Text-to-Speech Synthesis (TTS)

Computer Vision Reading Group (IITK)

January 25, 2022

A Survey on Neural Speech Synthesis

Xu Tan, Tao Qin, Frank Soong, Tie-Yan Liu

https://arxiv.org/abs/2106.15561

FastSpeech 2: Fast and High-Quality End-to-End Text to Speech

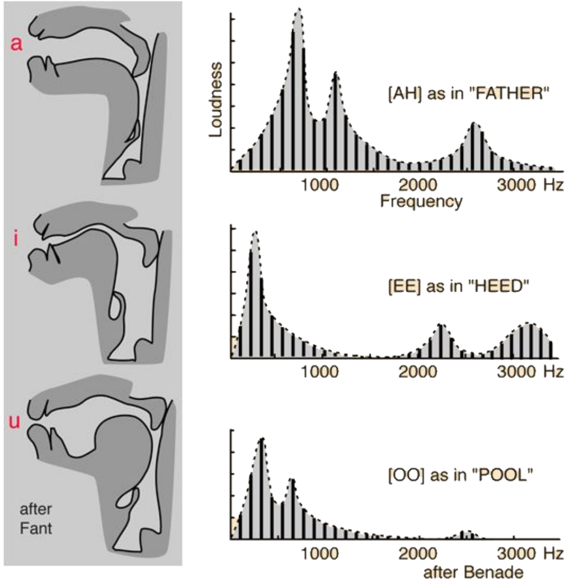
Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, Tie-Yan Liu

(ICLR 2021)

Speech Synthesis

- **Goal:** Synthesize *intelligible* and *natural speech* from text

- **History of Speech Synthesis**
  - **Articulatory synthesis** - by simulating the behaviour of human articulator such as lips, tongue, glottis, and vocal tract
    - Difficult to model, poor speech quality
  - **Formant synthesis** - *Additive synthesis* of speech based on rules that mimic the spectral properties (formant structure, noise levels) etc of human speech
    - Formants are the broad peaks in the frequency spectrum of the sound - F0, F1, F2
    - Intelligible but robotic speech
  - **Concatenative synthesis** - concatenates pieces of speech from a database of speech samples to generate the output waveform
    - Natural sounding but have glitches and artifacts, also need large database
  - **Statistical Parametric Synthesis**
  - **Neural Speech Synthesis**



Formant Structure of vowel sounds
http://hyperphysics.phy-astr.gsu.edu/hbase/Music/vowel.html

Statistical Parametric Speech Synthesis (SPSS)

- Predict acoustic parameters from text using a statistical model and then use the predicted acoustic parameters to produce speech
- Three components:
    - **Text analysis module** extracts the **linguistic features**, such as *phonemes*, *duration* and *POS tags* from the text
    - **Acoustic model** predicts the acoustic features, such as *fundamental frequency*, *spectrum* or *cepstrum*, from the linguistic features
        - *Hidden markov models (HMM)* are used as the acoustic models
        - Trained on pairs of linguistic and acoustic features
    - **Vocoder** synthesizes speech from the acoustic features

Text Analysis for Speech Synthesis

- **Speech-oriented text normalization**
  - Raw text is converted (including numbers, dates etc.) into spoken words (eg. 1984 → `nineteen eighty four`)
- **Word segmentation**
  - Relevant for character-based languages such as Chinese
- **Part-of-speech (POS) Tagging**
  - Tagging POS of each word (eg. noun, verb, preposition)
- **Prosody Prediction**
  - Prosody refers to the elements of speech such as rhythm, stress or intonation
  - Different tagging systems for different languages
  - *ToBI (tones and break indices)* system for English specifies
    - Tags for tones (eg. *pitch accents*, *phrase accents*, *boundary accents*)
    - Tags for break (how strong the break is b/w words)
- **Grapheme-to-Phoneme (G2P) conversion**
  - Converts *graphemes* (spelling) to *phonemes* (pronunciation)

