Machine Learning in Speech Synthesis

Alan W Black
Language Technologies Institute
Carnegie Mellon University
Overview

- Speech Synthesis History and Overview
  - From hand-crafted to data-driven techniques
- Text to Speech Processes
- Waveform synthesis
  - Unit selection and Statistical Parametric Synthesis
- Evaluation
- Voice conversion
Physical Models

- Blowing air through tubes…
  - von Kemplen’s synthesizer 1791

- Synthesis by physical models
  - Homer Dudley’s Voder. 1939
More Computation – More Data

- **Formant synthesis (60s-80s)**
  - Waveform construction from components

- **Diphone synthesis (80s-90s)**
  - Waveform by concatenation of small number of instances of speech

- **Unit selection (90s-00s)**
  - Waveform by concatenation of very large number of instances of speech

- **Statistical Parametric Synthesis (00s-..)**
  - Waveform construction from parametric models
Waveform Generation

- Formant synthesis
- Random word/phrase concatenation
- Phone concatenation
- Diphone concatenation
- Sub-word unit selection
- Cluster based unit selection
- Statistical Parametric Synthesis
Speech Synthesis

- **Text Analysis**
  - Chunking, tokenization, token expansion

- **Linguistic Analysis**
  - Pronunciations
  - Prosody

- **Waveform generation**
  - From phones and prosody to waveforms
Text processing

- **Find the words**
  - Splitting tokens too e.g. “04/11/2009”
  - Removing punctuation

- **Identifying word types**
  - Numbers: years, quantities, ordinals
  - 1996 sheep were stolen on 25 Nov 1996

- **Identifying words/abbreviations**
  - CIA, 10m, 12sf, WeH7200
Pronunciations

- **Giving pronunciation for each word**
  - A phoneme string (plus tone, stress …)

- **A constructed lexicon**
  - ("pencil" n (p eh1 n s ih l))
  - ("two" n (t uw1))

- **Letter to sound rules**
  - Pronunciation of out of vocabulary words
  - Machine learning prediction from letters
How do you pronounce new words
4% of tokens (in news) are new
You can’t synthesis then without pronunciations
You can’t recognize them without pronunciations
Letter-to-Sounds rules
Grapheme-to-Phoneme rules
Hand written rules

- \([\text{LeftContext}] X [\text{RightContext}] \rightarrow Y\)
- e.g.
  - \(c [h \ r] \rightarrow k\)
  - \(c [h] \rightarrow ch\)
  - \(c [i] \rightarrow s\)
  - \(c \rightarrow k\)
Need an existing lexicon
- Pronunciations: words and phones
- But different number of letters and phones

Need an alignment
- Between letters and phones
- checked -> ch eh k t
LTS: alignment

- checked -> ch eh k t

<table>
<thead>
<tr>
<th>c</th>
<th>h</th>
<th>e</th>
<th>c</th>
<th>k</th>
<th>e</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>ch</td>
<td>_</td>
<td>eh</td>
<td>k</td>
<td>_</td>
<td>_</td>
<td>t</td>
</tr>
</tbody>
</table>

- Some letters go to nothing
- Some letters go to two phones
  - box -> b aa k-s
  - table -> t ey b ax-l -
Find alignment automatically

- **Epsilon scattering**
  - Find all possible alignments
  - Estimate $p(L,P)$ on each alignment
  - Find most probable alignment

- **Hand seed**
  - Hand specify allowable pairs
  - Estimate $p(L,P)$ on each possible alignment
  - Find most probable alignment

- **Statistical Machine Translation (IBM model 1)**
  - Estimate $p(L,P)$ on each possible alignment
  - Find most probably alignment
Not everything aligns

- **0, 1, and 2 letter cases**
  - $e \rightarrow \epsilon$ "moved"
  - $x \rightarrow k-s, g-z$ "box" "example"
  - $e \rightarrow y-uw$ "askew"

- **Some alignment aren’t sensible**
  - $dept \rightarrow d\ ih\ p\ aa\ r\ t\ m\ ax\ n\ t$
  - $cmu \rightarrow s\ iy\ eh\ m\ y\ uw$
Training LTS models

- Use CART trees
  - One model for each letter
- Predict phone (epsilon, phone, dual phone)
  - From letter 3-context (and POS)
  - # # # c h e c c -> ch
  - # # c h e c k e k -> _
  - # c h e c k e -> eh
  - c h e c k e d -> k
LTS results

