

TSIA-206

# Text To Speech (TTS)



Geoffroy Peeters

contact: [geoffroy.peeters@telecom-paris.fr](mailto:geoffroy.peeters@telecom-paris.fr)

Télécom-Paris, IP-Paris, France

*some slides are from Chloé Clavel*

# Speech synthesis or Text To Speech (TTS)

- **Objective :**
  - be able to read any written text

## Automatic speech recognition (ASR)

 → "OK Google, directions home"

## Text-to-speech synthesis (TTS)

"Take the first left" → 

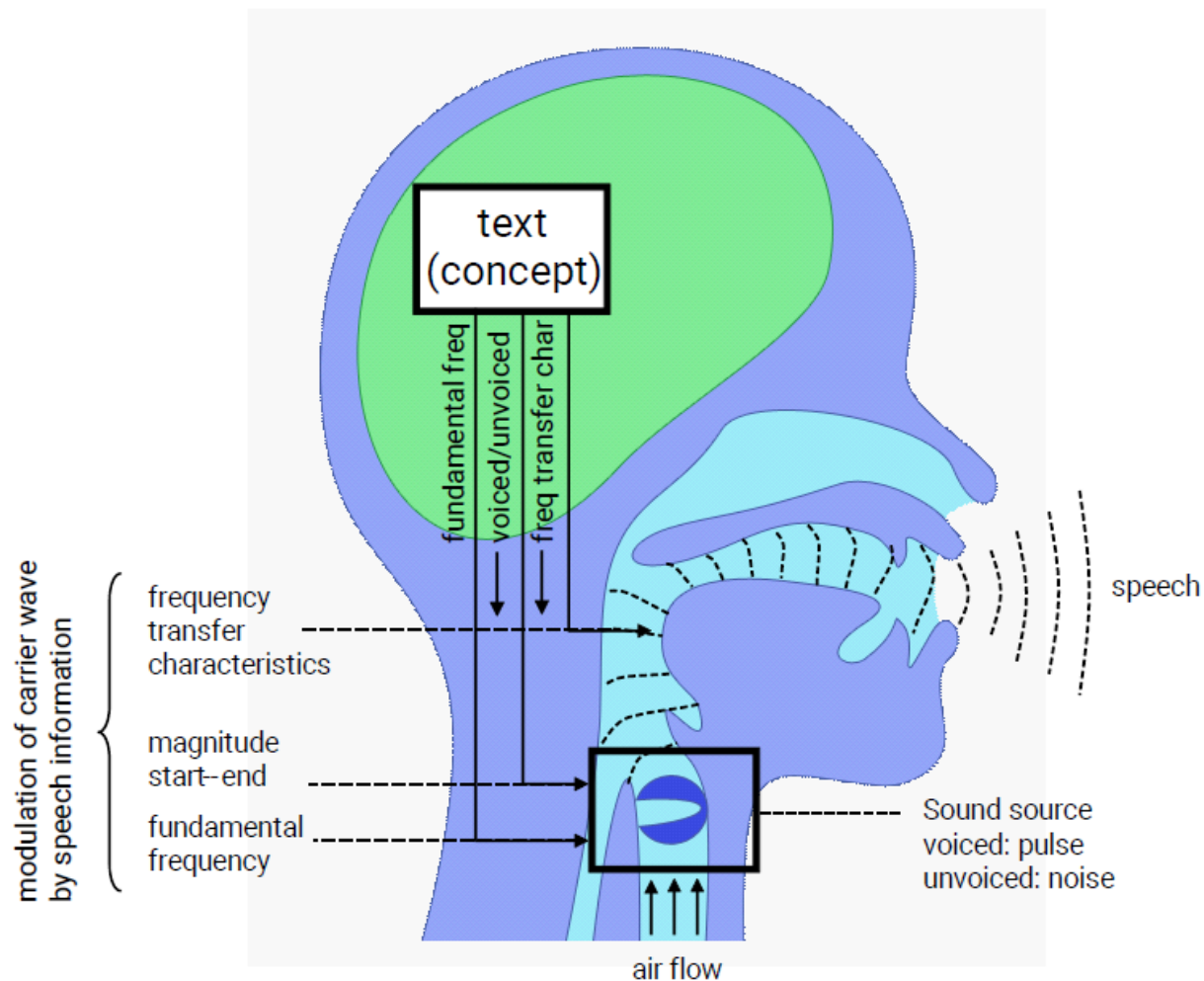
# Text To Speech (TTS)

## Corpora

- Corpora for **ASR** ?
  - **large** corpora with thousands of hours of speech from **many** speakers
  - $\Rightarrow$  ASR systems need to be speaker-**independent** (needs to generalise well to an unseen speaker)
- Corpora for **TTS**:
  - much **less** data but all for **one** speaker
  - $\Rightarrow$  TTS systems are generally speaker-**dependent** : trained to have a consistent voice
  - Example: LJ Speech Dataset [LINK](#)
    - 24 hours of one speaker reading audio books (Ito and Johnson, 2017)

# Text To Speech (TTS)

## Analogy with the mode of speech production



# Text To Speech (TTS)

## Production of the voice signal: the ancestors

- **Von Kempelen (1791)**

- Manual voice synthesiser
- Reproduction of the human vocal tract
- Production of vocal sounds
  - Ex : "nostrils" (narines) for making the "n" and "m" sounds
  - levers (leviers) and tubes dedicated to "s" and "sh" sounds

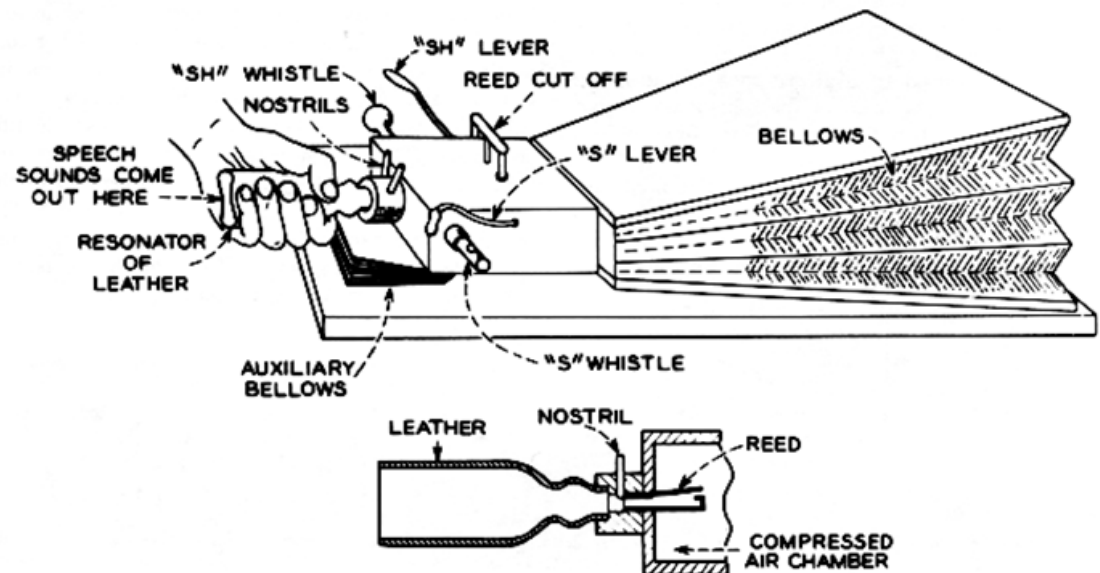
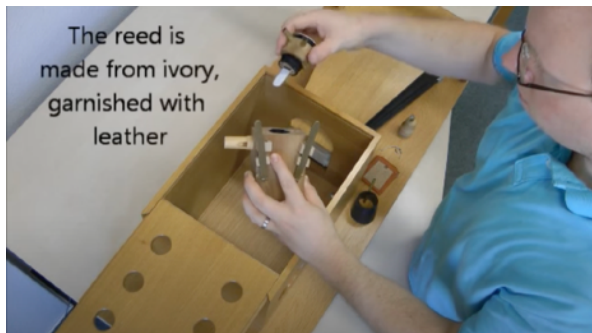