Grapheme to Phoneme Conversion

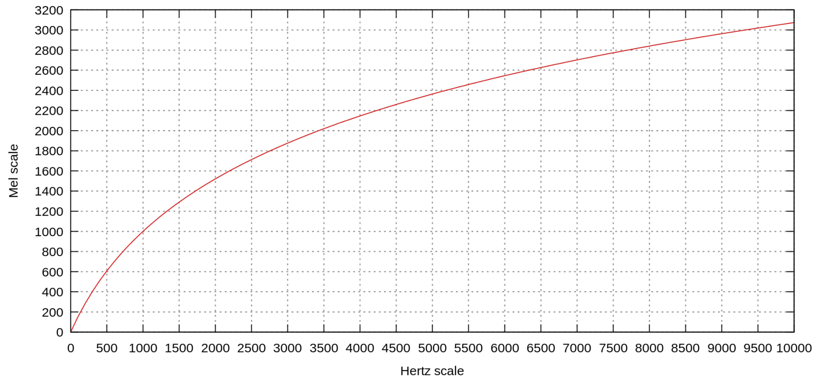
https://github.com/Kyubyong/g2p

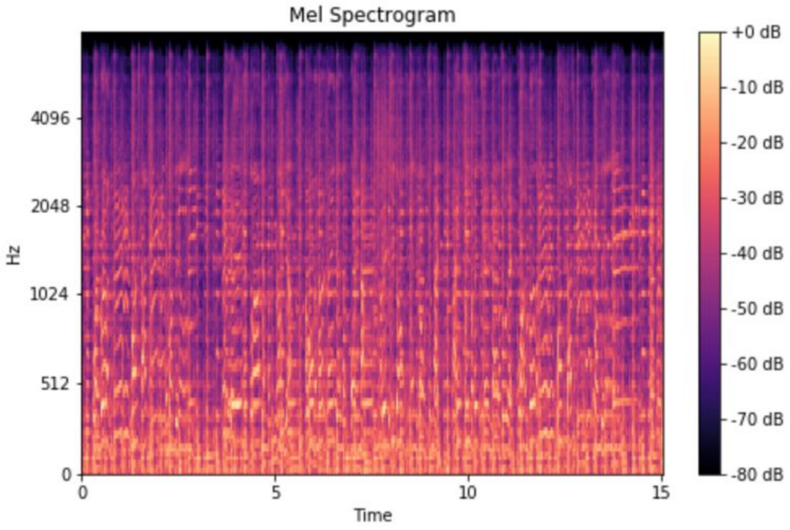
```
from g2p_en import G2p

texts = ["I have $250 in my pocket.", # number -> spell-out
         "popular pets, e.g. cats and dogs", # e.g. -> for example
         "I refuse to collect the refuse around here.", # homograph
         "I'm an activationist."] # newly coined word
g2p = G2p()
for text in texts:
    out = g2p(text)
    print(out)
>>> ['AY1', ' ', 'HH', 'AE1', 'V', ' ', 'T', 'UW1', ' ', 'HH',
'AH1', 'N', 'D', 'R', 'AH0', 'D', ' ', 'F', 'IH1', 'F', 'T', 'IY0',
' ', 'D', 'AA1', 'L', 'ER0', 'Z', ' ', 'IH0', 'N', ' ', 'M', 'AY1',
' ', 'P', 'AA1', 'K', 'AH0', 'T', ' ', '.']
>>> ['P', 'AA1', 'P', 'Y', 'AH0', 'L', 'ER0', ' ', 'P', 'EH1', 'T',
'S', ' ', ',', ' ', 'F', 'AO1', 'R', ' ', 'IH0', 'G', 'Z', 'AE1',
'M', 'P', 'AH0', 'L', ' ', 'K', 'AE1', 'T', 'S', ' ', 'AH0', 'N',
'D', ' ', 'D', 'AA1', 'G', 'Z']
>>> ['AY1', ' ', 'R', 'IH0', 'F', 'Y', 'UW1', 'Z', ' ', 'T', 'UW1',
' ', 'K', 'AH0', 'L', 'EH1', 'K', 'T', ' ', 'DH', 'AH0', ' ', 'R',
'EH1', 'F', 'Y', 'UW2', 'Z', ' ', 'ER0', 'AW1', 'N', 'D', ' ', 'HH',
'IY1', 'R', ' ', '.']
>>> ['AY1', ' ', 'AH0', 'M', ' ', 'AE1', 'N', ' ', 'AE2', 'K', 'T',
'IH0', 'V', 'EY1', 'SH', 'AH0', 'N', 'IH0', 'S', 'T', ' ', '.']
```