- Split lexicon into train/test 90%/10%
  - i.e. every tenth entry is extracted for testing

<table>
<thead>
<tr>
<th>Lexicon</th>
<th>Letter Acc</th>
<th>Word Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>OALD</td>
<td>95.80%</td>
<td>75.56%</td>
</tr>
<tr>
<td>CMUDICT</td>
<td>91.99%</td>
<td>57.80%</td>
</tr>
<tr>
<td>BRULEX</td>
<td>99.00%</td>
<td>93.03%</td>
</tr>
<tr>
<td>DE-CELEX</td>
<td>98.79%</td>
<td>89.38%</td>
</tr>
<tr>
<td>Thai</td>
<td>95.60%</td>
<td>68.76%</td>
</tr>
</tbody>
</table>
For letter V:
if (n.name is v)
    return 
    if (n.name is #)
        if (p.p.name is t)
            return f
        return v
    if (n.name is s)
        if (p.p.p.name is n)
            return f
        return v
    return v
But we need more than phones

- What about lexical stress
  - pr aal j eh k t -> pr aal j eh1 k t
- Two possibilities
  - A separate prediction model
  - Join model – introduce eh/eh1 (BETTER)

<table>
<thead>
<tr>
<th></th>
<th>LTP+S</th>
<th>LTPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>L no S</td>
<td>96.36%</td>
<td>96.27%</td>
</tr>
<tr>
<td>Letter</td>
<td>---</td>
<td>95.80%</td>
</tr>
<tr>
<td>W no S</td>
<td>76.92%</td>
<td>74.69%</td>
</tr>
<tr>
<td>Word</td>
<td>63.68%</td>
<td>74.56%</td>
</tr>
</tbody>
</table>
Does it really work

- 40K words from Time Magazine
  - 1775 (4.6%) not in OALD
  - LTS gets 70% correct (test set was 74%)

<table>
<thead>
<tr>
<th></th>
<th>Occurs</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Names</td>
<td>1360</td>
<td>76.6</td>
</tr>
<tr>
<td>Unknown</td>
<td>351</td>
<td>19.8</td>
</tr>
<tr>
<td>US Spelling</td>
<td>57</td>
<td>3.2</td>
</tr>
<tr>
<td>Typos</td>
<td>7</td>
<td>0.4</td>
</tr>
</tbody>
</table>
Prosody Modeling

- **Phrasing**
  - Where to take breaths

- **Intonation**
  - Where (and what size) are accents
  - F0 realization

- **Duration**
  - What is the length of each phoneme
Intonation Contour
Unit Selection
The “standard” method
“Select appropriate sub-word units from large databases of natural speech”

Parametric Synthesis: [NITECH: Tokuda et al]
HMM-generation based synthesis
Cluster units to form models
Generate from the models
“Take ‘average’ of units”
Unit Selection

• Target cost and Join cost [Hunt and Black 96]
  – Target cost is distance from desired unit to actual unit in the databases
    • Based on phonetic, prosodic metrical context
  – Join cost is how well the selected units join
“Hunt and Black” Costs

Target distance is:
- $C^t(t_i, u_i) = \sum_{j=1}^{p} w_j^t C_j^t(t_i, u_i)$

For examples in the database we can measure
- $AC^t(t_i, u_i)$

Therefore estimate $w_{1-j}$ from all examples of
- $AC^t(t_i, u_i) \approx \sum_{j=1}^{p} w_j^t C_j^t(t_i, u_i)$

Use linear regression

How well does it join:
- $C^c(u_{i-1}, u_i) = \sum_{k=1}^{p} w_k^c C_k^c(u_{i-1}, u_i)$
- if $(u_{i-1} == \text{prev}(u_i)) \ C^c = 0$
Find best path of units through db that minimise:
\[ C(t^n_1, u^n_1) = \sum_{i=1}^{n} C^t(t_i, u_i) + \sum_{i=2}^{n} C^c(u_{i-1}, u_i) + C^c(S, u_1) + C^c(u_n, S) \]

- Use Viterbi to find best set of units
- Note
  - Finding “longest” is typically not optimal
Clustering Units

