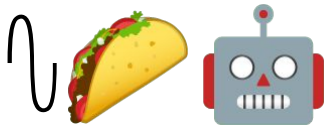
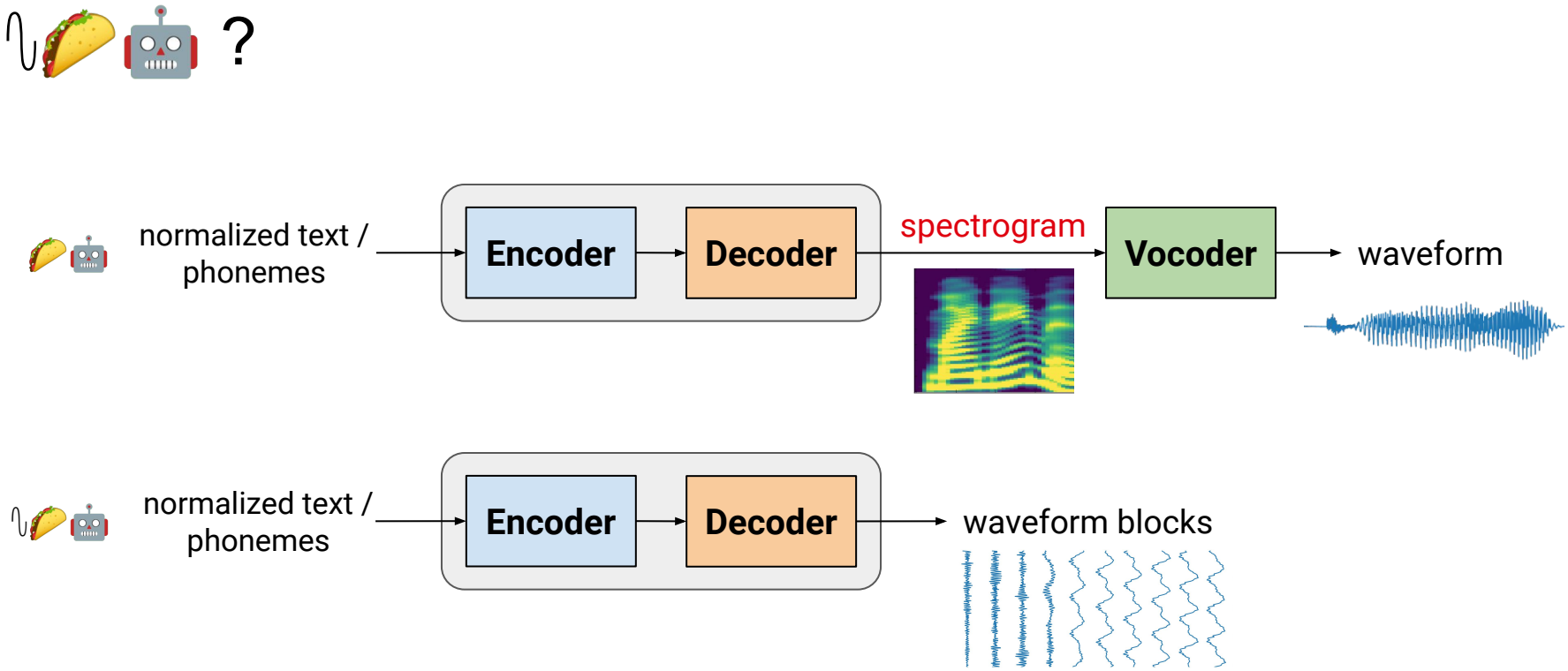
Wave-Tacotron: Spectrogram-free end-to-end text-to-speech synthesis