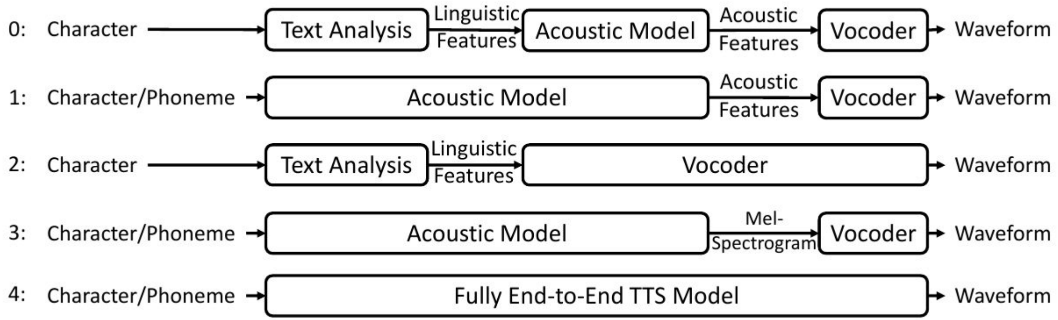
Acoustic Features

- Acoustic features are spectral properties of the waveform
- SPSS systems typically use:
  - *Mel-Frequency Cepstral Coefficients* (MFCC)
  - *Fundamental Frequency* (F0)
  - *Band Aperiodicity* (BAP)
- Recent neural models use *mel-spectrogram*
  - Spectrogram with frequencies in *mel scale*
  - *Mel scale* is a logarithmic scale of frequencies that represents how we perceive pitch
  - Provides better resolution for low frequencies





Neural Speech Synthesis



Stage	Models
0	SPSS [416, 356, 415, 425, 357]
1	ARST [375]
2	WaveNet [254], DeepVoice 1/2 [8, 87], Par. WaveNet [255], WaveRNN [150], HiFi-GAN [23]
3	DeepVoice 3 [270], Tacotron 2 [303], FastSpeech 1/2 [290, 292], WaveGlow [279], FloWaveNet [163]
4	Char2Wav [315], ClariNet [269], FastSpeech 2s [292], EATS [69], Wave-Tacotron [385], VITS [160]

- Neural based acoustic model (eg. RNN/LSTM, CNN or Transformers)
- Directly take *characters* or *phonemes* as input
- Acoustic features are simplified to *mel-spectrograms*
- Use neural models as vocoders
- **Advantages**
  - Do not require carefully designed linguistic or acoustic features
  - Better voice quality in terms of intelligibility and naturalness

FastSpeech 2: Fast and High-Quality End-to-End Text to Speech

Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, Tie-Yan Liu

ICLR 2021

Motivation

- Existing neural TTS models
    - Slow inference speed because of *autoregressive* decoders
        - Predicting current frame of mel-spectrogram depends on the previous output of the decoder
    - Less robust due to alignment errors in attention
        - leads to **word skipping** and **repeating**
    - Lack controllability, do not model variations in speech

