easy to fix)
  - Maybe even automatically detected
- Malfrere and Dutoit (1997)
Dynamic Time Warping

chard Sproat
Finding diphone boundaries

- Stable part in phones
  - For stops: 33% in
  - For phone-silence: 25% in
  - For other diphones: 50% in

- In time alignment case:
  - Given known diphone boundaries in prompt in label file
  - Use DTW to find same stable point in new speech

- Optimal coupling
  - Taylor and Isard 1991, Conkie and Isard 1996
  - Instead of precutting the diphones
    - Wait until we are about to concatenate the diphones together
    - Then take the 2 complete (uncut diphones)
    - Find optimal join points by measuring cepstral distance at potential join points, pick best

Slide modified from Richard Sproat
Concatenating diphones: junctures

- If waveforms are very different, will perceive a click at the junctures
  - So need to window them
- Also if both diphones are voiced
  - Need to join them pitch-synchronously
- That means we need to know where each pitch period begins, so we can paste at the same place in each pitch period.
  - Pitch marking or epoch detection: mark where each pitch pulse or epoch occurs
    - Finding the Instant of Glottal Closure (IGC)
  - (note difference from pitch tracking)
Epoch-labeling

- An example of epoch-labeling using “SHOW PULSES” in Praat:
Epoch-labeling: Electroglottograph (EGG) = laryngograph, Lx

- Straps on speaker’s neck near larynx
- Sends small high frequency current through adam’s apple
- Human tissue conducts well; air not as well
- Transducer detects how open the glottis is (l.e. amount of air between folds) by measuring impedance.
Less invasive way to do epoch-labeling

- Signal processing
  - E.g.:
Prosodic Modification

- Modifying pitch and duration independently
- Changing sample rate modifies both:
  - Chipmunk speech
- Duration: duplicate/remove parts of the signal
- Pitch: resample to change pitch
Speech as Short Term signals

Alan Black
Duration modification

- Duplicate/remove short term signals
Duration modification

- Duplicate/remove short term signals
Pitch Modification

- Move short-term signals closer together/further apart
Overlap-and-add (OLA)

Huang, Acero and Hon
Windowing

- Multiply value of signal at sample number $n$ by the value of a windowing function
- $y[n] = w[n]s[n]$

**rectangular**

$$w[n] = \begin{cases} 1 & 0 \leq n \leq L - 1 \\ 0 & \text{otherwise} \end{cases}$$

**hamming**

$$w[n] = \begin{cases} 0.54 - 0.46 \cos\left(\frac{2\pi n}{L}\right) & 0 \leq n \leq L - 1 \\ 0 & \text{otherwise} \end{cases}$$
Windowing

- $y[n] = w[n]s[n]$
Overlap and Add (OLA)

- Hanning windows of length $2N$ used to multiply the analysis signal
- Resulting windowed signals are added
- Analysis windows, spaced $2N$
- Synthesis windows, spaced $N$
- Time compression is uniform with factor of 2
- Pitch periodicity somewhat lost around 4th window

Huang, Acero, and Hon
TD-PSOLA™

- Time-Domain Pitch Synchronous Overlap and Add
- Patented by France Telecom (CNET)
  - Expired in 2004
- Very efficient
  - No FFT (or inverse FFT) required
- Can modify Hz up to two times or by half
TD-PSOLA™

- Windowed
- Pitch-synchronous
- Overlap-
- -and-add
Summary: Diphone Synthesis

- Well-understood, mature technology
- Augmentations
  - Stress
  - Onset/coda
  - Demi-syllables
- Problems:
  - Signal processing still necessary for modifying durations
  - Source data is still not natural
  - Units are just not large enough; can’t handle word-specific effects, etc
Problems with diphone synthesis

- Signal processing methods like TD-PSOLA leave artifacts, making the speech sound unnatural
- Diphone synthesis only captures local effects
  - But there are many more global effects (syllable structure, stress pattern, word-level effects)
Unit Selection Synthesis

• Generalization of the diphone intuition
  • Larger units
    • From diphones to sentences
  • Many many copies of each unit
    • 10 hours of speech instead of 1500 diphones (a few minutes of speech)
  • Little or no signal processing applied to each unit
    • Unlike diphones
Why Unit Selection Synthesis

- Natural data solves problems with diphones
  - Diphone databases are carefully designed but:
    - Speaker makes errors
    - Speaker doesn’t speak intended dialect
    - Require database design to be right
  - If it’s automatic
    - Labeled with what the speaker actually said
    - Coarticulation, schwas, flaps are natural

“There’s no data like more data”
- Lots of copies of each unit mean you can choose just the right one for the context
- Larger units mean you can capture wider effects
Recording for Google Assistant
Unit Selection Intuition

- Given a big database
- For each segment (diphone) that we want to synthesize
  - Find the unit in the database that is the best to synthesize this target segment
Unit Selection Intuition