- Cluster units [Donovan et al 96, Black et al 97]

\[
Adist(U, V) = \begin{cases} 
\text{if } |V| > |U| & Adist(V, U) \\
WD \cdot |U| & \sum_{i=1}^{n} \frac{W_j \cdot (\text{abs}(F_{ij}(U) - F_{ij}(V)))}{SD_j \cdot n \cdot |U|}
\end{cases}
\]

- Moves calculation to compile time

\(|U| = \text{number of frames in } U\\nF_{xy}(U) = \text{parameter } y \text{ of frame } x \text{ of unit } U\\nSD_j = \text{standard deviation of parameter } j\\nW_j = \text{weight for parameter } j\\WD = \text{duration penalty}\\\]
Unit Selection Issues

- Cost metrics
  - Finding best weights, best techniques etc
- Database design
  - Best database coverage
- Automatic labeling accuracy
  - Finding errors/confidence
- Limited domain:
  - Target the databases to a particular application
  - Talking clocks
  - Targeted domain synthesis
Old vs New

Unit Selection:  
large carefully labelled database  
quality good when good examples available  
quality will sometimes be bad  
no control of prosody

Parametric Synthesis:  
smaller less carefully labelled database  
quality consistent  
resynthesis requires vocoder, (buzzy)  
can (must) control prosody  
model size much smaller than Unit DB
Parametric Synthesis

• Probabilistic Models
  \[
  \arg\max(P(O|W))
  \]

• Simplification
  \[
  \arg\max(P(o_0|W), P(o_1|W), \ldots, P(o_n|W))
  \]

• Generative model
  – Predict acoustic frames from text
Frame (State) based prediction
- Ignores dynamics

Various solutions
- MLPG (maximum likelihood parameter generation)
- Trajectory HMMs
- Global Variance
- MGE, minimal generation error
**HTS (NITECH)**
- Based on HTK
- Predicts HMM-states
- (Default) uses MCEP and MLSA filter
- Supported in Festival

**Clustergen (CMU)**
- No use of HTK
- Predicts Frames
- (Default) uses MCEP and MLSA filter
- More tightly coupled with Festival
Synthesizer

Requires:
  Prompt transcriptions (txt.done.data)
  Waveform files (well recorded)

FestVox Labelling
  EHMM (Kishore)
  Context Independent models and forced alignment
  (Have used Janus labels too).

Parameter extraction:
  (HTS’s) melcep/mlsa filter for resynthesis
  F0 extraction

Clustering
  Wagon vector clustering
  for each HMM-state name
Clustering by CART

Update to Wagon (Edinburgh Speech Tools).
   Tight coupling of features with FestVox utts
Support for arbitrary vectors
Define impurity on clusters of $N$ vectors

$$
\left( \sum_{i=1}^{24} \sigma_i \right) \ast N
$$

Clustering
   F0 and MCEP
   Tested jointly and separately
Features for clustering (51):
   phonetic, syllable, phrasal context
Three models:
- Spectral (MCEP) CART tree
- F0 CART tree
- Duration CART tree

F0 model:
- Smoothed extracted F0 through all speech (i.e. unvoiced regions get F0 values)
- Chose voicing at runtime phonetically
CLUSTERGEN Synthesis

Generate phoneme strings (as before)

For each phone:
  - Find HMM-state names: ah_1, ah_2, ah_3
  - Predict duration of each
  - Create empty mcep vector to fill duration
  - Predict mcep values from cluster tree
  - Predict F0 value from cluster tree

Use MLSA filter to regenerate speech
Objective Score

**CLUSTERGEN**

Mean Mel Cepstral Distortion over test set

$$
\frac{10}{\ln 10} \sqrt{2 \sum_{d=1}^{24} \left( mc_d^{(t)} - mc_d^{(e)} \right)^2}
$$