FIG. 10. Wheatstone's reconstruction of von Kempelen's speaking machine.<sup>1</sup>

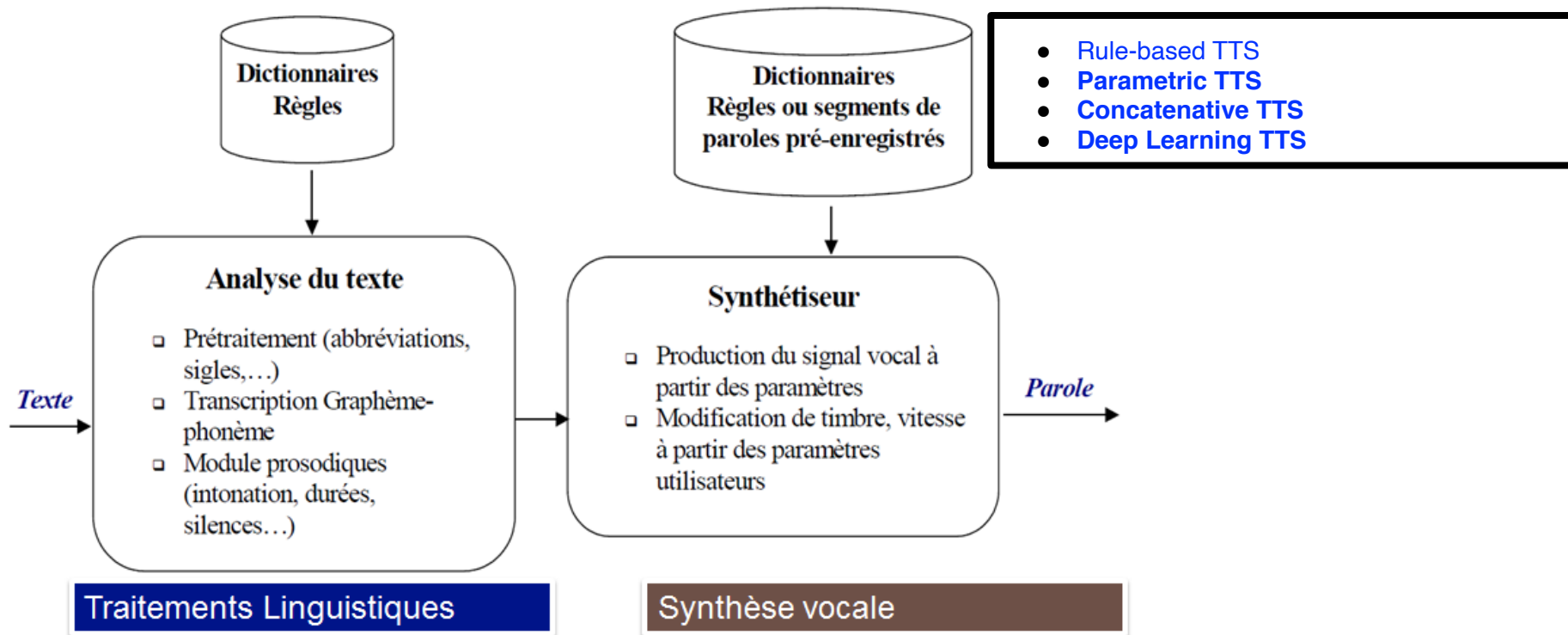
The Journal of the Acoustical Society of America

[https://www.youtube.com/watch?v=k\\_YUB\\_S6Gpo](https://www.youtube.com/watch?v=k_YUB_S6Gpo)

<https://www.youtube.com/playlist?list=PLYw4zi2N1f7f42TOd4U4UXTqQGoBjjS5Z>

# Text To Speech (TTS)

## Classical architecture



# Text To Speech (TTS)

## Different approaches

- 1. Rule-Based Approaches [Klatt, 1980]
- 2. **Concatenative synthesis** [Hunt & Black, 1996].
- 3. **Parametric synthesis**:
  - generative synthesis from models (HMM) [Zen et al., 2009].
  - Contents and characteristics of the speech signal can be controlled via the inputs to the model
- 4. **Deep learning approaches**
  - WaveNet [Van Den Oord et al., 2016].
- **Comparing 2, 3, 4**
  - <https://web.archive.org/web/20170106185014/https://deepmind.com/blog/wavenet-generative-model-raw-audio/>

# Text To Speech (TTS)

## Concatenative synthesis



# Text To Speech (TTS)

## Concatenative synthesis [Hunt & Black, 1996]

- **Principle :**

- Assembling speech segments (stored in a database) corresponding to the phoneme sequence
- Purely acoustic smoothing of discontinuities that may appear at the points of concatenation
- Note : Requires only limited knowledge of the speech signal
- Unit selection and concatenation
  - Concatenate best pre-recorded speech units
  - **Target cost + Concatenation cost**
- Criteria :
  - Intelligibility
  - Naturalness

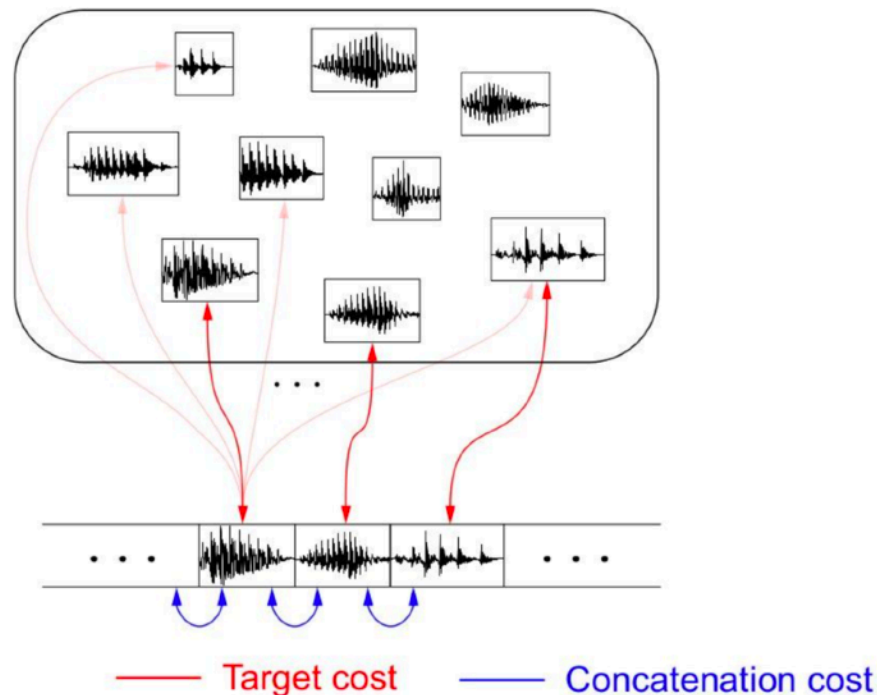


Schéma tiré de [1]

# Text To Speech (TTS)

## Concatenative synthesis

- **Database of speech segments :**
  - From mono-speaker speech recordings with a wide linguistic diversity
  - Split the acoustic signal in a relevant acoustic unit (the diphone)

