Artificial Neural Network-Based Speech Synthesis

Heiga Zen

July 28th, 2016@SPCC
Outline

Quick recap of HMM-based SPSS
  Training & synthesis
  Limitations

ANN-based SPSS
  Feedforward NN
  Recurrent NN

Google industry application experience
  Deploy LSTM-RNN-based SPSS to products
What are you going to learn?

- Quick recap of HMM-based statistical parametric speech synthesis
  - Training & generation
  - Limitations

- Speech synthesis by ANN
  - Motivations
  - Feed-forward NN
  - Recurrent NN

- Google industry application experience
  - Deploy LSTM-RNN-based SPSS into Google products
  - Voice morphing (by Yannis Agiomyrgiannakis)
Statistical parametric speech synthesis (SPSS) [2]

1. Extract \{acoustic, linguistic\} features from \{speech, text\}
2. Model relationship between linguistic & acoustic features
3. Predict acoustic features from linguistic features
4. Reconstruct speech from acoustic features

SPSS can use any acoustic model, but HMM-based one is very popular → HMM-based speech synthesis [1]
Statistical parametric speech synthesis (SPSS) [2]

Pros

• Small footprint
• Flexibility
• Robustness

Cons

• Segmental naturalness
Major factors for naturalness degradation

- **Vocoder analysis/synthesis**
  - *How to parameterize speech?*

- **Acoustic model**
  - *How to represent relationship between speech & text?*

- **Oversmoothing**
  - *How to generate speech from model?*
Formulation of SPSS

Training

- Extract linguistic features $l$ & acoustic features $o$
- Train acoustic model $\Lambda$ given $(o, l)$

$$\hat{\Lambda} = \arg \max_{\Lambda} p(o \mid l, \Lambda)$$

Synthesis

- Extract $l$ from text to be synthesized
- Generate most probable $o$ from $\hat{\Lambda}$ then reconstruct waveform

$$\hat{o} = \arg \max_{o} p(o \mid l, \hat{\Lambda})$$
Formulation of SPSS

Training

- Extract linguistic features $l$ & acoustic features $o$
- Train acoustic model $\Lambda$ given $(o, l)$

$$\hat{\Lambda} = \arg \max_{\Lambda} p(o \mid l, \Lambda)$$

Synthesis

- Extract $l$ from text to be synthesized
- Generate most probable $o$ from $\hat{\Lambda}$ then reconstruct waveform

$$\hat{o} = \arg \max_{o} p(o \mid l, \hat{\Lambda})$$
Speech production process

- text (concept)
- modulation of carrier wave by speech information
  - frequency transfer characteristics
  - magnitude start-end
  - fundamental frequency
- air flow
- fundamental freq
- voiced/unvoiced freq transfer char
- Sound source voiced: pulse
  unvoiced: noise
Source-filter model

Source excitation part

- pulse train
- white noise

V/UV

F0

Vocal tract resonance part

- linear time-invariant system $h(n)$

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

$X(e^{j\omega}) = H(e^{j\omega})E(e^{j\omega})$

$H\left(e^{j\omega}\right)$ should be defined by HMM state-output vectors

e.g., mel-cepstrum, line spectral pairs
Structure of state-output (observation) vectors

\[ O_t \]

- **Vocal tract part**
  - \( c_t \)
  - \( \Delta c_t \)
  - \( \Delta^2 c_t \)

- **Source excitation part**
  - \( p_t \)
  - \( \Delta p_t \)
  - \( \Delta^2 p_t \)

- Mel-cepstral coefficients
- \( \Delta \) Mel-cepstral coefficients
- \( \Delta \Delta \) Mel-cepstral coefficients
- \( \log F0 \) (log(0) when unvoiced)
- \( \Delta \log F0 \) (log(0) when unvoiced)
- \( \Delta \Delta \log F0 \) (log(0) when unvoiced)
Training – HMM-based acoustic modeling

$$p(o \mid l, \Lambda) = \sum_{\forall q} p(o \mid q, \Lambda)P(q \mid l, \Lambda) \quad q: \text{hidden states}$$

$$= \sum_{\forall q} \prod_{t=1}^{T} p(o_t \mid q_t, \Lambda)P(q \mid l, \Lambda) \quad q_t: \text{hidden state at } t$$

$$= \sum_{\forall q} \prod_{t=1}^{T} \mathcal{N}(o_t; \mu_{qt}, \Sigma_{qt})P(q \mid l, \Lambda)$$

ML estimation of HMM parameters → Baum-Welch (EM) algorithm [3]
Training – Linguistic features