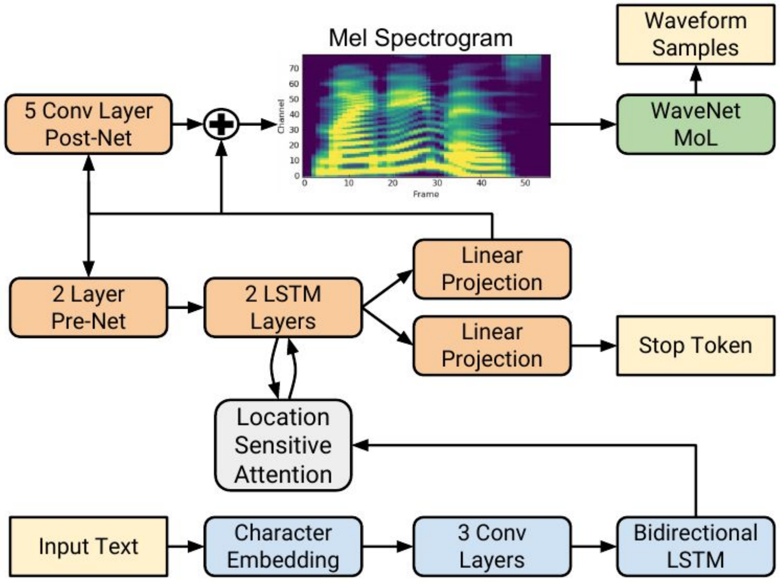
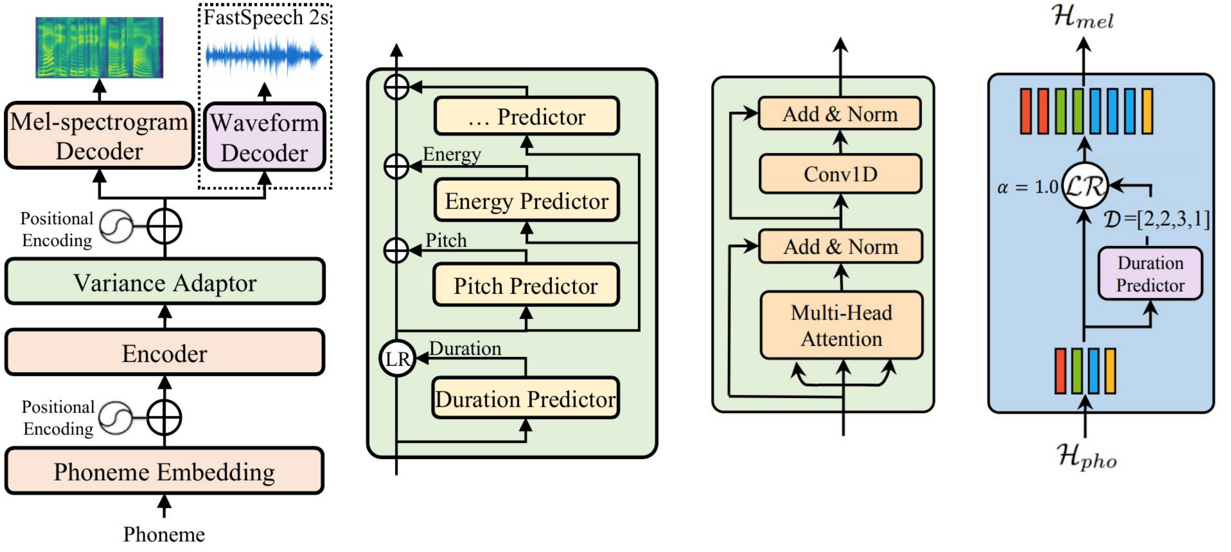


Fig. 1. Block diagram of the Tacotron 2 system architecture.