**MCD:** Voice Conversion ranges 4.5-6.0

**MCD:** CG scores 4.0-8.0

*smaller is better*
Example CG Voices

7 Arctic databases:
1200 utterances, 43K segs, 1hr speech

awb  bdl  
clb  jmk  
ksp  rms  
slt  

awb  bdl  
clb  jmk  
ksp  rms  
slt  
Database size vs Quality

slt_arctic data size

<table>
<thead>
<tr>
<th>Utts</th>
<th>Clusters</th>
<th>RMS F0</th>
<th>MCD</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>230</td>
<td>24.29</td>
<td>6.761</td>
</tr>
<tr>
<td>100</td>
<td>435</td>
<td>19.47</td>
<td>6.278</td>
</tr>
<tr>
<td>200</td>
<td>824</td>
<td>17.41</td>
<td>6.047</td>
</tr>
<tr>
<td>500</td>
<td>2227</td>
<td>15.02</td>
<td>5.755</td>
</tr>
<tr>
<td>1100</td>
<td>4597</td>
<td>14.55</td>
<td>5.685</td>
</tr>
</tbody>
</table>
Making it Better

- Label data, build model
- But maybe there are better labels
- So find labels that maximize model accuracy
Move Labels

Cluster trees

Predicted gaussians

Actual values
Move Labels

- Use EHMM to label segments/HMM states
- Build Clustergen Model
- Iterate
  - Predict Cluster (mean/std) for each frame
  - For each label boundary
    - If \( \text{dist}(\text{actual}_{\text{after}}, \text{pred}_{\text{before}}) < \text{dist}(\text{actual}_{\text{after}}, \text{pred}_{\text{after}}) \), move label forward
    - If \( \text{dist}(\text{actual}_{\text{before}}, \text{pred}_{\text{after}}) < \text{dist}(\text{actual}_{\text{before}}, \text{pred}_{\text{before}}) \), move label backward
Distance Metric

- **Distance from predicted to actual**
  - Euclidean
  - $F_0$, static, deltas, voicing
  - With/without standard deviation normalization
  - Weighting

- **Best choice**
  - Static without stddev normalization
  - (This is closest to MCD)
ML with 10 iterations

- rms voice (66 minutes of speech)
  - train 1019 utts, test 113 utts (every tenth)

<table>
<thead>
<tr>
<th>Pass</th>
<th>Move</th>
<th>+ve</th>
<th>-ve</th>
<th>MCD</th>
<th>stddev</th>
<th>F0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5.247</td>
<td>1.965</td>
<td>13.990</td>
</tr>
<tr>
<td>1</td>
<td>48211</td>
<td>23162</td>
<td>25949</td>
<td>5.121</td>
<td>1.846</td>
<td>14.251</td>
</tr>
<tr>
<td>2</td>
<td>40731</td>
<td>20223</td>
<td>20508</td>
<td>5.090</td>
<td>1.794</td>
<td>14.220</td>
</tr>
<tr>
<td>3</td>
<td>35059</td>
<td>17835</td>
<td>17224</td>
<td>5.073</td>
<td>1.779</td>
<td>14.267</td>
</tr>
<tr>
<td>4</td>
<td>33083</td>
<td>16503</td>
<td>16580</td>
<td>5.061</td>
<td>1.765</td>
<td>14.260</td>
</tr>
<tr>
<td>5</td>
<td>31131</td>
<td>15518</td>
<td>15613</td>
<td>5.046</td>
<td>1.753</td>
<td>14.306</td>
</tr>
<tr>
<td>6</td>
<td>29693</td>
<td>14813</td>
<td>14880</td>
<td>5.042</td>
<td>1.754</td>
<td>14.287</td>
</tr>
<tr>
<td>7</td>
<td>28361</td>
<td>14143</td>
<td>14218</td>
<td>5.042</td>
<td>1.757</td>
<td>14.240</td>
</tr>
<tr>
<td>8</td>
<td>27571</td>
<td>13730</td>
<td>13841</td>
<td>5.035</td>
<td>1.740</td>
<td>14.239</td>
</tr>
<tr>
<td>9</td>
<td>26839</td>
<td>13457</td>
<td>13382</td>
<td>5.040</td>
<td>1.750</td>
<td>14.187</td>
</tr>
</tbody>
</table>
## Move Labels