Linguistic features: phonetic, grammatical, & prosodic features

- **Phoneme**
  phoneme identity, position

- **Syllable**
  length, accent, stress, tone, vowel, position

- **Word**
  length, POS, grammar, prominence, emphasis, position, pitch accent

- **Phrase**
  length, type, position, intonation

- **Sentence**
  length, type, position

... → Impossible to have enough data to cover all combinations
Training – Example

Heiga Zen
Artificial Neural Network-Based Speech Synthesis
July 28th, 2016
Formulation of SPSS

Training
- Extract linguistic features $l$ & acoustic features $o$
- Train acoustic model $\Lambda$ given $(o, l)$

$$\hat{\Lambda} = \arg \max_{\Lambda} p(o \mid l, \Lambda)$$

Synthesis
- Extract $l$ from text to be synthesized
- Generate most probable $o$ from $\hat{\Lambda}$ then reconstruct waveform

$$\hat{o} = \arg \max_{o} p(o \mid l, \hat{\Lambda})$$
Synthesis – Predict most probable acoustic features

\[
\hat{o} = \arg \max_{o} p(o \mid l, \hat{\Lambda})
\]

\[
= \arg \max_{o} \sum_{q} p(o, q \mid l, \hat{\Lambda})
\]

\[
\approx \arg \max_{o} \max_{q} p(o, q \mid l, \hat{\Lambda})
\]

\[
\approx \arg \max_{o} p(o \mid \hat{q}, \hat{\Lambda}) \quad \text{s.t.} \quad \hat{q} = \arg \max_{q} P(q \mid l, \hat{\Lambda})
\]

\[
= \arg \max_{o} \mathcal{N} (o; \mu_{\hat{q}}, \Sigma_{\hat{q}})
\]

\[
= \mu_{\hat{q}}
\]

\[
= \begin{bmatrix} \mu_{\hat{q}_1}^\top, \ldots, \mu_{\hat{q}_T}^\top \end{bmatrix}^\top
\]
\( \hat{o} \rightarrow \text{step-wise} \rightarrow \text{discontinuity can be perceived} \)

\[ \mathbf{O}_t = \begin{bmatrix} \mathbf{c}_t^\top, \Delta \mathbf{c}_t^\top \end{bmatrix}^\top \]

\[ \Delta \mathbf{c}_t = \mathbf{c}_t - \mathbf{c}_{t+1} \]

\begin{align*}
\mathbf{c}_t & \quad \mathbf{c}_{t-1} \\
\Delta \mathbf{c}_t & \quad \Delta \mathbf{c}_{t-1} \\
\mathbf{c}_{t+1} & \quad \mathbf{c}_{t+2} \\
\Delta \mathbf{c}_{t+1} & \quad \Delta \mathbf{c}_{t+2}
\end{align*}
\[
\hat{o} = \arg \max_{o} p(o \mid \hat{q}, \hat{\Lambda}) \quad s.t. \quad o = Wc \\
\hat{c} = \arg \max_{c} \mathcal{N}(Wc; \mu\hat{q}, \Sigma\hat{q}) \\
= \arg \max_{c} \log \mathcal{N}(Wc; \mu\hat{q}, \Sigma\hat{q}) \\
\frac{\partial}{\partial c} \log \mathcal{N}(Wc; \mu\hat{q}, \Sigma\hat{q}) \propto W^\top \Sigma^{-1}\hat{q} Wc - W^\top \Sigma^{-1}\mu\hat{q} \\
W^\top \Sigma^{-1}\hat{q} Wc = W^\top \Sigma^{-1}\mu\hat{q}
\]

where

\[
\mu_q = [\mu_{q1}^\top, \mu_{q2}^\top, \ldots, \mu_{qT}^\top]^\top \\
\Sigma_q = \text{diag} [\Sigma_{q1}, \Sigma_{q2}, \ldots, \Sigma_{qT}]
\]
Synthesis – Most probable acoustic features under constraints between static & dynamic features

Static

Dynamic

Mean

Variance

$\hat{c}$
Waveform reconstruction

- Pulse train
- White noise
- Excitation
- Generated excitation parameter (log F0 with V/UV)
- Generated spectral parameter (cepstrum, LSP)
- Linear time-invariant system
  \[ x(n) = h(n) \ast e(n) \]
- Synthesized speech
Output probability only depends on the current state

Within the same state, statistics are constant

→ Step-wise statistics

Using dynamic feature constraints

→ Ad hoc & introduces inconsistency betw. training & synthesis [6]
HMM-based acoustic model – Limitations (2)

Difficulty to integrate feature extraction & modeling

- Spectra or waveforms are high-dimensional & highly correlated
- Hard to be modeled by HMMs with Gaussian + digonal covariance

→ Use low dimensional approximation (e.g., cepstra, LSPs)
HMM-based acoustic model – Limitations (3)
Data fragmentation