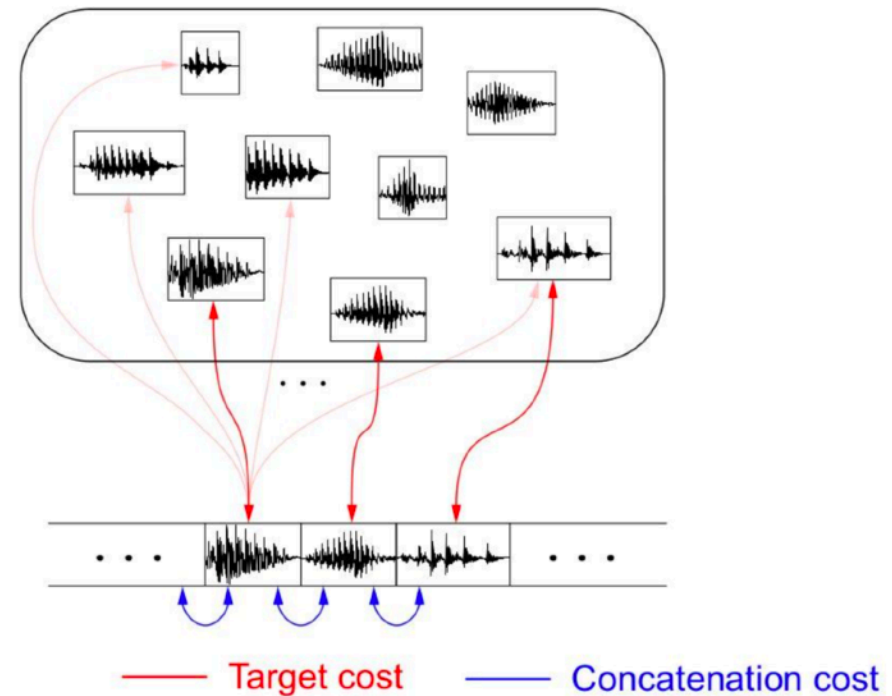
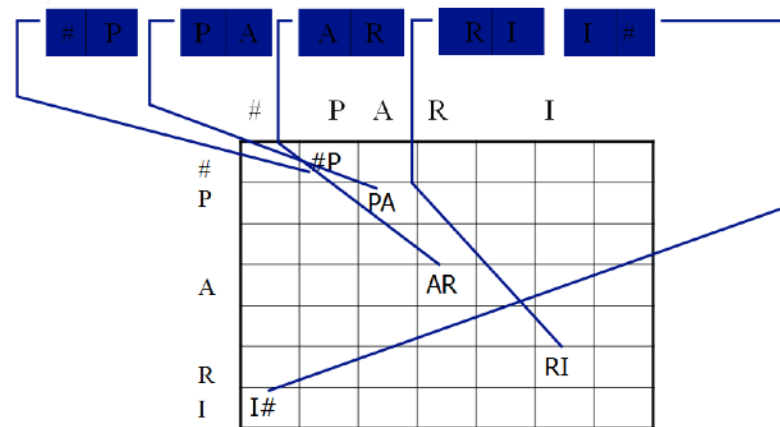
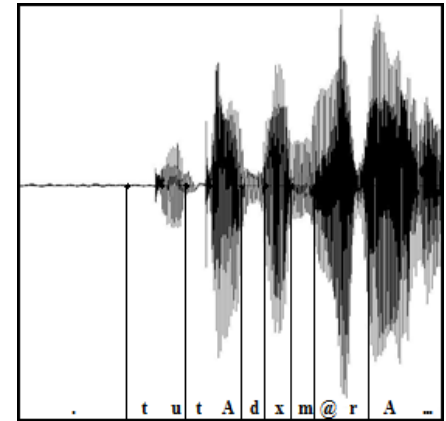


Schéma tiré de [1]

# Text To Speech (TTS)

## Concatenative synthesis

- **Diphones:** (blue) acoustic unit that begins in the middle of the stable range of one phoneme and ends in the middle of the stable range of the next phoneme.
- Allows for the inclusion of coarticulation phenomena.
- Splitting signal into diphones requires to have an alignment between the signal and the phonemes



# Text To Speech (TTS)

## Concatenative synthesis

- Selection of synthesis units (e.g. diphones)
- Building the acoustic segment database (from a speech signal database)
- Assembling already co-articulated acoustic segments corresponding to the phoneme sequence
- Purely acoustic smoothing of discontinuities that may appear at the points of concatenation

Note : Requires only limited knowledge of the speech signal.

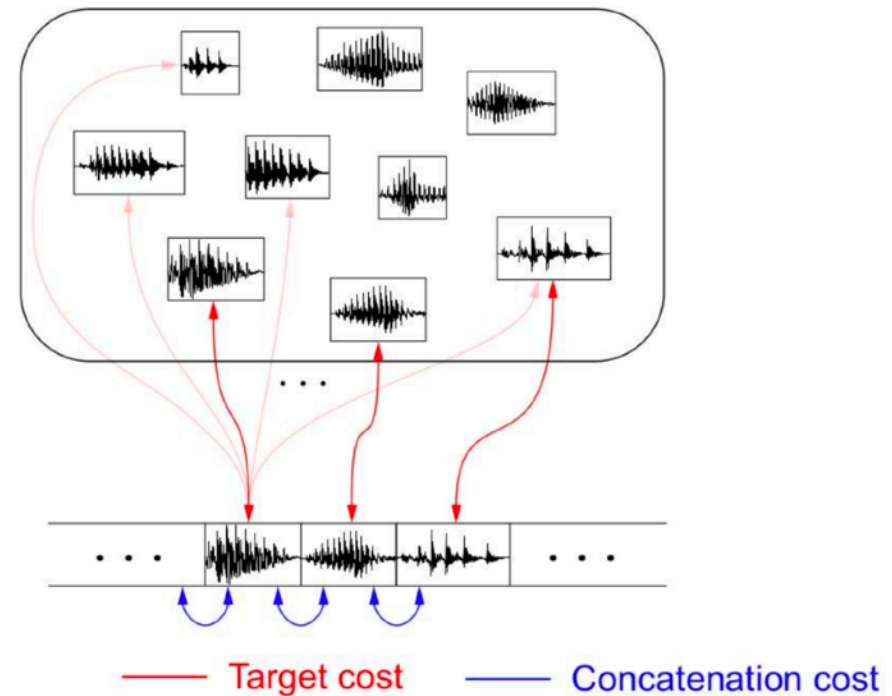


Schéma tiré de [1]

# Text To Speech (TTS)

## Concatenative synthesis

- Considering the input phoneme sequence **select the synthesis units** (speech segment) that will minimise future concatenation problems.
- **Static selection**
  - Pair of phonemes of the input phonetic chain -> selection of the corresponding diphone signal
  - Only one choice per unit
- **Dynamic** selection among several instances of the same diphone
  - Dataset: instances of the same diphone :
    - with different prosodies
    - positioned in different phonetic contexts.
  - Choice made :
    - at the time of the synthesis
    - according to a global selection cost
  - Selection cost depending on:
    - Representation cost:
      - phonetic context as close as possible to the phonetic string to be synthesized
      - prosody as close as possible to the prosody to be produced
    - Concatenation cost:
      - starts and ends with the fewest spectral discontinuities
  - Method:
    - Use of dynamic programming (Viterbi)

