Text-to-speech synthesis

Reminder

Our academic program coordinator/assistant program officer Nishi will be sending around a course evaluation. Please fill this out! I read every one and try to be as responsive as possible.

Terminological note

This task is traditionally called *text-to-speech synthesis* (TTS) or simply *speech synthesis*.

Much like *speech-to-text*, the term *text-to-speech* by itself is commonly used among non-specialists but is a shibboleth.

Downstream applications

- Voice user interfaces
- Audiobooks for the visually impaired
- Public service announcement systems

General issues

- Intelligibility vs. naturalness
- High-quality vs. low-latency
- Domain-specific vs. general-purpose

Outline

- Evaluation
- A brief history
- The frontend and backend
- Parametric approaches
- Concatenative approaches
- Neural network-based approaches
- Software and data

Evaluation

Intelligibility vs. naturalness

Intelligibility refers to the ease with which synthesized speech is understood by a speaker.

Naturalness refers to the degree to which synthesized speech resembles that of human speech.

These two features trade off because naturalness involves modeling of coarticulation and reduction, which arguably reduce intelligibility.

Evaluation

Mean opinion score (MOS): 5-point Likert scale measures of intelligibility and naturalness, respectively, averaged across subjects.

The Blizzard Challenge shared tasks run annual human quality evaluations including experts, paid undergrads, and volunteers.

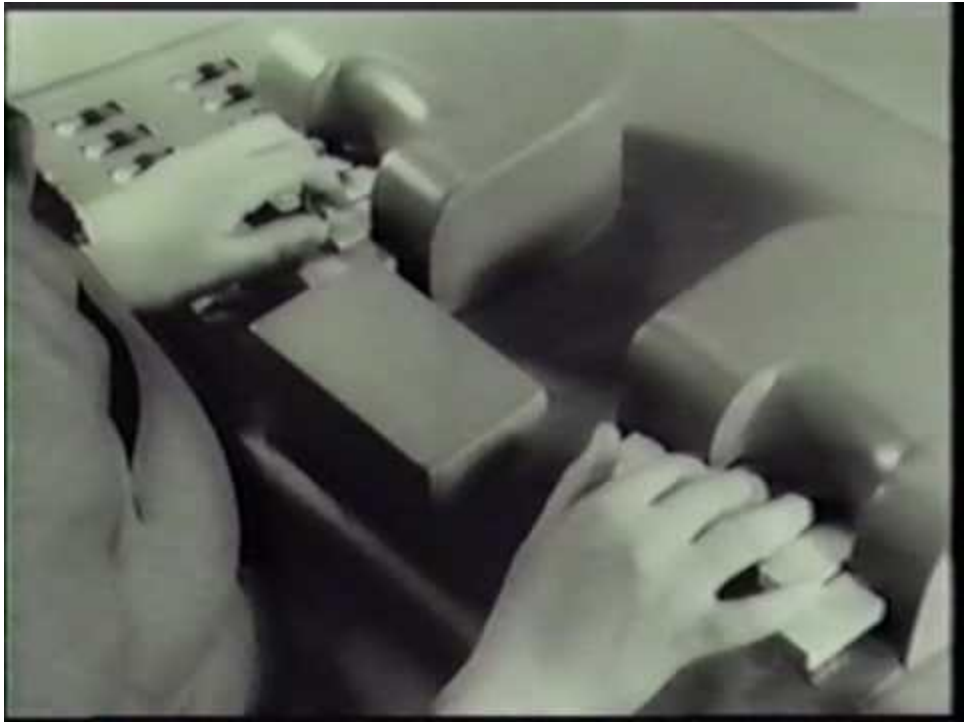
However, coverage of non-standard words, homographs, pronunciations, etc. can be used as objective measurements of the frontend quality.

A brief history of TTS

History of TTS

1791: Wolfgang von Kempelen's *speaking machine*
1937: Homer Dudley's *vocoder* and *voder*
1960s: Linear predictive coding
1961: First computer "singing": "Daisy Bell (Bicycle Built For Two)", featured in *2001: A Space Odyssey* (1968)
1970s: Work begins on MITalk, later commercialized as DECtalk (1980s)
1978: Texas Instruments Speak & Spell, featured in Kraftwerk's "Nummern" (1981)
1980s: Parametric synthesis
1996: Unit selection concatenative synthesis
2000s: Hybrid concatenative/parametric synthesis
2016: WaveNet

(Image credit: https://en.wikipedia.org/wiki/Wolfgang_von_Kempelen%27s_speaking_machine)