- Trees split input into clusters & put representative distributions
  → Inefficient to represent dependency betw. ling. & acoust. feats.
- Minor features are never used (e.g., word-level emphasis [7])
  → Little or no effect
Alternatives – Stepwise statistics

- Autoregressive HMMs (ARHMMs) [8]
- Linear dynamical models (LDMs) [9, 10]
- Trajectory HMMs [6]
- …

Most of them use clustering → Data fragmentation
Often employ trees from HMM → Sub-optimal
Alternatives – Difficulty to integrate feature extraction

- Statistical vocoder [11]
- Minimum generation error with log spectral distortion [12]
- Waveform-level model [13]
- Mel-cepstral analysis-integrated HMM [14]

Use clustering to build tying structure → Data fragmentation
Often employ trees from HMM → Sub-optimal
Alternatives – Data fragmentation

- Factorized decision tree [7, 15]
- Product of experts [16]

Each tree/expert still has data fragmentation → Data fragmentation
Fix other trees while building one tree [17, 18] → Sub-optimal
Linguistic → Acoustic mapping

- **Training**
  Learn relationship between linguistic & acoustic features

- **Synthesis**
  Map linguistic features to acoustic ones

- **Linguistic features used in SPSS**
  - Phoneme, syllable, word, phrase, utterance-level features
  - Around 50 different types
  - Sparse & correlated

**Effective modeling is essential**
Decision tree-based acoustic model

HMM-based acoustic model & alternatives
→ Actually decision tree-based acoustic model

Regression tree: linguistic features → Stats. of acoustic features

Replace the tree with a general-purpose regression model
→ Artificial neural network
ANN-based acoustic model [19] – Overview

\[ h_t = f (W_{hl}l_t + b_h) \]
\[ \hat{o}_t = W_{oh}h_t + b_o \]
\[ \hat{\Lambda} = \arg\min_{\Lambda} \sum_t \|o_t - \hat{o}_t\|_2 \quad \Lambda = \{W_{hl}, W_{oh}, b_h, b_o\} \]

\[ \hat{o}_t \approx \mathbb{E} [o_t \mid l_t] \rightarrow \text{Replace decision trees & Gaussian distributions} \]
ANN-based acoustic model [19] – Motivation (1)

Distributed representation [20, 21]

- Fragmented: \( n \) terminal nodes \( \rightarrow \) \( n \) classes (linear)
- Distributed: \( n \) binary units \( \rightarrow \) \( 2^n \) classes (exponential)
- Minor features (e.g., word-level emphasis) can affect synthesis
ANN-based acoustic model [19] – Motivation (2)

Integrate feature extraction [22, 23, 24]

- Layered architecture with non-linear operations
- Can model high-dimensional/correlated linguistic/acoustic features

→ Feature extraction can be embedded in model itself
ANN-based acoustic model [19] – Motivation (3)
Implicitly mimic layered hierarchical structure in speech production

Concept → Linguistic → Articulator → Vocal tract → Waveform
DNN-based speech synthesis [19] – Implementation

Heiga Zen
Artificial Neural Network-Based Speech Synthesis
July 28th, 2016
DNN-based speech synthesis [19] – Example

Heiga Zen
Artificial Neural Network-Based Speech Synthesis
July 28th, 2016

Compared HMM- & DNN-based TTS w/ similar # of parameters

- US English, professional speaker, 30 hours of speech data
- Preference test
- 173 test sentences, 5 subjects per pair
- Up to 30 pairs per subject
- Crowd-sourced

<table>
<thead>
<tr>
<th>Preference scores (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMM</td>
</tr>
<tr>
<td>---------------</td>
</tr>
<tr>
<td>15.8</td>
</tr>
<tr>
<td>16.1</td>
</tr>
<tr>
<td>12.7</td>
</tr>
</tbody>
</table>

Heiga Zen
Artificial Neural Network-Based Speech Synthesis
July 28th, 2016
From HMMs to DNNs
Where do the improvements come from? [25]
From HMMs to DNNs
Where do the improvements come from? [25]