# Text To Speech (TTS)

## Parametric synthesis

# Parametric synthesis: generative synthesis from models

- ## Model-based, generative synthesis

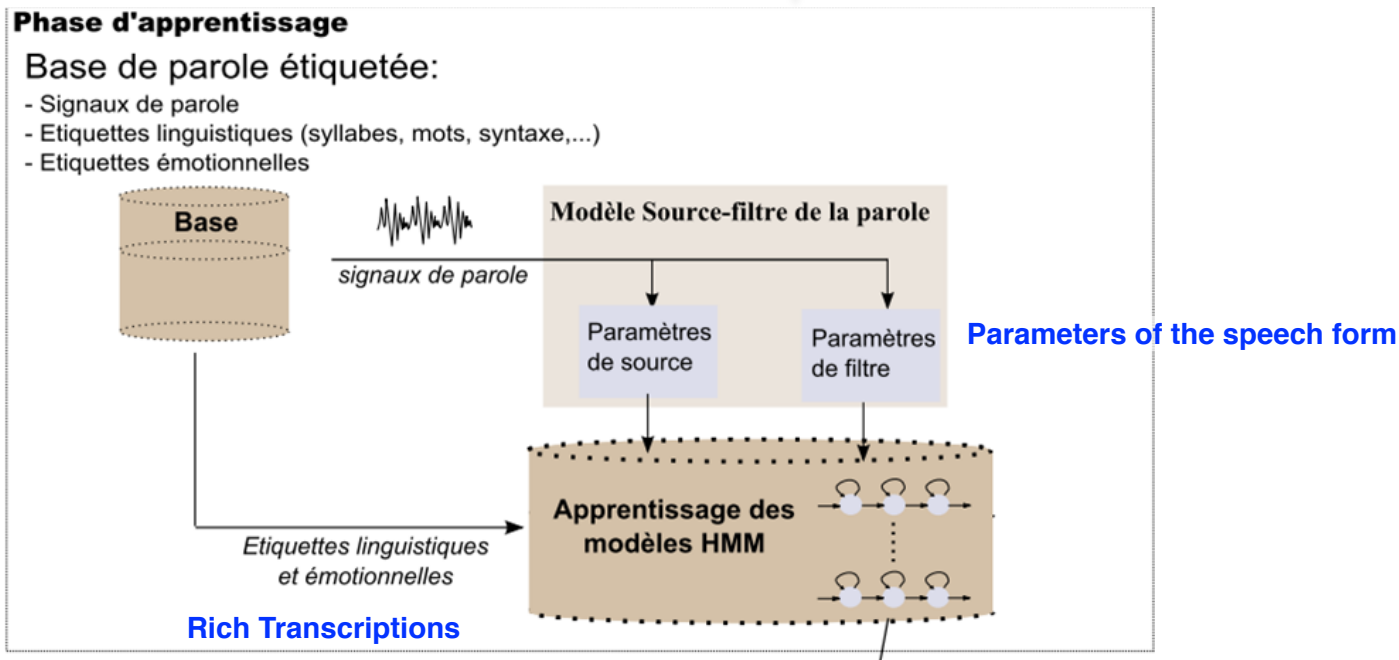
- Instead of text : use **phonetic and prosodic transcription**
- Instead of speech waveform : use **parameters of speech waveform** (e.g. source and filter parameters) and then use a **vocoder**
- Model built from a training corpus :
  - Learn mapping between transcription <-> parameters of speech waveform

# Text To Speech (TTS)

## Parametric synthesis: generative synthesis from models

- HMM synthesis : training phase**

Models learned a mapping between transcription  $\leftrightarrow$  parameters of speech waveform



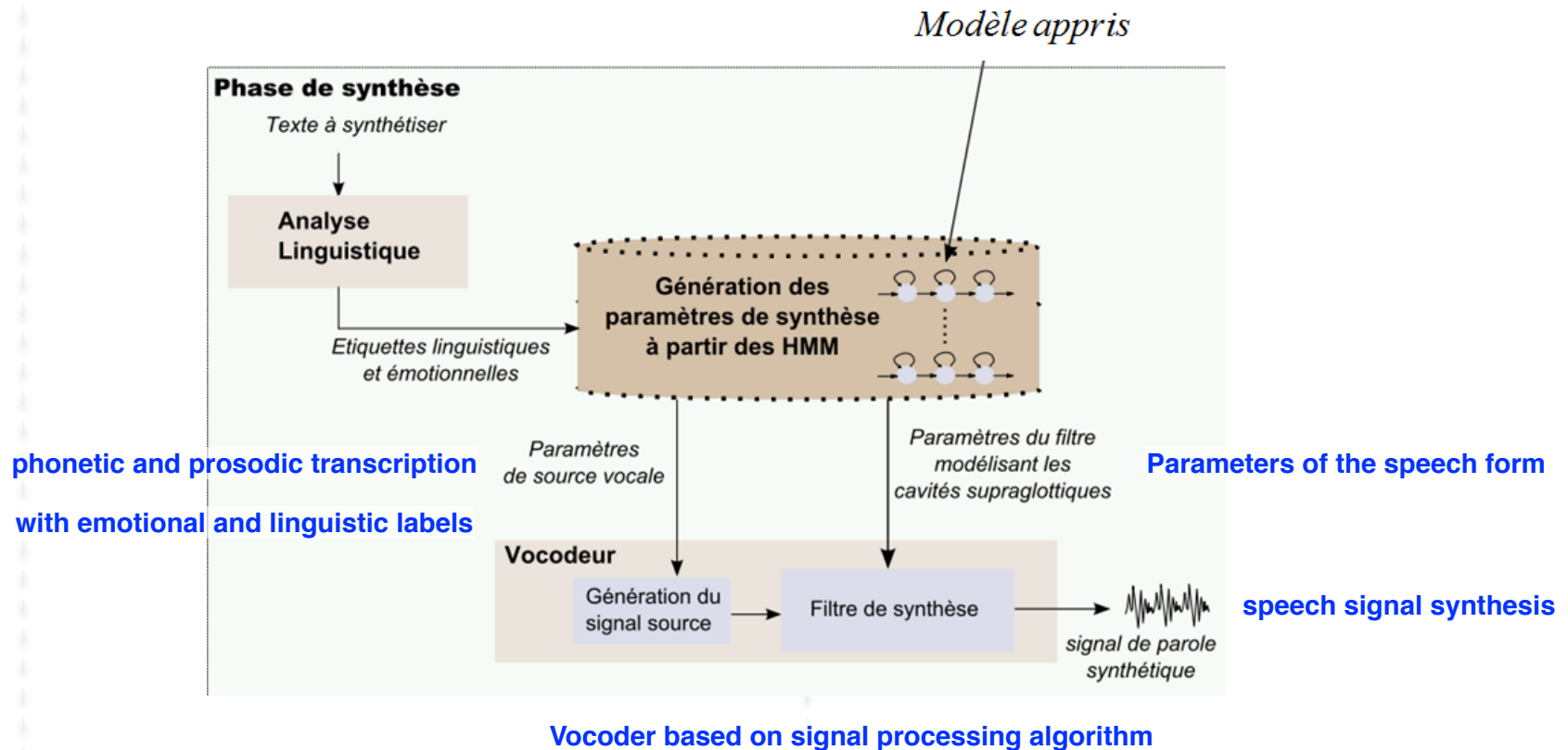


# Text To Speech (TTS)

## Parametric synthesis: generative synthesis from models

- Ex : HMM synthesis

Models learned a mapping between transcription  $\leftrightarrow$  parameters of speech waveform

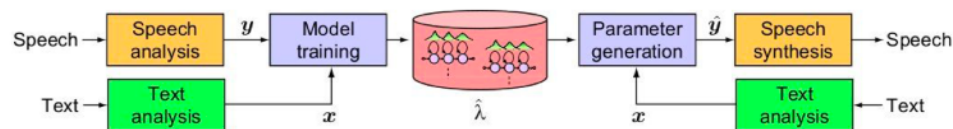


# Text To Speech (TTS)

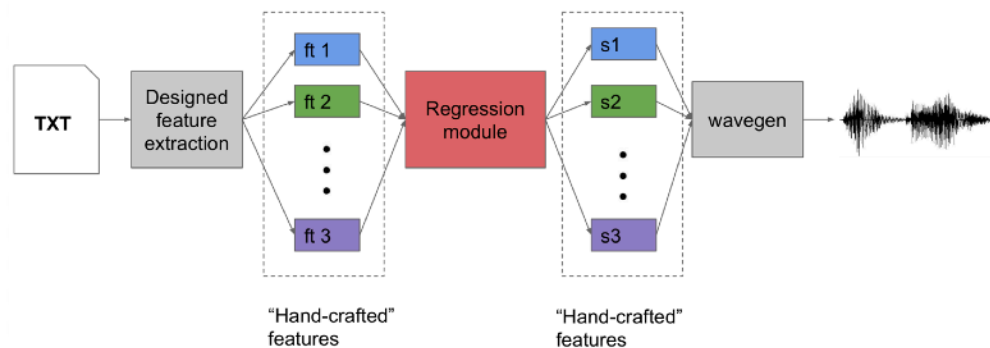
## Parametric synthesis: statistical Speech Synthesis