R. J. Weiss, R. J. Skerry-Ryan, E. Battenberg, S. Mariooryad, and D. P. Kingma

IEEE ICASSP 2021

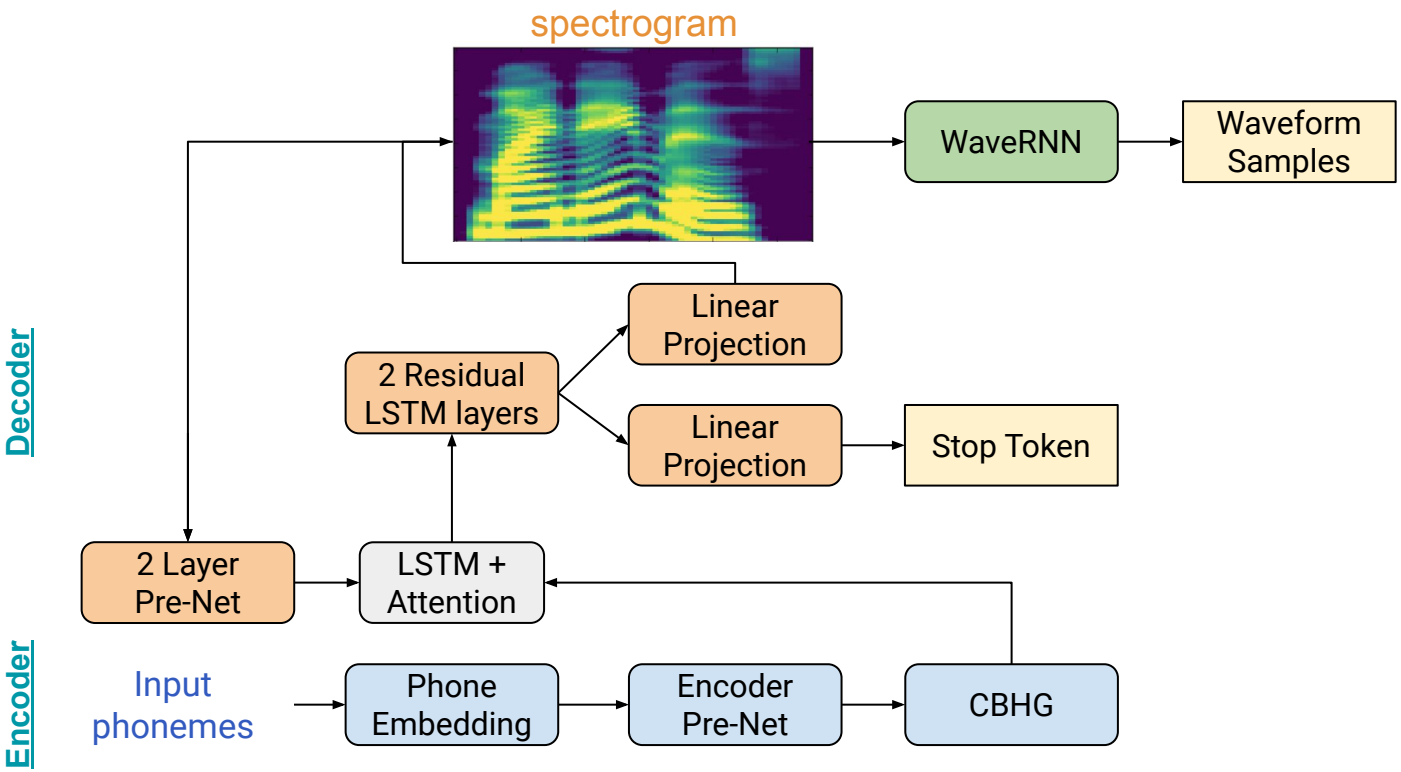
Google Research



- TTS* in **one sequence-to-sequence model**
  - no vocoder
- Directly predict sequence of ~10-50ms *waveform blocks*
  - no spectrograms anywhere
- Goal: fast waveform generation