<table>
<thead>
<tr>
<th>System</th>
<th>Regression model</th>
<th>Regression target unit</th>
<th>Stream modelling</th>
<th>Variance</th>
<th>Duration-derived features</th>
<th>Enhancement method</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>decision tree</td>
<td>state</td>
<td>separate</td>
<td>context-dependent</td>
<td>no</td>
<td>GV</td>
</tr>
<tr>
<td>D2</td>
<td>decision tree</td>
<td>state</td>
<td>separate</td>
<td>context-dependent</td>
<td>no</td>
<td>postfilter</td>
</tr>
<tr>
<td>N1</td>
<td>neural network</td>
<td>state</td>
<td>separate</td>
<td>context-dependent</td>
<td>no</td>
<td>postfilter</td>
</tr>
<tr>
<td>N2</td>
<td>neural network</td>
<td>state</td>
<td>separate</td>
<td>fixed</td>
<td>no</td>
<td>postfilter</td>
</tr>
<tr>
<td>N3</td>
<td>neural network</td>
<td>state</td>
<td>combined</td>
<td>fixed</td>
<td>no</td>
<td>postfilter</td>
</tr>
<tr>
<td>N4</td>
<td>neural network</td>
<td>frame</td>
<td>separate</td>
<td>fixed</td>
<td>no</td>
<td>postfilter</td>
</tr>
<tr>
<td>N5</td>
<td>neural network</td>
<td>frame</td>
<td>combined</td>
<td>fixed</td>
<td>no</td>
<td>postfilter</td>
</tr>
<tr>
<td>N6</td>
<td>neural network</td>
<td>frame</td>
<td>combined</td>
<td>fixed</td>
<td>yes</td>
<td>postfilter</td>
</tr>
</tbody>
</table>
From HMMs to DNNs
Where do the improvements come from? [25]
From HMMs to DNNs
Where do the improvements come from? [25]

- **Significant improvement**
  - Regression model: Decision tree $\rightarrow$ Neural network
  - Model unit: State $\rightarrow$ Frame
  - Using duration-derived features

- **No significant difference**
  - Separate stream $\rightarrow$ Combined stream
  - Context-dependent variance $\rightarrow$ Context-independent variance
  - GV $\rightarrow$ Postfiltering
Other applications of DNNs for SPSS

- **Integrate acoustic feature extraction**
  - Use auto-encoder bottleneck features of raw spectra [26, 27]
  - Integrate maximum-likelihood cepstral analysis into loss [24, 28]

- **Use DNN with sinusoidal vocoder** [29]
  - Use sinusoidal vocoder parameters with multi-style learning

- **Use DNN instead of HMM for guiding unit selection** [30]
  - DNNs give targets for unit selection
Feedforward NN-based acoustic model – Limitation

Each frame is mapped independently → **Smoothing is still essential**

<table>
<thead>
<tr>
<th>Preference scores (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN with dyn</td>
</tr>
<tr>
<td>67.8</td>
</tr>
</tbody>
</table>
Add dependency between frames

- Minimum generation error training of DNN [31]
  → Train DNN under constraints between static & dynamic features
- Recurrent NN (RNN) [32]
  → Incorporate dynamics into neural network itself
RNN-based acoustic model [33, 34]

\[ h_t = f \left( W_{hl} l_t + W_{hh} h_{t-1} + b_h \right) \]
\[ \hat{o}_t = W_{oh} h_t + b_o \]
\[ \hat{\Lambda} = \arg\min_\Lambda \sum_t \| o_t - \hat{o}_t \|_2 \quad \Lambda = \{ W_{hl}, W_{hh}, W_{oh}, b_h, b_o \} \]

- **DNN**: \( \hat{o}_t \approx \mathbb{E} \left[ o_t \mid l_t \right] \)
- **RNN**: \( \hat{o}_t \approx \mathbb{E} \left[ o_t \mid l_1, \ldots, l_t \right] \)
RNN-based acoustic model [33, 34]

- Only able to use previous contexts
  → Bidirectional RNN [32]: $\hat{o}_t \approx \mathbb{E} [o_t | l_1, \ldots, l_T]$

- Trouble accessing long-range contexts
  - Information in hidden layers loops quickly decays over time
  - Prone to being overwritten by new information from inputs
  → Long short-term memory (LSTM) [35]
LSTM unit

LSTM → NN-representation of random access memory

Gate output: 0 -- 1

Input gate == 1 → Write memory

Forget gate == 0 → Reset memory

Output gate == 1 → Read memory

- Gates in a LSTM unit: 0/1 switch controlling information flow
- Can have longer memory

Heiga Zen
Artificial Neural Network-Based Speech Synthesis
July 28th, 2016
47 of 83
LSTM-RNN-based SPSS [33, 34]

Waveform

Acoustic features (outputs)

Acoustic feature prediction LSTM

Linguistic features (frame)

Durations (targets)

Duration prediction LSTM

Linguistic features (phoneme)

phoneme

h e l ou

syllable

h e2 l ou1

word

hello

Linguistic Structure

Heiga Zen

Artificial Neural Network-Based Speech Synthesis

July 28th, 2016
Unidirectional LSTM-RNN-based SPSS [34]

Subjective preference test (same US English data)

DNN: 3 layers, 1024 units
LSTM: 1 layer, 256 LSTM units

<table>
<thead>
<tr>
<th></th>
<th>DNN with dyn</th>
<th>LSTM with dyn</th>
<th>No pref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN with dyn</td>
<td>18.4</td>
<td>34.9</td>
<td>47.6</td>
</tr>
<tr>
<td>LSTM with dyn</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSTM with dyn</td>
<td>21.0</td>
<td>12.2</td>
<td>66.8</td>
</tr>
<tr>
<td>LSTM without dyn</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

→ Smoothing was still effective
Why?