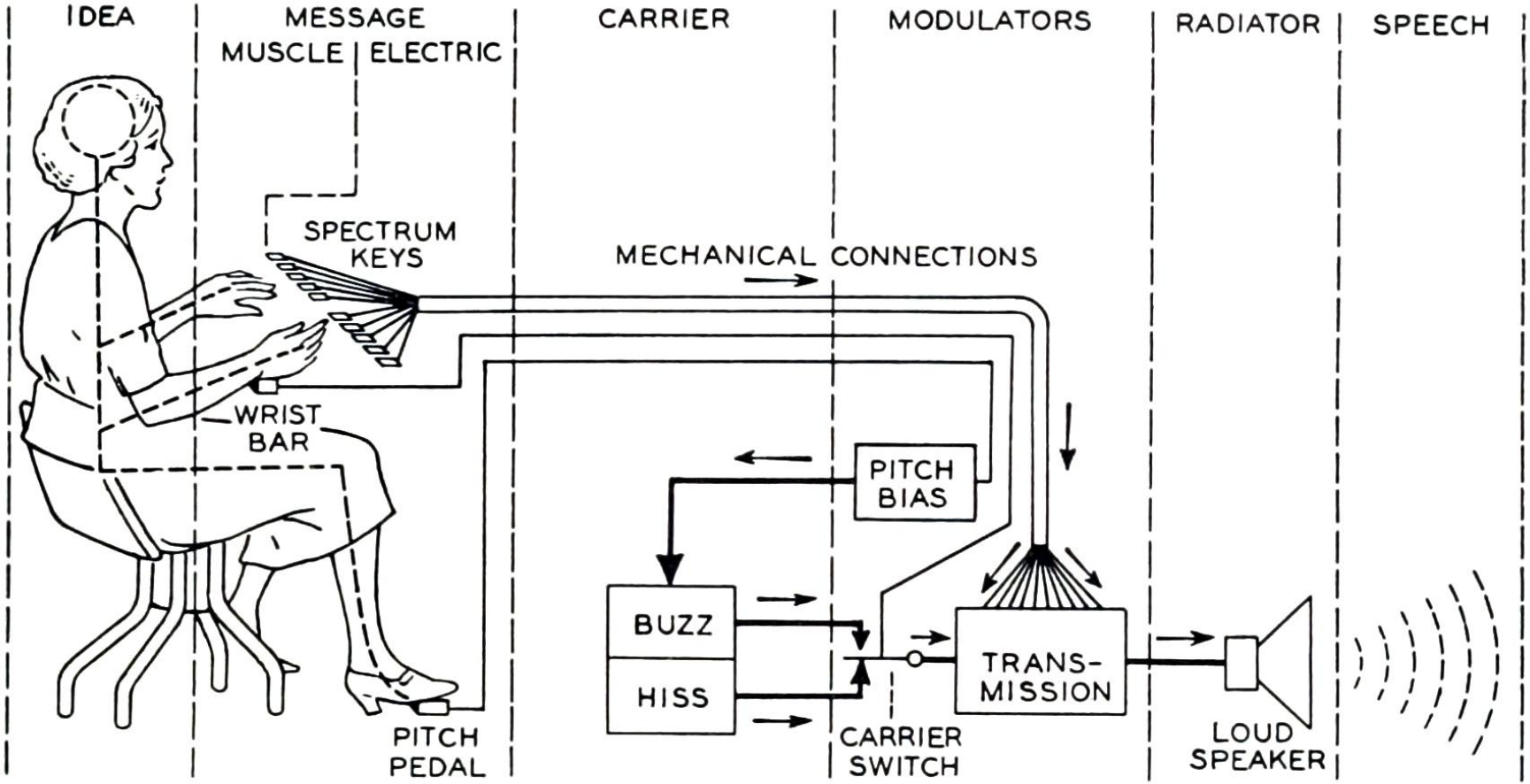


Fig. 8—Schematic circuit of the voder.

(Image credit: https://en.wikipedia.org/wiki/Voder)


Give me
your answ

(Image credit: https://en.wikipedia.org/wiki/Speak_%26_Spell_(toy))

Speak & Spell emulator

WaveNet samples

The frontend and backend

The backend

Most narrowly, we can define speech synthesis as a problem of mapping from phone or phoneme sequences (with or without intonational information) to waveforms. We refer to the system that performs this mapping as the *backend*.

The backend must handle coarticulation, segment timing/duration, and (possibly) pitch/intonation.

The frontend

The frontend is responsible for mapping from human-readable text to phone sequences (and possibly intonation). This may include subtasks such as

- *sentence boundary detection*
- *text normalization*,
- *homograph disambiguation*, and
- *grapheme-to-phoneme conversion*.

Systems used to generate sentences from templates (etc.) are usually not considered part of the frontend.

"End-to-end" synthesis

Virtually all of the neural network research on speech synthesis has focused on the backend (though see, e.g., Zhang et al. 2019).

Some newer neural network systems (e.g. Tacotron) are described as "end-to-end" insofar as they eliminate certain frontend modules (e.g., g2p). However, these usually depend on other frontend processing (e.g., sentence boundary detection, text normalization, etc.) so the "end-to-end" term is often somewhat undeserved.

Parametric synthesis

Articulatory synthesis

The earliest digital synthesizers, following von Kempelen's lead, tried to construct digital models of the human speech organs, performing synthesis by modifying the shape of the simulated "tongue", "lips", etc.

Parametric synthesis

In contrast, *parametric synthesis*, one of the first reasonably effective methods, mimics the physics of the human vocal tract—essentially the physics of a tube of fixed length, closed at one end—without directly modeling the underlying articulators.