* really normalized-text- or phoneme-to-speech

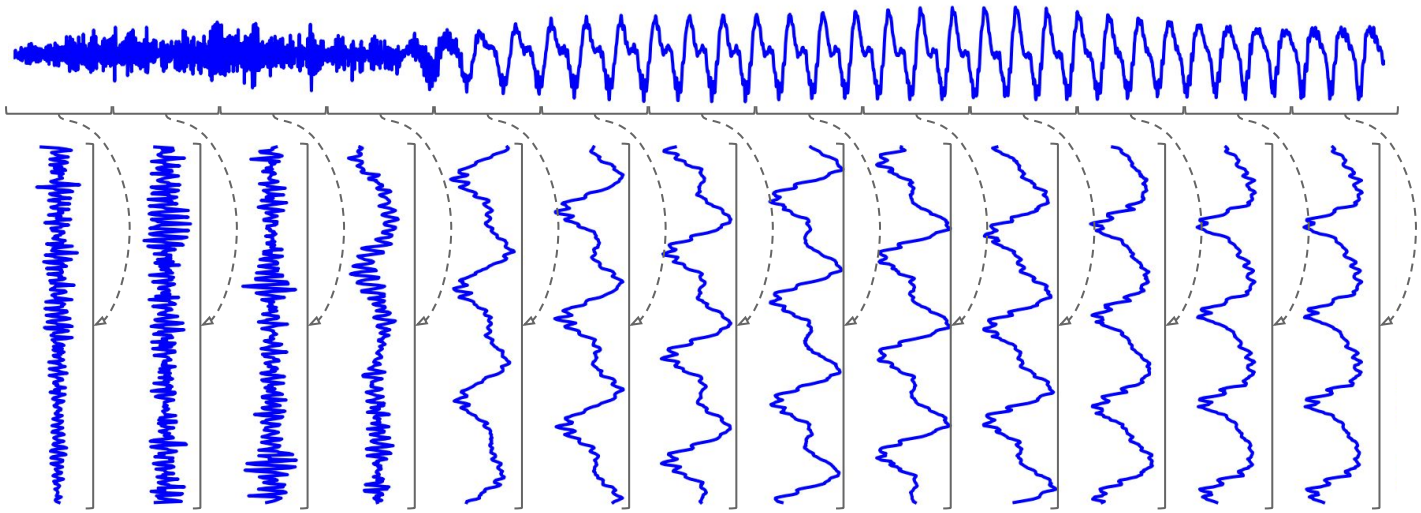
Tacotron



- **Encoder** maps input phonemes to latent representation
- **Autoregressive decoder** generates mel spectrogram one frame at a time
- Separately trained vocoder network to invert spectrogram to waveform
  - e.g., WaveRNN, slow sample-by-sample autoregressive network

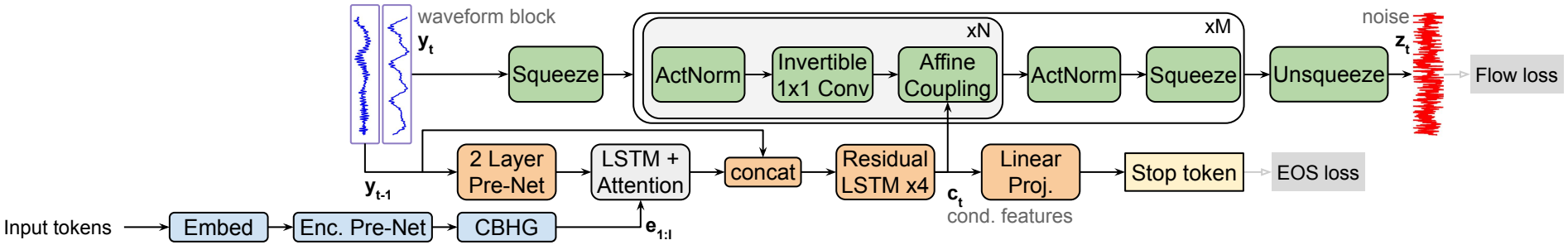
Wang, et al., Tacotron: Towards End-to-End Speech Synthesis. Interspeech 2017.
Shen, et al., Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions. ICASSP 2018.
Kalchbrenner, et al., Efficient Neural Audio Synthesis. ICML 2018.

s/spectrogram frame/waveform block/



- Segment waveform into non-overlapping blocks
  - K = 960 samples, 40 ms at 24 kHz sample rate
- *Block-autoregressive* generation, each decoder step generates a new block
  - waveform samples in each block generated in parallel, much faster than WaveRNN

Architecture



- Replace post-net and vocoder with conditional **normalizing flow**

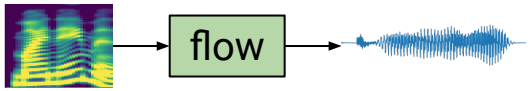
$$P(\mathbf{y}_t \mid \mathbf{c}_t) = P(\mathbf{y}_t \mid \mathbf{y}_{1:t-1}, \mathbf{e}_{1:l})$$
$$= P(\mathbf{y}_t \mid \text{previous waveform blocks, text})$$

- Tacotron encoder/decoder predicts flow conditioning features

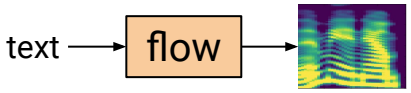
- Train end-to-end, maximize likelihood of training data

Related work

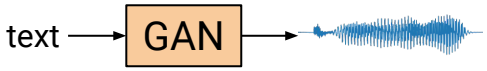
- Flow-based neural vocoders
  - generate waveforms from mel spectrograms
  - WaveGlow [Prenger et al., 2019], FloWaveNet [Kim et al., 2019] WaveFlow [Ping et al., 2020]



- Flow TTS models
  - generate mel spectrograms from text
  - parallel: Flow-TTS [Miao et al., 2020], Glow-TTS [Kim, Kim, et al., 2020]
  - autoregressive: Flowtron [Valle et al., 2021]