Gates in LSTM units: 0/1 switch controlling information flow
Can produce rapid change in outputs
→ Discontinuity

Gate output: 0 → 1
Input gate == 1
→ Write memory
Forget gate == 0
→ Reset memory
Output gate == 1
→ Read memory

Heiga Zen
Artificial Neural Network-Based Speech Synthesis
July 28th, 2016  50 of 83
How?

- Using loss function incorporating continuity
- Integrate smoothing $\rightarrow$ Recurrent output layer [34]

$$h_t = \text{LSTM}(l_t) \quad \hat{o}_t = W_{oh}h_t + W_{oo}\hat{o}_{t-1} + b_o$$

Works pretty well

<table>
<thead>
<tr>
<th></th>
<th>LSTM with dyn (Feedforward)</th>
<th>LSTM without dyn (Recurrent)</th>
<th>No pref.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>21.8</td>
<td>21.0</td>
<td>57.2</td>
</tr>
</tbody>
</table>

Having two smoothing together doesn’t work well $\rightarrow$ Oversmoothing?

<table>
<thead>
<tr>
<th></th>
<th>LSTM with dyn (Recurrent)</th>
<th>LSTM without dyn (Recurrent)</th>
<th>No pref.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>16.6</td>
<td>29.2</td>
<td>54.2</td>
</tr>
</tbody>
</table>
Other types & applications of RNNs for SPSS

- **Clockwork RNN** [36, 37]
  - Simple recurrent units running at different clock rate
  - Can learn short and long-term dependency

- **Simplified LSTM & gated recurrent unit (GRU)** [38]
  - Simpler structure than LSTM unit
  - Less computation & similar performance

- **Use RNN as a postfilter** [39]
  - RNN to learn mapping from Clustergen outputs to natural speech
ANN-based speech synthesis – Summary

- **HMM**
  - Discontinuity due to step-wise statistics
  - Difficult to integrate feature extraction
  - Fragmented representation

- **Feedforward NN**
  - Easier to integrate feature extraction
  - Distributed representation
  - Discontinuity due to frame-by-frame independent mapping

- **Recurrent NN**
  - Smooth
Thanks!
Deploy LSTM-RNN-based SPSS to products

Client-side (local) TTS for Android

Google Text-to-speech

Google Inc., Tools

PEGI 3

This app is compatible with all of your devices.

Installed

Heiga Zen

Artificial Neural Network-Based Speech Synthesis

July 28th, 2016
Requirements on mobile devices

- Totally run on a device (no network)
- Small footprint & RAM
- Low CPU & buttery usage
- Low latency (essential for accessibility users)
LSTM-RNN-based SPSS [33, 34]
Low-latency TTS by unidirectional LSTM-RNN [34]

HMM / DNN

- Smoothing by dyn. needs to solve set of $T$ linear equations

$$W^\top \Sigma_{\hat{q}}^{-1} W c = W^\top \Sigma_{\hat{q}}^{-1} \mu_{\hat{q}}$$

$T$: Utterance length

- Order of operations to determine the first frame $c_1$ (latency)
  - Cholesky decomposition [5] $\rightarrow \mathcal{O}(T)$
  - Recursive approximation [40] $\rightarrow \mathcal{O}(L)$
    $L$: lookahead, $10 \sim 30$

Unidirectional LSTM with recurrent output layer [34]

- No smoothing required, fully time-synchronous w/o lookahead
- Order of latency $\rightarrow \mathcal{O}(1)$
Low-latency TTS by LSTM-RNN [34] – Implementation

Linguistic Structure

Acoustic feature prediction LSTM

Linguistic features (phoneme)

Acoustic features (outputs)

Linguistic features (frame)

Durations (targets)

Duration prediction LSTM

phoneme

syllable

word

Waveform

9

12

10

10

⇒

⇒

⇒

⇒

9

12

10

10

h

e

l

ou

h e2

l ou1

hello

Linguistic Structure

Heiga Zen

Artificial Neural Network-Based Speech Synthesis

July 28th, 2016
Network architecture

49 dense output

⇐ Encourage smooth trajectory

RNN / Linear

LSTMP

LSTMP

LSTMP

FF / ReLU

⇐ Embed to continuous space

~ 400 sparse input
Results – HMM / LSTM-RNN

Subjective 5-scale Mean Opinion Score test (i18n)