• What does “best” unit mean?
  • **Target cost**: Closest match to the target description, in terms of
    • Phonetic context
    • F0, stress, phrase position
  • **Join cost**: Best join with neighboring units
    • Matching formants + other spectral characteristics
    • Matching energy
    • Matching F0

\[
C(t_1^n, u_1^n) = \sum_{i=1}^{n} C^{\text{target}}(t_i, u_i) + \sum_{i=2}^{n} C^{\text{join}}(u_{i-1}, u_i)
\]
Targets and Target Costs

- A measure of how well a particular unit in the database matches the internal representation produced by the prior stages
- Features, costs, and weights
- Examples:
  - /ih-t/ from stressed syllable, phrase internal, high F0, content word
  - /n-t/ from unstressed syllable, phrase final, low F0, content word
  - /dh-ax/ from unstressed syllable, phrase initial, high F0, from function word “the”
Target Costs

- Comprised of k subcosts
  - Stress
  - Phrase position
  - F0
  - Phone duration
  - Lexical identity

- Target cost for a unit:

\[
C^t(t_i, u_i) = \sum_{k=1}^{p} w^t_k C^t_k(t_i, u_i)
\]
How to set target cost weights (1)

- What you really want as a target cost is the **perceivable acoustic difference** between two units.
- But we can’t use this, since **the target is not acoustic** yet, we haven’t synthesized it!
- We have to use features that we get from the TTS upper levels (phones, prosody).
- But we do have lots of acoustic units in the database.
- We could **use the acoustic distance between these** to help set the weights on the acoustic features.
How to set target cost weights (2)

- Clever Hunt and Black (1996) idea:
- Hold out some utterances from the database
- Now synthesize one of these utterances
  - Compute all the phonetic, prosodic, duration features
  - Now for a given unit in the output
  - For each possible unit that we COULD have used in its place
  - We can compute its acoustic distance from the TRUE ACTUAL HUMAN utterance.
- This acoustic distance can tell us how to weight the phonetic/prosodic/duration features
Join (Concatenation) Cost

- Measure of smoothness of join
- Measured between two database units (target is irrelevant)
- Features, costs, and weights
- Comprised of k subcosts:
  - Spectral features
  - F0
  - Energy
- Join cost:

\[ C^j(u_{i-1}, u_i) = \sum_{k=1}^{p} w_k^j C_k^j(u_{i-1}, u_i) \]
Join costs

- Hunt and Black 1996
- If $u_{i-1} == \text{prev}(u_i)$
  $cc=0$
- Used
  - MFCC (mel cepstral features)
  - Local F0
  - Local absolute power
  - Hand tuned weights
Join costs

- The join cost can be used for more than just part of search
- Can use the join cost for optimal coupling (Isard and Taylor 1991, Conkie 1996), i.e., finding the best place to join the two units.
  - Vary edges within a small amount to find best place for join
  - This allows different joins with different units
  - Thus labeling of database (or diphones) need not be so accurate
Total Costs

- Hunt and Black 1996
- We now have weights (per phone type) for features set between target and database units
- Find best path of units through database that minimize:

\[
C(t^n_1,u^n_1) = \sum_{i=1}^{n} C^{target}(t_i,u_i) + \sum_{i=2}^{n} C^{join}(u_{i-1},u_i)
\]

\[
\hat{u}^{n}_1 = \arg\min_{u_1,\ldots,u_n} C(t^n_1,u^n_1)
\]

- Standard problem solvable with Viterbi search with beam width constraint for pruning

Slide from Paul Taylor
Unit Selection Search

TARGETS

#  s-ih  ih-k  k-s  s-#  #

UNITS

#  s-ih$_1$  ih-k$_1$  k-s$_1$  s-#$_1$  #  

s-ih$_2$  ih-k$_2$  k-s$_2$  s-#$_2$  

s-ih$_3$  ih-k$_3$  

Target Costs

Join Cost
Database creation (1)

- Good speaker
  - Professional speakers are always better:
    - Consistent style and articulation
  - Ideally (according to AT&T experiments):
    - Record 20 professional speakers (small amounts of data)
    - Build simple synthesis examples
    - Get many (200?) people to listen and score them
    - Take best voices
- Correlates for human preferences:
  - High power in unvoiced speech
  - High power in higher frequencies
  - Larger pitch range

Text from Paul Taylor and Richard Sproat
Database creation (2)

- Good recording conditions
- Good script
  - Application dependent helps
    - Good word coverage
    - News data synthesizes as news data
    - News data is bad for dialog.
  - Good phonetic coverage, especially wrt context
  - Low ambiguity
  - Easy to read
- Annotate at phone level, with stress, word information, phrase breaks

Text from Paul Taylor and Richard Sproat
Creating database

- Unlike diphones, prosodic variation is a good thing
- Accurate annotation is crucial
- Pitch annotation needs to be very very accurate
- Phone alignments can be done automatically, as described for diphones
Practical System Issues