### Statistical Speech Synthesis

- 1) Represent speech waveform using parameters
  - from text to phoneme (pronunciation)
  - from phoneme to phoneme (with linguistic features)
  - speech features : phoneme duration, fundamental frequency, cepstra
- 2) Use statistic generative model
  - Regression of the parameters using HMM
- 3) Reconstruct waveform from generated parameters
  - vocoder (phase reconstruction)

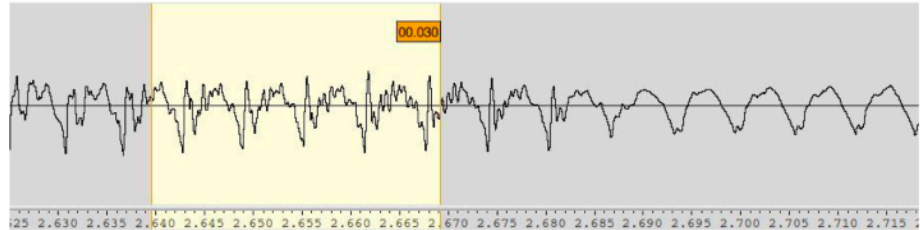


Text to Speech: Textual features  $\rightarrow$  Spectrum of speech (many coefficients)

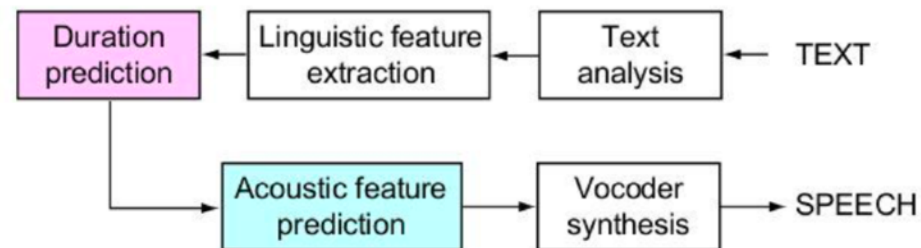


### Statistical Speech Synthesis

- 1) Represent speech waveform using parameters
  - from text to phoneme (pronunciation)
  - from phoneme to phoneme (with linguistic features)
  - speech features : phoneme duration, fundamental frequency, cepstra
- 2) Use statistic generative model
  - Regression of the parameters using HMM
- 3) Reconstruct waveform from generated parameters
  - vocoder (phase reconstruction)



- Rate: ~ 5 ms. (200Hz)
- Spectral features (envelope)
- Excitation features (fundamental frequency, pitch)
- Representation that allows reconstruction: vocoders (Straight, Ahocoder, ...)



Reminder STFT  
START →

# Short-Time Fourier Transform (STFT)

## Signal reconstruction by Overlap-and-Add (OLA, $\text{STFT}^{-1}$ )

- If we don't modify the STFT

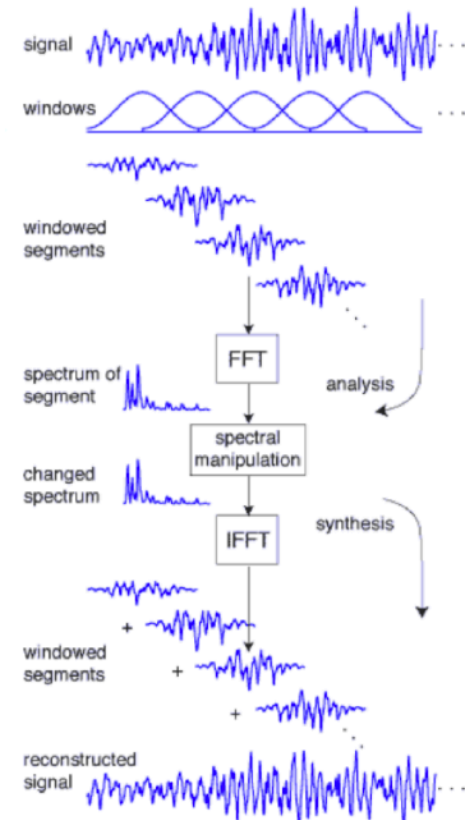
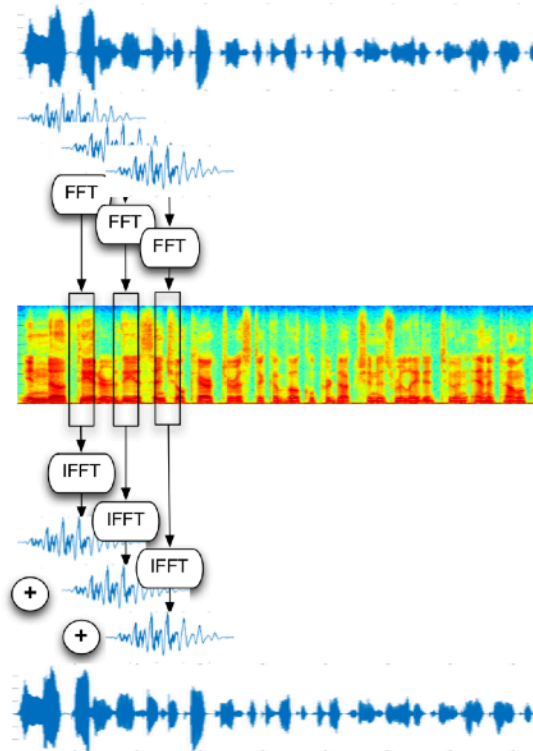
- we can reconstruct the signal  $x(n)$  using the STFT information  $X(k, pR)$  using the **OverLap-and-Add (OLA) method**
- If we denote by  $n = pR$ ,  $p$  the frame number,  $R$  the hop size

$$X(k, pR) = \sum_{m=0}^{N-1} x(m)w(pR - m)e^{-j2\pi \frac{k}{N}m}$$

$$\begin{aligned} y(m, pR) &= \frac{1}{N} \sum_{k=0}^{N-1} X(k, pR)e^{+j2\pi \frac{k}{N}m} \\ &= x(m)w(pR - m) \end{aligned}$$

$$\begin{aligned} \hat{y}(m) &= \sum_p y(m, pR) \\ &= \sum_p x(m)w(pR - m) \\ &= x(m) \sum_p w(pR - m) \end{aligned}$$

$$x(m) = \frac{\sum_p y(m, pR)}{\sum_r w(pR - m)}$$



# Short-Time Fourier Transform (STFT)

## Signal reconstruction using Griffin & Lim algorithm (I)

- If we modify the STFT
  - we can use the **Griffin & Lim algorithm** to reconstruct the audio signal
  - **STFT redundancy**
    - STFT with 50% overlap
    - → STFT has twice the number of data than the signal !
      - informations contained in the STFT are not independent of each other
  - **Modification of the module of the STFT(filtering, denoising, ...)**
    - it is not necessarily possible to find a time signal corresponding to an arbitrary STFT
  - **Griffin & Lim algorithm**
    - from any STFT (not necessarily valid), it is possible to modify it iteratively to obtain a temporal signal which corresponds to it (by changing the phase for example)