Better

5-scale MOS

HMM  LSTM-RNN

Results – HMM / LSTM-RNN

Subjective preference test (i18n)

Preference scores (%)

HMM
LSTM-RNN
Better
## Results – HMM / LSTM-RNN
### Latency & Battery/CPU usage

#### Latency (Nexus 7 2013)

<table>
<thead>
<tr>
<th>Sentence</th>
<th>HMM</th>
<th>LSTM-RNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>very short (1 character)</td>
<td>26/30</td>
<td>37/72</td>
</tr>
<tr>
<td>short (~30 characters)</td>
<td>123/172</td>
<td>63/88</td>
</tr>
<tr>
<td>long (~80 characters)</td>
<td>311/418</td>
<td>118/190</td>
</tr>
</tbody>
</table>
## Results – HMM / LSTM-RNN

### Real-time ratio / Power usage

#### Real-time ratio

<table>
<thead>
<tr>
<th>Speaking rates</th>
<th>Real-time ratio HMM</th>
<th>Real-time ratio LSTM-RNN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1×</td>
<td>0.18</td>
<td>0.32</td>
</tr>
<tr>
<td>2×</td>
<td>0.18</td>
<td>0.44</td>
</tr>
<tr>
<td>3×</td>
<td>0.16</td>
<td>0.59</td>
</tr>
</tbody>
</table>

#### CPU usage

HMM $\rightarrow$ LSTM-RNN: $+48\%$

#### Battery usage (Daily usage by a blind Googler)

HMM: $2.8\%$ of 1475 mAH $\rightarrow$ LSTM-RNN: $4.8\%$ of 1919 mAH
Results – HMM / LSTM-RNN

Summary

- **Naturalness**
  
  LSTM-RNN > HMM

- **Latency**
  
  LSTM-RNN < HMM

- **CPU/Battery usage**
  
  LSTM-RNN > HMM

- **Footprint**
  
  LSTM-RNN ≈ HMM

This version was released to limited devices in July 2015
Further optimization

- **Disk footprint**
  HMM $\rightarrow$ 8-bit quantized [41]
  RNN $\rightarrow$ Float
  $\quad\rightarrow$ *Weight quantization*

- **Computational cost at inference**
  HMM $\rightarrow$ Traversing decision trees (state) + parameter generation
  RNN $\rightarrow$ Matrix-Vector multiplication (frame)
  $\quad\rightarrow$ *Multi-frame inference*

- **Robustness**
  HMM $\rightarrow$ “Soft” alignments using the Baum-Welch algorithm
  RNN $\rightarrow$ Typically relies on fixed alignments [19]
  $\quad\rightarrow$ $\epsilon$-contaminated *Gaussian loss function*

Detail: [https://arxiv.org/abs/1606.06061](https://arxiv.org/abs/1606.06061)
Weight quantization

8-bit quantization of ANN weights to reduce footprint [42]

<table>
<thead>
<tr>
<th>Language</th>
<th>Preference scores (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>int8</td>
</tr>
<tr>
<td>English (GB)</td>
<td>13.0</td>
</tr>
<tr>
<td>English (NA)</td>
<td>8.0</td>
</tr>
<tr>
<td>French</td>
<td>4.7</td>
</tr>
<tr>
<td>German</td>
<td>12.5</td>
</tr>
<tr>
<td>Italian</td>
<td>12.0</td>
</tr>
<tr>
<td>Spanish (ES)</td>
<td>8.8</td>
</tr>
</tbody>
</table>

No degradation by weight quantization
Multi-frame inference

Most of inputs are constant in a phoneme & output varies slowly
→ Bundle multiple targets to a single one [43]

Data augmentation
Multi-frame inference

4-frame inference w/ data augmentation

<table>
<thead>
<tr>
<th>Language</th>
<th>Preference scores (%)</th>
<th>4-frame+</th>
<th>1-frame</th>
<th>No pref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>English (GB)</td>
<td></td>
<td>25.7</td>
<td>20.2</td>
<td>54.2</td>
</tr>
<tr>
<td>English (NA)</td>
<td></td>
<td>8.5</td>
<td>6.2</td>
<td>85.3</td>
</tr>
<tr>
<td>French</td>
<td></td>
<td>18.8</td>
<td>18.6</td>
<td>62.6</td>
</tr>
<tr>
<td>German</td>
<td></td>
<td>19.3</td>
<td>22.2</td>
<td>58.5</td>
</tr>
<tr>
<td>Italian</td>
<td></td>
<td>13.5</td>
<td>14.4</td>
<td>72.1</td>
</tr>
<tr>
<td>Spanish (ES)</td>
<td></td>
<td>12.8</td>
<td>17.0</td>
<td>70.3</td>
</tr>
</tbody>
</table>