<table>
<thead>
<tr>
<th>Voice</th>
<th>2006</th>
<th>2008 base</th>
<th>2008 ml</th>
</tr>
</thead>
<tbody>
<tr>
<td>ahw</td>
<td>-</td>
<td>5.234</td>
<td>5.057</td>
</tr>
<tr>
<td>awb</td>
<td>6.557</td>
<td>4.445</td>
<td>4.483</td>
</tr>
<tr>
<td>bdl</td>
<td>6.129</td>
<td>5.685</td>
<td>5.467</td>
</tr>
<tr>
<td>clb</td>
<td>5.417</td>
<td>4.838</td>
<td>4.698</td>
</tr>
<tr>
<td>jmk</td>
<td>6.165</td>
<td>5.398</td>
<td>5.239</td>
</tr>
<tr>
<td>ksp</td>
<td>5.980</td>
<td>5.289</td>
<td>5.140</td>
</tr>
<tr>
<td>rms</td>
<td>5.731</td>
<td>5.247</td>
<td>5.035</td>
</tr>
<tr>
<td>rrx</td>
<td>-</td>
<td>5.298</td>
<td>5.160</td>
</tr>
<tr>
<td>slt</td>
<td>5.713</td>
<td>5.170</td>
<td>4.983</td>
</tr>
</tbody>
</table>

Average improvement 0.172 (excluding awb)
Does it sound better

- **rms**
  - abtest (10 utterances)
    - ml 7
    - base 1
    - $= 2$

- **Slt**
  - abtest
    - ml 7
    - base 2
    - $= 1$
Arctic MLSB improvements

Move Label Segment Boundaries  ΔMCD

Power (MCD cep0)

Spectrum (MCD 1-24)
Grapheme Based Synthesis

- **Synthesis without a phoneme set**
- **Use the letters as phonemes**
  - (“alan” nil (a l a n))
  - (“black” nil (b l a c k))
- **Spanish (easier ?)**
  - 419 utterances
  - HMM training to label databases
  - Simple pronunciation rules
  - Polici’a -> p o l i c i’ a
  - Cuatro -> c u a t r o
<table>
<thead>
<tr>
<th>Word</th>
<th>Castillian</th>
<th>gloss</th>
</tr>
</thead>
<tbody>
<tr>
<td>casa</td>
<td>/k a s a/</td>
<td>house</td>
</tr>
<tr>
<td>cesa</td>
<td>/th e s a/</td>
<td>stop</td>
</tr>
<tr>
<td>cine</td>
<td>/th i n e/</td>
<td>cinema</td>
</tr>
<tr>
<td>cosa</td>
<td>/k o s a/</td>
<td>thing</td>
</tr>
<tr>
<td>cuna</td>
<td>/k u n a/</td>
<td>cradle</td>
</tr>
<tr>
<td>hechizo</td>
<td>/e ch i th o/</td>
<td>charm, spell</td>
</tr>
</tbody>
</table>

In Spanish the letter “c” may be pronounced /k/, /ch/ and /th/ or /s/ (depending on dialect). The choice of phone is determined by the letter context.
English Grapheme Synthesis

- Use Letters are phones
- 26 “phonemes”
  - (“alan” n (a l a n))
  - (“black” n (b l a c k))
- Build HMM acoustic models for labeling
- For English
  - “This is a pen”
  - “We went to the church at Christmas”
  - Festival intro
  - “do eight meat”
- Requires method to fix errors
  - Letter to letter mapping
Common Data Sets

- **Data drive techniques need data**
- **Diphone Databases**
  - CSTR and CMU US English Diphone sets (kal and ked)
- **CMU ARCTIC Databases**
  - 1200 phonetically balanced utterances (about 1 hour)
  - 7 different speakers (2 male 2 female 3 accented)
  - EGG, phonetically labeled
  - Utterances chosen from out-of-copyright text
  - Easy to say
  - Freely distributable
  - Tools to build your own in your own language
Blizzard Challenge

- **Realistic evaluation**
  - Under the same conditions

- **Blizzard Challenge [Black and Tokuda]**
  - Participants build voice from common dataset
  - Synthesis test sentences
  - Large set of listening experiments
  - Since 2005, now in 7th year
  - 18 groups in 2010
  - Audio books in 2012
How to test synthesis

- **Blizzard tests:**
  - **Do you like it?** (MOS scores)
  - **Can you understand it?**
    - SUS sentence
    - The unsure steaks overcame the zippy rudder

- **Can’t this be done automatically?**
  - Not yet (at least not reliably enough)
  - But we now have lots of data for training techniques

- **Why does it still sound like robot?**
  - Need better (appropriate testing)
SUS Sentences

- sus_00022
- sus_00012
- sus_00005
- sus_00017
SUS Sentences

- The serene adjustments foresaw the acceptable acquisition
- The temperamental gateways forgave the weatherbeaten finalist
- The sorrowful premieres sang the ostentatious gymnast
- The disruptive billboards blew the sugary endorsement
Voice Identity

What makes a voice identity

- **Lexical Choice**:
  - Woo-hoo,
  - I pity the fool ...