Reminder STFT

← STOP

# Text To Text To Speech (TTS)

## Deep learning approach



# Text To Speech (TTS)

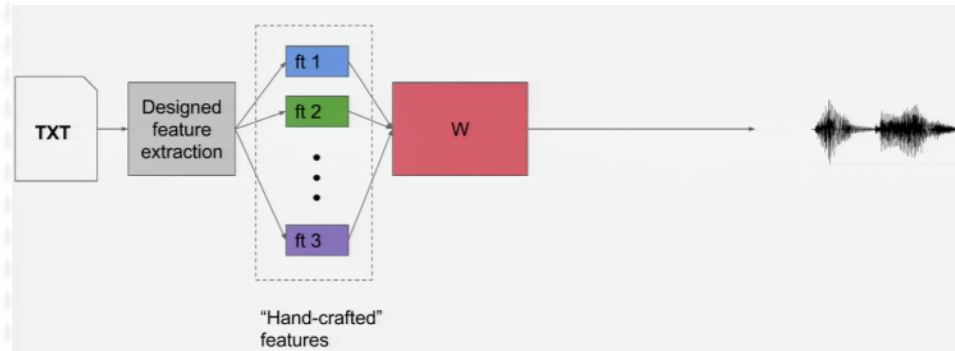
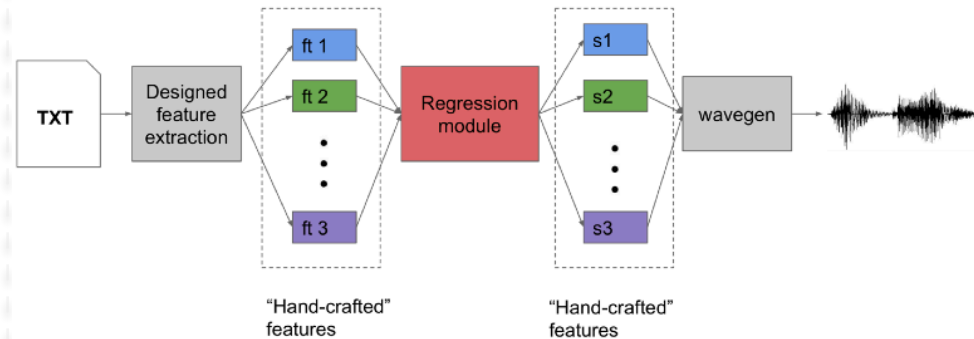
## Deep learning approach

### WaveNet

#### – Summary

- Based on PixelCNN
- Generative model operating directly on audio samples
- Stack of Convolutional Neural Networks (dilated convolution)
- Output :
  - categorical distribution  $\hat{y}$  softmax
- Conditioning :
  - prosody parameters, speaker

Text to Speech: Textual features  $\rightarrow$  Spectrum of speech (many coefficients)



## – Generative model operating directly on audio samples

- raw audio = challenging to model because structure at very different scales
- Based on PixelCNN
- High resolution signal and long-term dependencies
  - dependencies at **different level**

## – Auto-regressive model

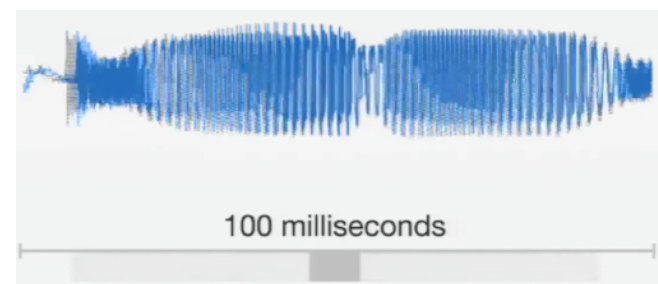
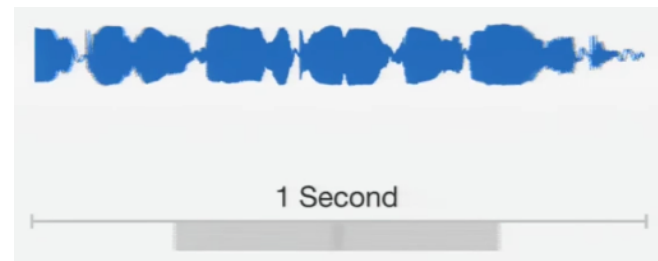
- next sample is (almost) reconstructed from **linear** causal combination of past samples

$$p(x_1, x_2, \dots, x_T) = \prod_{t=1}^T p(x_t | x_1, \dots, x_{t-1})$$

$$x_t = \sum_{p=1}^P \alpha_p x_{t-p} + \epsilon_t$$

## – Neural Auto-regressive model:

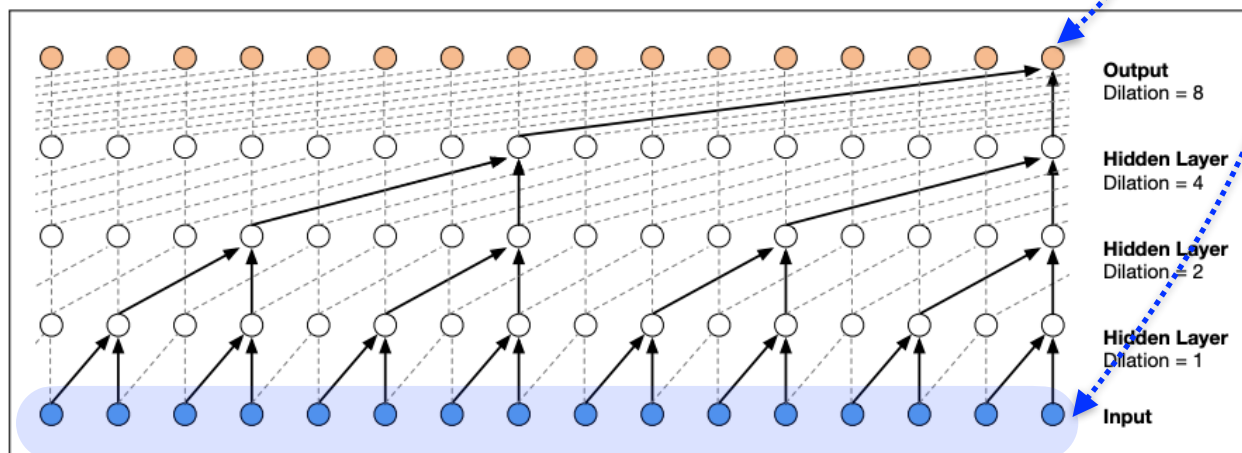
- $x_t = f_{\theta}(x_1, \dots, x_{t-1})$



- The probability distribution is modelled by a stack of **special convolution layers (causal dilated convolutions + a specific non-linearity function)**

$\mathbf{x} = \{x_0, x_1, \dots, x_{N-1}\}$  : raw waveform

$$p(\mathbf{x} \mid \lambda) = p(x_0, x_1, \dots, x_{N-1} \mid \lambda) = \prod_{n=0}^{N-1} p(x_n \mid x_0, \dots, x_{n-1}, \lambda)$$



The output is then passed through a softmax which makes 256-way decision (in order to obtain a sample compressed into 8 bits)

**Figure 26.15** Dilated convolutions, showing one dilation cycle size of 4, i.e., dilation values of 1, 2, 4, 8. Figure from [van den Oord et al. \(2016\)](#).

## – Neural Auto-regressive model:

- $x_t = f_\theta(x_1, \dots, x_{t-1})$
- using RNN  $\Rightarrow$  costly
- using CNN  $\Rightarrow$  cheap (all convolution can be done in parallel)

## – Causal Convolution

- problem : require many layers, or large filters to increase the receptive field.

## – Dilated Convolution (convolution à trou) :

- skipping input values with a certain step
- increase the receptive field by orders of magnitude

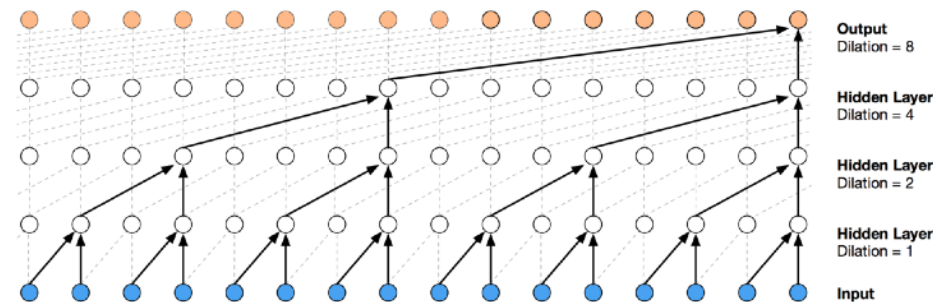
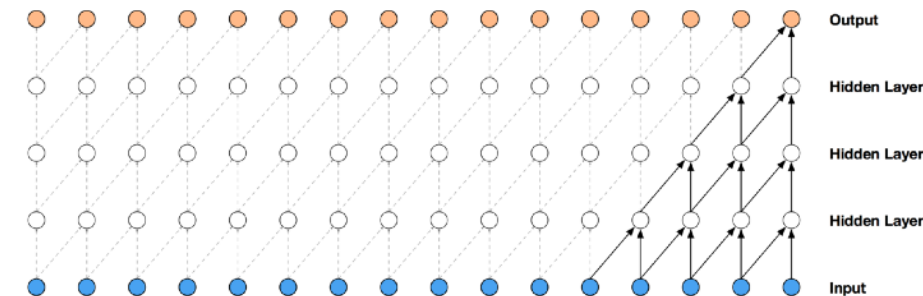
## – Stacked dilated convolutions

- dilation is doubled for every layer up to a limit and then repeated

– 1,2,4,...,512,1,2,4,...,512,1,2,4,..., 512.