FastSpeech 2 Architecture

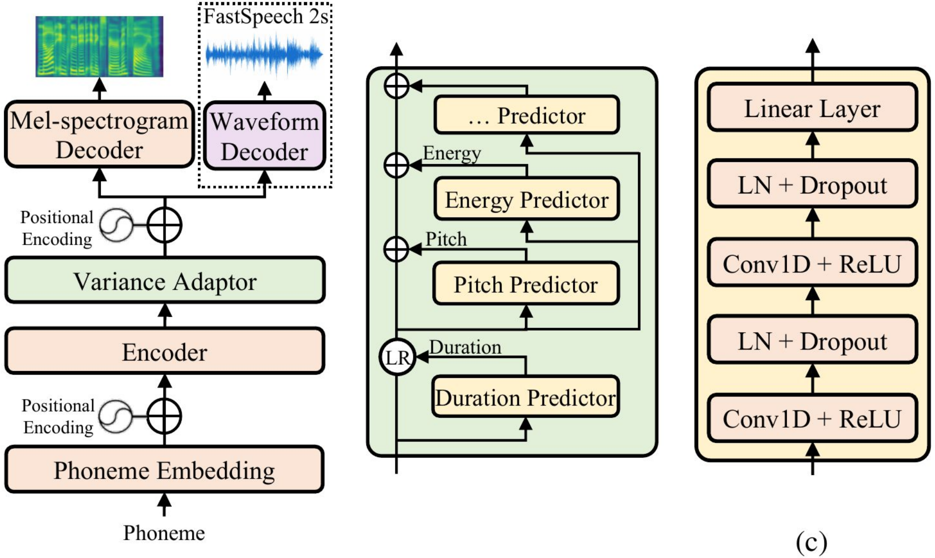
- Encoder
  - Stack of feed-forward transformer (FFT) blocks
- Variance Adaptor
  - Adds variance information eg. *duration*, *pitch*, *energy* to the phoneme hidden sequences
- Non-autoregressive generation of mel-spectrograms
  - Sequences are generated in parallel without depending on previous elements
  - Possible because of length regulator



(a) FastSpeech 2 (b) Variance adaptor (b) FFT Block (c) Length Regulator

Variance Adaptor

- Extract duration, pitch, and energy from the recordings
- Duration is used for expanding the hidden sequence
- Pitch and energy values are converted into embedding vectors that are then added to the phoneme hidden sequence
- During training,
  - Use ground-truth duration, pitch, and energy for producing target speech
  - Train the predictors with ground-truth as target
- At inference,
  - Use duration, pitch, and energy predicted from the trained predictors for



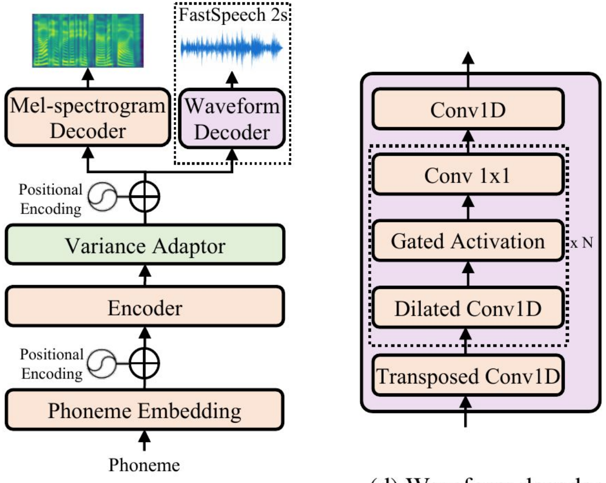
(a) FastSpeech 2

(b) Variance adaptor

(c) Duration/pitch/energy predictor

FastSpeech 2s (End-to-End)

- Challenges in text-to-waveform generation
  - Large information gap between input and output
    - Waveforms have more variance information that mel-spectrograms (eg. phase information)
  - Difficult to train on full text sequences
    - Waveforms contain too many samples (difficult to fit into memory)
    - Partial text sequences do not capture enough context for speech synthesis
- Waveform Decoder in FastSpeech 2s
  - Adversarial loss in decoder to implicitly recover the phase information
  - Leverages Mel-spectrogram
    - Mel-spectrogram decoder is trained on full text sequences
    - Waveform decoder is trained on sliced hidden sequences
    - Mel-spectrogram decoder is discarded after training