- **Phonetic choice**

- **Intonation and duration**

- **Spectral qualities (vocal tract shape)**

- **Excitation**
Voice Conversion techniques

- **Full ASR and TTS**
  - Much too hard to do reliably
- **Codebook transformation**
  - ASR HMM state to HMM state transformation
- **GMM based transformation**
  - Build a mapping function between frames
Learning VC models

- **First need to get parallel speech**
  - Source and Target say same thing
  - Use DTW to align (in the spectral domain)
  - Trying to learn a functional mapping
  - 20-50 utterances

- **“Text-independent” VC**
  - Means no parallel speech available
  - Use some form of synthesis to generate it
VC Training process

- Extract F0, power and MFCC from source and target utterances
- DTW align source and target
- Loop until convergence
  - Build GMM to map between source/target
  - DTW source/target using GMM mapping
VC Training process

Source F0 → log → Compute Means And Std. Devs.
Target F0 → log → Compute Means And Std. Devs.

Source Speaker Filter Features

Add Dynamic Features → Power Threshold → DTW

Target Speaker Filter Features

Add Dynamic Features → Power Threshold → Train GMM W/EM

Iterate
### Voice Transformation

- **Festvox GMM transformation suite (Toda)**

<table>
<thead>
<tr>
<th></th>
<th>awb</th>
<th>bdl</th>
<th>jmk</th>
<th>slt</th>
</tr>
</thead>
<tbody>
<tr>
<td>awb</td>
<td>🎧</td>
<td>🎧</td>
<td>🎧</td>
<td>🎧</td>
</tr>
<tr>
<td>bdl</td>
<td>🎧</td>
<td>🎧</td>
<td>🎧</td>
<td>🎧</td>
</tr>
<tr>
<td>jmk</td>
<td>🎧</td>
<td>🎧</td>
<td>🎧</td>
<td>🎧</td>
</tr>
<tr>
<td>slt</td>
<td>🎧</td>
<td>🎧</td>
<td>🎧</td>
<td>🎧</td>
</tr>
</tbody>
</table>
VC in Synthesis

- Can be used as a post filter in synthesis
  - Build kal_diphone to target VC
  - Use on all output of kal_diphone

- Can be used to convert a full DB
  - Convert a full db and rebuild a voice
Style/Emotion Conversion

- **Unit Selection (or SPS)**
  - Require lots of data in desired style/emotion

- **VC technique**
  - Use as filter to main voice (same speaker)
  - Convert neutral to angry, sad, happy …
Can you say that again?

- **Voice conversion for speaking in noise**
- **Different quality when you repeat things**
- **Different quality when you speak in noise**
  - Lombard effect (when very loud)
  - “Speech-in-noise” in regular noise
Collect data
- Randomly play noise in person’s ears
- Normal
- In Noise

Collect 500 of each type

Build VC model
- Normal -> in-Noise

Actually
- Spectral, duration, f0 and power differences
Synthesis in Noise

- For bus information task
- **Play different synthesis information utts**
  - With SIN synthesizer
  - With SWN synthesizer
  - With VC (SWN→SIN) synthesizer
- **Measure their understanding**
  - SIN synthesizer better (in Noise)
  - SIN synthesizer better (without Noise for elderly)
Transterpolation

- Incrementally transform a voice $X\%$
  - BDL-SLT by 10%
  - SLT-BDL by 10%

- Count when you think it changes from M-F

- Fun but what are the uses …
De-identification

- **Remove speaker identity**
  - But keep it still human like

- **Health Records**
  - HIPAA laws require this
  - Not just removing names and SSNs

- **Remove identifiable properties**
  - Use Voice conversion to remove spectral
  - Use F0/duration mapping to remove prosodic
  - Use ASR/MT techniques to remove lexical
Summary

- **Data-driven speech synthesis**
  - Text processing
  - Prosody and pronunciation
  - Waveform synthesis

- **Finding the right optimization**
  - Find an objective metric that correlates with human perception