## – Generation is slow :

- predictions sequential (2 m to generate 1 s)



## – Input/output signal representation

- using  $\mu$ -law  $\Rightarrow$  8 bits  $\Rightarrow$  256 categories

- $$f(x_t) = \text{sign}(x_t) \frac{\log(1 + \mu |x_t|)}{\log(1 + \mu)}$$
  - with  $-1 < x_t < 1$  and  $\mu = 256$

## – Softmax layer

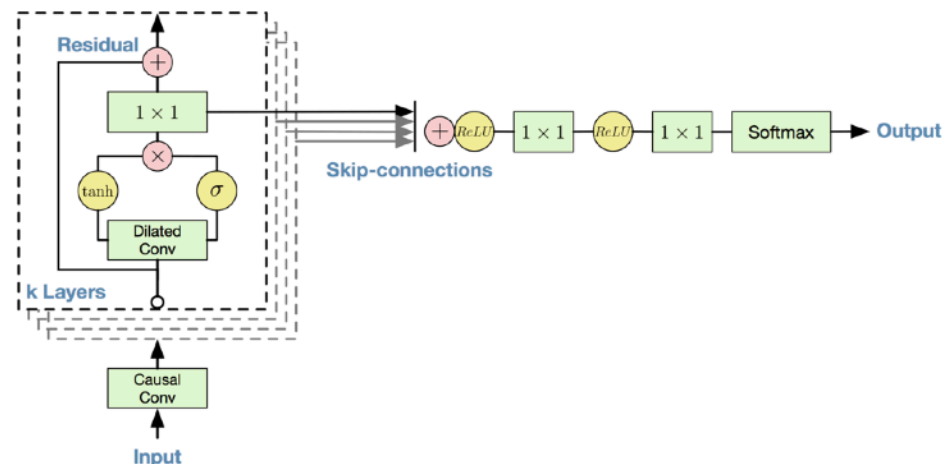
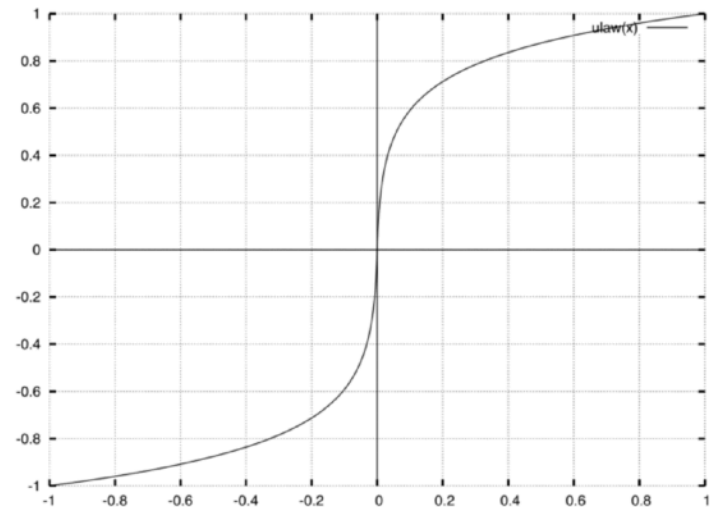
- trained to maximize log-likelihood of the data w.r.t. parameters

## – Gated Activation Units

- $\mathbf{z} = \tanh(W_{f,k} * \mathbf{x}) \odot \sigma(W_{g,k} * \mathbf{x})$
- with
  - $*$  the convolution
  - $\odot$  the element-wise multiplication

## – Residual and Skip connections

- to speed up convergence and enable training of much deeper models



- **Conditional Wavenet** :  $p(\mathbf{x} | \mathbf{h}) = \prod_{t=1}^T p(x_t | x_1, \dots, x_{t-1}, \mathbf{h})$ 
  - conditioning the model on other input variables  $\mathbf{h}$
- **Global conditioning** (speaker ID, music genre, instruments)
  - $\mathbf{h}$ : single latent representation
  - influences the output distribution across all time steps,
    - e.g. a speaker embedding in a TTS model
  - $\mathbf{z} = \tanh(W_{f,k} * \mathbf{x} + V_{f,k}^T \mathbf{h}) \odot \sigma(W_{g,k} * \mathbf{x} + V_{g,k}^T \mathbf{h})$ 
    - $V_{*,k}$  is a learnable linear projection
    - $V_{*,k}^T \mathbf{h}$  is broadcasted over the time dimension
- **Local conditioning** (text, linguistic features + f0 + duration model)
  - $h_t$ : time series, possibly with a lower sampling frequency than the audio signal,
    - e.g. linguistic features in a TTS model
  - First transform this time series using a transposed convolutional network (learned upsampling)
    - (maps it to a new time series  $\mathbf{y} = f(\mathbf{h})$  with the same resolution as the audio signal)
  - $\mathbf{z} = \tanh(W_{f,k} * \mathbf{x} + V_{f,k}^T * \mathbf{y}) \odot \sigma(W_{g,k} * \mathbf{x} + V_{g,k}^T * \mathbf{y})$

## – Perceptual MOS tests

Speech samples	Subjective 5-scale MOS in naturalness	
	North American English	Mandarin Chinese
LSTM-RNN parametric	$3.67 \pm 0.098$	$3.79 \pm 0.084$
HMM-driven concatenative	$3.86 \pm 0.137$	$3.47 \pm 0.108$
<b>WaveNet (L+F)</b>	<b><math>4.21 \pm 0.081</math></b>	<b><math>4.08 \pm 0.085</math></b>
Natural (8-bit $\mu$ -law)	$4.46 \pm 0.067$	$4.25 \pm 0.082$
Natural (16-bit linear PCM)	$4.55 \pm 0.075$	$4.21 \pm 0.071$

Table 1: Subjective 5-scale mean opinion scores of speech samples from LSTM-RNN-based statistical parametric, HMM-driven unit selection concatenative, and proposed WaveNet-based speech synthesizers, 8-bit  $\mu$ -law encoded natural speech, and 16-bit linear pulse-code modulation (PCM) natural speech. WaveNet improved the previous state of the art significantly, reducing the gap between natural speech and best previous model by more than 50%.

## – Audio examples


- <https://deepmind.com/blog/article/wavenet-generative-model-raw-audio>

## – Pytorch code

- <https://github.com/vincentherrmann/pytorch-wavenet>
- <https://github.com/Dankrushen/Wavenet-PyTorch>
- <https://github.com/ButterscotchV/Wavenet-PyTorch>

## Making Music

Since WaveNets can be used to model any audio signal, we thought it would also be fun to try to generate music. Unlike the TTS experiments, we didn't condition the networks on an input sequence telling it what to play (such as a musical score); instead, we simply let it generate whatever it wanted to. When we trained it on a dataset of classical piano music, it produced fascinating samples like the ones below:




▶ 0:00 / 0:10   

▶ 0:00 / 0:10   

▶ 0:00 / 0:10   

▶ 0:00 / 0:10   

▶ 0:00 / 0:10   

▶ 0:00 / 0:10   

<https://deepmind.com/blog/wavenet-generative-model-raw-audio/>



- **Statistical parametric TTS**

- text frontend extracting various linguistic features, a duration model,
- acoustic feature prediction model
- complex signal-processing-based vocoder

- **WaveNet ?**

- slow due (sample-level autoregressive)
- requires conditioning on linguistic features from an existing TTS frontend (not end-to-end)
- only replaces the vocoder and acoustic model.