(a) FastSpeech 2

(d) Waveform decoder

Experiments

- Dataset
  - LJSpeech dataset
  - 13100 audio clips and corresponding text transcripts
  - Text sequence is converted into phoneme sequence
  - Raw waveforms are converted into 80-dimensional mel-spectrograms with frame size and hop size of 1024 and 256 with respect to sample rate 22050
- Implementation Details
  - Stack of 4 FFT blocks in encoder and melspectrogram decoder
  - Model is optimized with mean absolute error (MAE) on mel-spectrograms
  - Parallel WaveGAN is used as vocoder (mel-spectrogram → waveform)

Results: Audio Quality

Method	MOS
GT	4.30 ± 0.07
GT (Mel + PWG)	3.92 ± 0.08
Tacotron 2 (Shen et al., 2018) (Mel + PWG)	3.70 ± 0.08
Transformer TTS (Li et al., 2019) (Mel + PWG)	3.72 ± 0.07
FastSpeech (Ren et al., 2019) (Mel + PWG)	3.68 ± 0.09
FastSpeech 2 (Mel + PWG)	3.83 ± 0.08
FastSpeech 2s	3.71 ± 0.09

(a) The MOS with 95% confidence intervals.

Method	CMOS
FastSpeech 2	0.000
FastSpeech	-0.885
Transformer TTS	-0.235

(b) CMOS comparison.

Table 1: Audio quality comparison.

Results: Training Time & Inference Speed

Method	Training Time (h)	Inference Speed (RTF)	Inference Speedup
Transformer TTS (Li et al., 2019)	38.64	9.32×10^{-1}	/
FastSpeech (Ren et al., 2019)	53.12	1.92×10^{-2}	$48.5\times$
FastSpeech 2	**17.02**	1.95×10^{-2}	$47.8\times$
FastSpeech 2s	92.18	$\mathbf{1.80 \times 10^{-2}}$	$\mathbf{51.8}\times$

Table 2: The comparison of training time and inference latency in waveform synthesis. The training time of *FastSpeech* includes teacher and student training. RTF denotes the real-time factor, that is the time (in seconds) required for the system to synthesize one second waveform. The training and inference latency tests are conducted on a server with 36 Intel Xeon CPUs, 256GB memory, 1 NVIDIA V100 GPU and batch size of 48 for training and 1 for inference. Besides, we do not include the time of GPU memory garbage collection and transferring input and output data between the CPU and the GPU. The speedup in waveform synthesis for FastSpeech is larger than that reported in Ren et al. (2019) since we use Parallel WaveGAN as the vocoder which is much faster than WaveGlow.

More Accurate Variance Information

Method	σ	γ	$\mathcal{K}$	DTW
GT	54.4	0.836	0.977	/
Tacotron 2	44.1	1.28	1.311	26.32
TransformerTTS	40.8	0.703	1.419	24.40
FastSpeech	50.8	0.724	-0.041	24.89
FastSpeech 2	**54.1**	0.881	**0.996**	24.39
FastSpeech 2 - CWT	42.3	0.771	1.115	25.13
FastSpeech 2s	53.9	**0.872**	0.998	**24.37**

Table 3: Standard deviation (σ), skewness (γ), kurtosis ($\mathcal{K}$) and average DTW distances (DTW) of pitch in ground-truth and synthesized audio.

Method	*FastSpeech*	*FastSpeech 2*	*FastSpeech 2s*
MAE	0.142	0.131	0.133

Table 4: The mean absolute error (MAE) of the energy in synthesized speech audio.

Ablation Study

Setting	CMOS
FastSpeech 2	0
FastSpeech 2 - energy	-0.040
FastSpeech 2 - pitch	-0.245
FastSpeech 2 - pitch - energy	-0.370

(a) CMOS comparison for FastSpeech 2.

Setting	CMOS
FastSpeech 2s	0
FastSpeech 2s - energy	-0.160
FastSpeech 2s - pitch	-1.130
FastSpeech 2s - pitch - energy	-1.355

(b) CMOS comparison for FastSpeech 2s.

Table 6: CMOS comparison in the ablation studies.