No degradation by multi-frame inference
$\epsilon$-contaminated Gaussian loss

Use heavier-tailed distribution as to compute loss $\rightarrow$ Robust to outliers

$$\mathcal{L}(z; x, \Lambda) = -\log \{ (1 - \epsilon) \mathcal{N}(z; f(x; \Lambda), \Sigma) + \epsilon \mathcal{N}(z; f(x; \Lambda), c\Sigma) \}$$
### Preference scores (%)

<table>
<thead>
<tr>
<th>Language</th>
<th>CG</th>
<th>L2</th>
<th>No pref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>English (GB)</td>
<td>27.4</td>
<td>18.1</td>
<td>54.5</td>
</tr>
<tr>
<td>English (NA)</td>
<td>7.6</td>
<td>6.8</td>
<td>85.6</td>
</tr>
<tr>
<td>French</td>
<td>24.6</td>
<td>15.9</td>
<td>59.5</td>
</tr>
<tr>
<td>German</td>
<td>17.1</td>
<td>20.8</td>
<td>62.1</td>
</tr>
<tr>
<td>Italian</td>
<td>16.0</td>
<td>10.6</td>
<td>73.4</td>
</tr>
<tr>
<td>Spanish (ES)</td>
<td>16.0</td>
<td>13.4</td>
<td>70.6</td>
</tr>
</tbody>
</table>
### Objective measures

- HMMs & LSTM-RNNs were quantized into 8-bit integers
- Same training data & text processing front-end
- Average disk footprint: **HMM: 1,560KB**  **LSTM-RNN: 454.5KB**
- HMM: Time-recursive parameter generation [40] w/ 10-frame delay

<table>
<thead>
<tr>
<th>Length</th>
<th>Latency (ms)</th>
<th>Total (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LSTM</td>
<td>HMM</td>
</tr>
<tr>
<td>character</td>
<td>12.5</td>
<td>19.5</td>
</tr>
<tr>
<td>word</td>
<td>14.6</td>
<td>25.3</td>
</tr>
<tr>
<td>sentence</td>
<td>31.4</td>
<td>55.4</td>
</tr>
<tr>
<td>paragraph</td>
<td>64.1</td>
<td>117.7</td>
</tr>
</tbody>
</table>
## Results – HMM / LSTM-RNN

### Subjective evaluation

<table>
<thead>
<tr>
<th>Language</th>
<th>LSTM</th>
<th>HMM</th>
<th>No pref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>English (GB)</td>
<td>31.6</td>
<td>28.1</td>
<td>40.3</td>
</tr>
<tr>
<td>English (NA)</td>
<td><strong>30.6</strong></td>
<td>15.9</td>
<td>53.5</td>
</tr>
<tr>
<td>French</td>
<td>68.6</td>
<td>8.4</td>
<td>23.0</td>
</tr>
<tr>
<td>German</td>
<td>52.8</td>
<td>19.3</td>
<td>27.9</td>
</tr>
<tr>
<td>Italian</td>
<td>84.8</td>
<td>2.9</td>
<td>12.3</td>
</tr>
<tr>
<td>Spanish (ES)</td>
<td><strong>72.6</strong></td>
<td>10.6</td>
<td>16.8</td>
</tr>
</tbody>
</table>
Results – HMM / LSTM-RNN

Summary

- **Naturalness**
  LSTM-RNN > HMM

- **Latency**
  LSTM-RNN < HMM

- **CPU/Battery usage**
  LSTM-RNN ≈ HMM

- **Footprint**
  LSTM-RNN < HMM

This version was released to most of devices in February 2016
HMM-guided unit selection / LSTM-RNN-based SPSS
Subjective preference test