- **Tacotron:**

- synthesises speech directly from characters
- trained on <text, audio> pairs
- Seq2Seq with attention
  - Predicts Mel-Spectrogram from characters
- Post-processing network
  - Mel to Linear spectrogram
- Griffin & Lim for phase

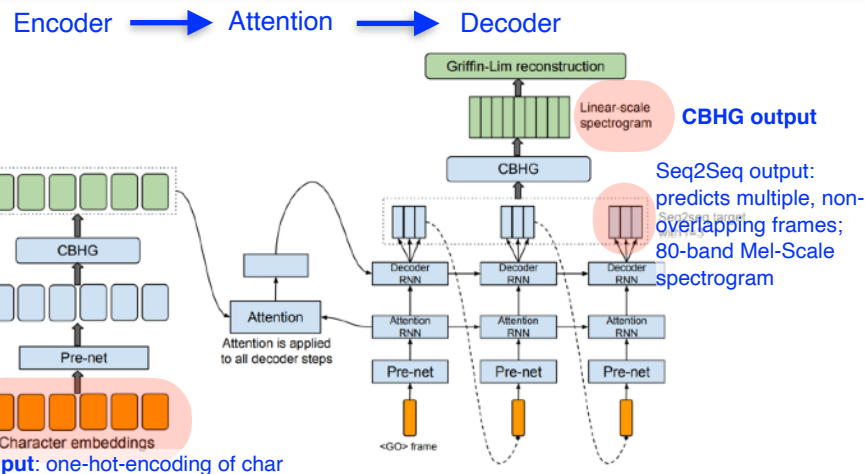


Figure 1: Model architecture. The model takes characters as input and outputs the corresponding raw spectrogram, which is then fed to the Griffin-Lim reconstruction algorithm to synthesize speech.

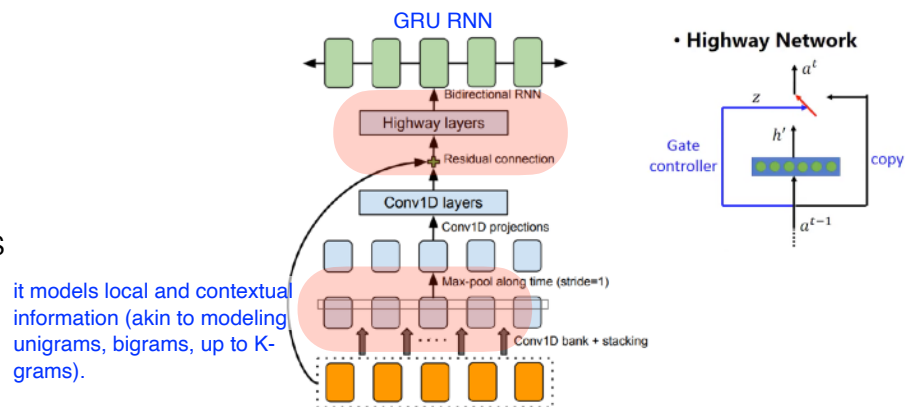


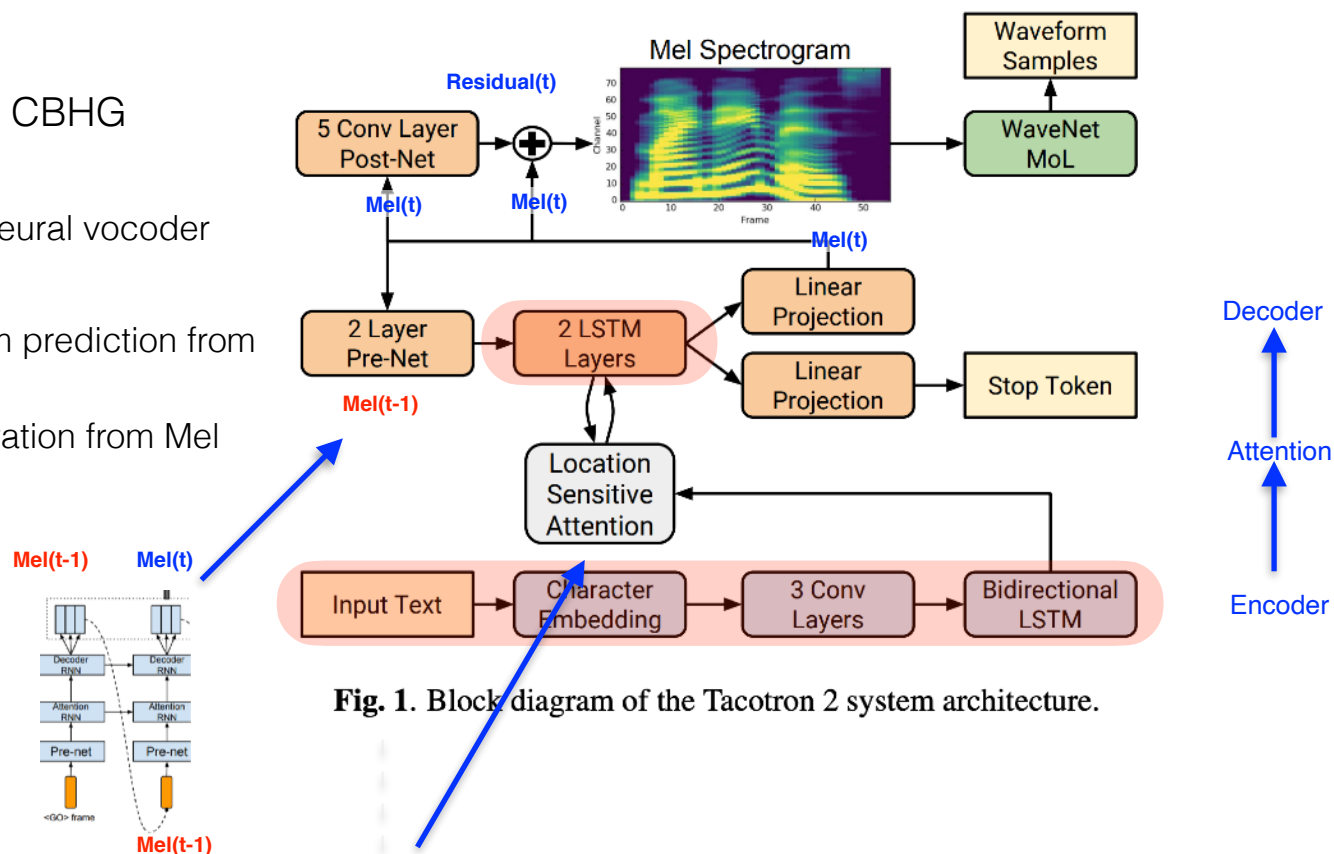
Figure 2: The CBHG (1-D convolution bank + highway network + bidirectional GRU) module adapted from Lee et al. (2016).

- **Audio examples**

- <https://google.github.io/tacotron/publications/tacotron/index.html>

## • Tacotron 2:

- get rid of the complex CBHG
- get rid of Griffin & Lim
  - use WaveNet as a neural vocoder
- Train separately
  - the Mel Spectrogram prediction from char
  - the Waveform generation from Mel Spectrogram



**Fig. 1.** Block diagram of the Tacotron 2 system architecture.

LSA: uses cumulative attention weights from previous decoder time steps as an additional feature. This encourages the model to move forward consistently through the input

- **Audio examples**

- <https://google.github.io/tacotron/publications/tacotron2/index.html>



## • VALL-E

- synthesise high-quality personalised speech with only a 3-second enrolled recording of an unseen speaker
- TTS:
  - train a neural codec language model (called VALL-E) using discrete codes derived from an off-the-shelf neural audio codec model
  - regard TTS as a conditional language modelling task rather than continuous signal regression as in previous work