Formant-based synthesis

Dennis Klatt's MITalk introduced a simple set of parameters which control a set of *excitations* filtered by *formants*. These parameters include:

- the glottal noise source: impulses of white noise
- formant frequencies (and for nasals, anti-frequencies)
- FO and voicing amplitude

Some limitations include:

- the need to estimate or otherwise manually tune the parameters
- poor handling of coarticulation

Klatt formant synthesizer

emulator

HMM-based synthesis

HMM synthesizers are capable of learning the parameters of parametric synthesis models from data using the familiar expectation maximization algorithm. This allows one to use high dimensional, non-human-interpretable speech codings such as MFCCs.

Concatenative synthesis

Concatenative synthesis

Concatenative synthesizers concatenate pre-joined phone-like units together, doing some simple signal processing at the "joins" (boundaries between units) and imposing an intonational curve.

Traditionally a database of *diphones* are used. Thus the word *cat* spoken in isolation might be represented as

pau/k k/ae ae/t t/pau

Unit-selection synthesis

Unit selection is a variant of concatenative synthesis in which units larger than diphones are stored. The model then selects units so as to

- use the longest possible stored units
- avoid local signal processing issues (the "join cost")

Since such systems prefer to use longer units ("playback"), there is an easy way to improve quality; simply ask the voice talent to record stimuli that

- are frequently synthesized
- contain frequent n-phones, or
- currently have poor MOS scores.

Parametric-concatenative hybridization

In a unit selection system, there may be some contexts in which no stored unit can provide a "good join" with the adjacent units. Rather than using a unit which joins poorly, one may prefer to fall back on a parametric system to generate an easily-joined unit.

A hybrid system of this form can be build from simple regression models (to estimate join costs) and classification models (to predict whether to use a stored unit or a parametric unit).

Neural network speech synthesis

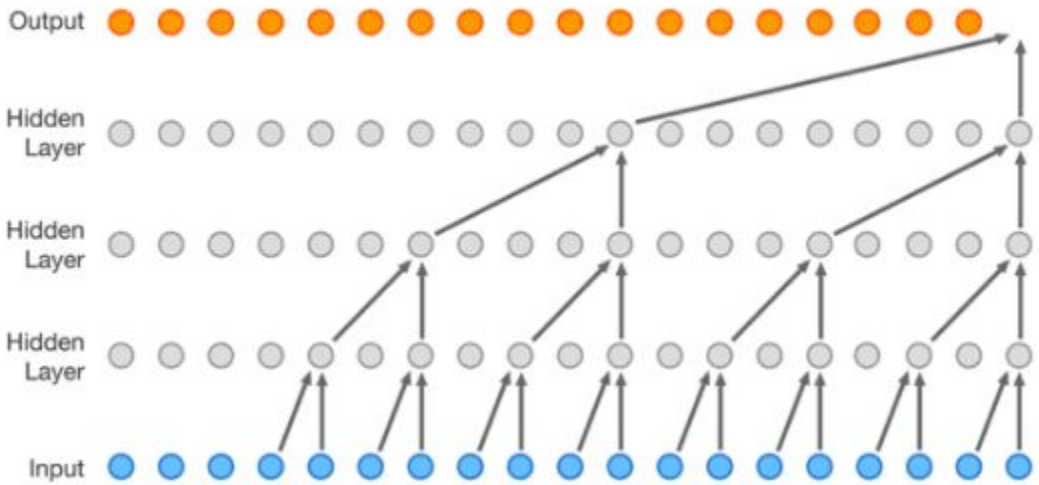
WaveNet

WaveNet is a neural network-based synthesis model using the PixelCNN architecture. It directly generates audio streams from phoneme sequences.

Two new ideas: *dilated convolution* and *companding*.

Dilated convolution

To work at the audio sample level, the model needs to be able look quite a bit far back in the signal generated thus far.



Companding

Recall that "CD-quality" audio uses 16-bit samples, so predicting a sample is a massive 2^{16}-way (= 65,536) multinomial classification problem.

Rather than solving this, WaveNet solves a simpler 256-way multinomial classification problem and uses a non-linear sigmoid mapping back onto the full 16-bit sample space. Your ear probably doesn't notice...

Software & Data

Free (*gratis*, at least) software

- Festival is a venerable open-source general-purpose speech synthesis library.
- Mozilla Voice TTS is an open-source NN-based TTS engine implemented in TensorFlow, based on Google's Tacotron.
- Google Cloud Text-to-Speech provides a free tier with high-quality synthesizers for several dozen languages: one simply crafts a JSON query with the transcription and it replies with the audio.

Data

- The Blizzard Challenge datasets are commonly used by researchers.
- The Mozilla Common Voice project is crowdsourcing free multilingual data.

Unlike ASR, where systems are often built with hundreds of hours of speech, decent quality can be achieved in modern TTS systems with as few as a few hours.

Outstanding problems

- Developing high-quality understudies for rapid development
- Incorporating linguistically-informed models of intonation
- Reducing frontend errors
- Controlling latency/quality tradeoffs
- Internationalization

Project ideas

- Implement a fragment of a TTS frontend in a language other than English.
- Implement a WaveNet synthesizer using your own voice.