<table>
<thead>
<tr>
<th>Language</th>
<th>LSTM</th>
<th>Hybrid</th>
<th>No pref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arabic</td>
<td>13.9</td>
<td>22.1</td>
<td>64.0</td>
</tr>
<tr>
<td>Cantonese</td>
<td>25.1</td>
<td>7.3</td>
<td>67.6</td>
</tr>
<tr>
<td>Danish</td>
<td>37.0</td>
<td>49.1</td>
<td>13.9</td>
</tr>
<tr>
<td>Dutch</td>
<td>29.1</td>
<td>46.8</td>
<td>24.1</td>
</tr>
<tr>
<td>English (GB)</td>
<td>22.5</td>
<td>65.1</td>
<td>12.4</td>
</tr>
<tr>
<td>English (NA)</td>
<td>23.3</td>
<td>61.8</td>
<td>15.0</td>
</tr>
<tr>
<td>French</td>
<td>28.4</td>
<td>50.3</td>
<td>21.4</td>
</tr>
<tr>
<td>German</td>
<td>20.8</td>
<td>58.5</td>
<td>20.8</td>
</tr>
<tr>
<td>Greek</td>
<td>42.5</td>
<td>21.4</td>
<td>36.1</td>
</tr>
<tr>
<td>Hindi</td>
<td>42.5</td>
<td>36.4</td>
<td>21.1</td>
</tr>
<tr>
<td>Hungarian</td>
<td>56.5</td>
<td>30.3</td>
<td>13.3</td>
</tr>
<tr>
<td>Indonesian</td>
<td>18.9</td>
<td>57.8</td>
<td>23.4</td>
</tr>
<tr>
<td>Italian</td>
<td>28.1</td>
<td>49.0</td>
<td>22.9</td>
</tr>
<tr>
<td>Japanese</td>
<td>47.4</td>
<td>28.8</td>
<td>23.9</td>
</tr>
<tr>
<td>Korean</td>
<td>40.6</td>
<td>25.8</td>
<td>33.5</td>
</tr>
<tr>
<td>Mandarin</td>
<td>48.6</td>
<td>17.5</td>
<td>33.9</td>
</tr>
<tr>
<td>Norwegian</td>
<td>54.1</td>
<td>30.8</td>
<td>15.1</td>
</tr>
<tr>
<td>Polish</td>
<td>14.6</td>
<td>75.3</td>
<td>10.1</td>
</tr>
<tr>
<td>Portuguese (BR)</td>
<td>31.4</td>
<td>37.8</td>
<td>30.9</td>
</tr>
<tr>
<td>Russian</td>
<td>26.7</td>
<td>49.1</td>
<td>24.3</td>
</tr>
<tr>
<td>Spanish (ES)</td>
<td>21.0</td>
<td>47.1</td>
<td>31.9</td>
</tr>
<tr>
<td>Spanish (NA)</td>
<td>22.5</td>
<td>55.6</td>
<td>21.9</td>
</tr>
<tr>
<td>Swedish</td>
<td>48.3</td>
<td>33.6</td>
<td>18.1</td>
</tr>
<tr>
<td>Thai</td>
<td>71.3</td>
<td>8.8</td>
<td>20.0</td>
</tr>
<tr>
<td>Turkish</td>
<td>61.3</td>
<td>20.8</td>
<td>18.0</td>
</tr>
<tr>
<td>Vietnamese</td>
<td>30.8</td>
<td>30.8</td>
<td>38.5</td>
</tr>
</tbody>
</table>
Deploy LSTM-RNN-based SPSS into products

- **Client-side**
  - Better than HMM-based SPSS in naturalness, latency, & footprint
  - Met HMM-based SPSS in CPU usage
  - Deployed in Google TTS app for Android

- **Server-side**
  - Matched HMM-guided unit selection TTS in 13 of 26 languages
  - Used in Google’s services using TTS (e.g., Now, Translate, Maps)


Kalman filter based speech synthesis.  

Linear dynamical models in speech synthesis.  

Statistical approach to vocal tract transfer function estimation based on factor analyzed trajectory hmm.  

Minimum generation error training with direct log spectral distortion on LSPs for HMM-based speech synthesis.  

Statistical parametric speech synthesis with joint estimation of acoustic and excitation model parameters.  

Integration of spectral feature extraction and modeling for HMM-based speech synthesis.  

Context adaptive training with factorized decision trees for HMM-based statistical parametric speech synthesis.  

Product of experts for statistical parametric speech synthesis.  
*A clustering technique for factor analysis-based eigenvoice models.*  
(in Japanese).

Statistical parametric speech synthesis based on speaker and language factorization.  


Distributed representation.  

Deep learning: Theoretical motivations.  

Towards minimum perceptual error training for DNN-based speech synthesis.  

Multiple feed-forward deep neural networks for statistical parametric speech synthesis.  
<table>
<thead>
<tr>
<th>Reference</th>
<th>Authors</th>
<th>Title</th>
<th>Conference</th>
<th>Pages</th>
</tr>
</thead>
</table>
[31] Z. Wu and S. King.
Improving trajectory modelling for DNN-based speech synthesis by using stacked bottleneck features and minimum
generation error training.

Bidirectional recurrent neural networks.

[33] Y. Fan, Y. Qian, and F. Soong.
TTS synthesis with bidirectional LSTM based recurrent neural networks.

[34] H. Zen and H. Sak.
Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech
synthesis.

Long short-term memory.

A clockwork RNN.

An investigation of recurrent neural network architectures for statistical parametric speech synthesis.