• Size of typical system:
  ~300M

• Speed:
  • For each diphone, average of 1000 units to choose from, so:
    • 1000 target costs
    • 1000x1000 join costs
    • Each join cost, say 30x30 float point calculations
    • 10-15 diphones per second
    • 10 billion floating point calculations per second

• But commercial systems must run ~50x faster than real time

• Heavy pruning essential: 1000 units -> 25 units
Unit Selection Summary

• Advantages
  • Quality is far superior to diphones
  • Natural prosody selection sounds better

• Disadvantages:
  • Quality can be very bad in places
    • HCI problem: mix of very good and very bad is quite annoying
  • Synthesis is computationally expensive
  • Can’t synthesize everything you want:
    • Unit selection (unlike diphone synth) can’t move emphasis
    • Unit selection gives good (but possibly incorrect) result
Recap: Joining Units (+F0 + duration)

- For unit selection, just like diphone, need to join the units
  - Pitch-synchronously
- For diphone synthesis, need to modify F0 and duration
  - For unit selection, in principle also need to modify F0 and duration of selection units
- But in practice, if unit-selection database is big enough (commercial systems)
  - no prosodic modifications (selected targets may already be close to desired prosody)

Alan Black
Joining Units (just like diphones)

- Dumb:
  - just join
- Better: at zero crossings
- TD-PSOLA
  - Time-domain pitch-synchronous overlap-and-add
  - Join at pitch periods (with windowing)
Evaluation of TTS

• Intelligibility Tests
  • Diagnostic Rhyme Test (DRT)
    • Humans do listening identification choice between two words differing by a single phonetic feature
      • Voicing, nasality, sustenation, sibilation
    • 96 rhyming pairs
    • Veal/feel, meat/beat, vee/bee, zee/thee, etc
      • Subject hears “veal”, chooses either “veal” or “feel”
      • Subject also hears “feel”, chooses either “veal” or “feel”
    • % of right answers is intelligibility score.

• Overall Quality Tests
  • Have listeners rate space on a scale from 1 (bad) to 5 (excellent) (Mean Opinion Score)
  • AB Tests (prefer A, prefer B) (preference tests)

Huang, Acero, Hon
Parametric Synthesis

- Developed by Tokuda and Zen
- Proposed in mid-'90s, popular since 2007ish
- Big idea: Use classifiers/regressors to predict all of F0, duration, spectral envelope. Synthesize everything
- Initial work uses the same HMM we used for ASR, but in reverse
Parametric Synthesis

+ Small footprint
+ Don’t need huge amount of data to train
+ Flexible: easier to modify pitch for emotional change, or use MLLR adaptation to change voice characteristics
+ Smooth: no discontinuities in spectrum and prosody due to join artifacts
  - Too smooth: flat, monotone, spectral smearing in time
  - Vocoding effects: buzzy unnatural sound
HMM synthesis

\[ a_{ij} : \text{state transition probability} \]

\[ b_i(o_t) : \text{output probability} \]

---

Tokuda et al 2009
HTS system overview

(Tokuda, Zen, & Black. 2009)
What does the HMM produce?

Figure 10: ML-based excitation scheme proposed by Maia et al. for HMM-based speech synthesis: filters $H_v(z)$ and $H_u(z)$ are associated with each state.

(Tokuda, Zen, & Black. 2009)
Synthesis with source-filter model

Piece-wise stationary, source-filter generative model $p(x \mid o)$

Vocal source

- Pulse train (voiced)
  - Fundamental frequency
  - Aperiodicity, voicing, ...

- White noise (unvoiced)

Vocal tract filter

- Cepstrum, LPC, ...
- Overlap/shift windowing

Speech

$x(n) = h(n) * e(n)$

Slide from Heiga Zen
Key Questions in Parametric Synthesis

+ **What parameters do we predict?** Usually MFCCs for spectrum, log F0, voicing/excitation

+ **How do we combine them?** Exact parameterization and combining them well reduces robotic buzzy effects

+ **How do we make predictions?** Choice of HMM, machine learning approaches. Less important than the vocoding/combination issues
HTS Example

- Listen to the “low level” buzzy quality characteristic of most parametric systems
- Listen to clarity/impact of plosives compared to concatenative example
Comparing vocoder/excitation models

Figure 11: Waveforms from top to bottom: natural speech and its residual, speech and excitation synthesized with simple periodic pulse-train or white-noise excitation, speech and excitation synthesized with STRAIGHT vocoding method, and speech and excitation synthesized with ML excitation method.

(Tokuda, Zen, & Black. 2009)
Generating the mean of each state:

Tokuda et al 2009
Observations generated from HMM

Tokuda et al. 2009
Choosing a sequence of means constrained by deltas and double-deltas

Tokuda and Zen 2009