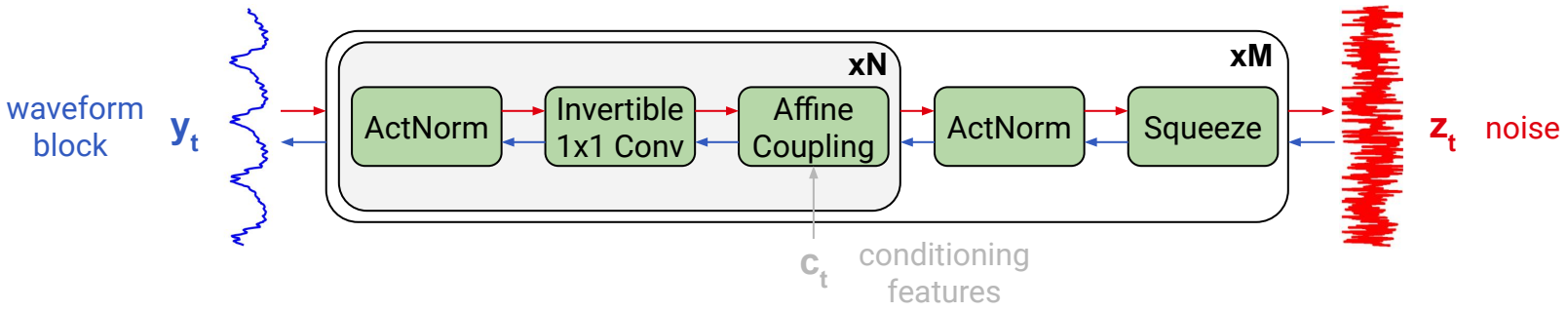


- Direct-to-waveform TTS
  - adversarial training
  - use mel spectrograms to help learn alignment, or as loss functions
  - EATS [Donahue et al., 2021], Fastspeech 2s [Ren et al., 2021]



Prenger, et al., WaveGlow: A Flow-based Generative Network for Speech Synthesis. ICASSP 2019.
Kim, et al., FloWaveNet : A Generative Flow for Raw Audio. ICML 2019.
Ping, et al., Waveflow: A compact flow-based model for raw audio. ICML 2020.
Miao, et al., Flow-TTS: A non-autoregressive network for text to speech based on flow. ICASSP 2020.
Kim, et al., Glow-TTS: A generative flow for text-to-speech via monotonic alignment search. NeurIPS 2020.
Valle, et al., Flowtron: an autoregressive flow-based generative network for text-to-speech synthesis. ICLR 2021.
Donahue, et al., End-to-end Adversarial Text-to-Speech, ICLR 2021.
Ren, et al., FastSpeech 2: Fast and High-Quality End-to-End Text to Speech, ICLR 2021.

Normalizing flow



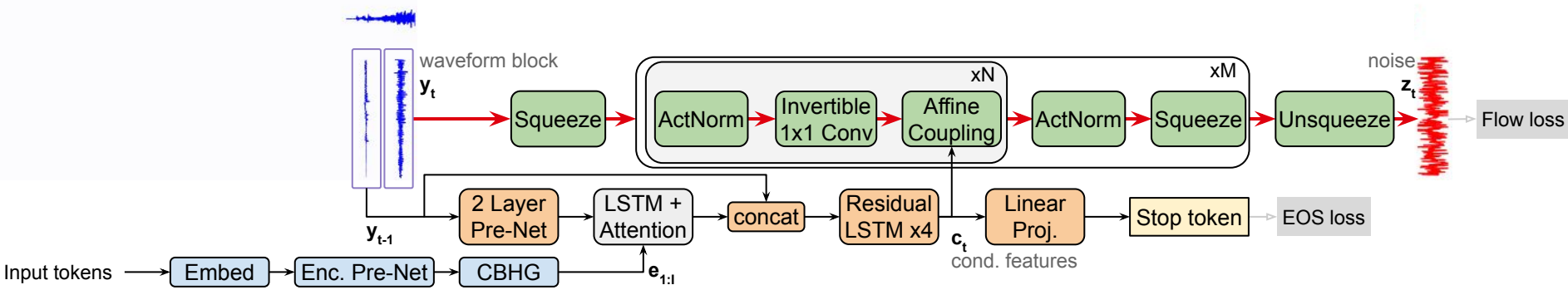
- Model **joint distribution** of K samples: $P(y_{t1}, y_{t2}, \ldots, y_{tK} \mid c_t)$
  - multiscale architecture, similar to FloWaveNet [Kim et al., 2019] neural vocoder
    - M = 5 stages, each with N = 12 steps

- Invertible network
  - training: transform waveform block into noise
  - sampling: transform noise sample into waveform block using inverse

- Change of variables $y_t = g(z_t; c_t)$
- Maximize likelihood $P(y_t \mid c_t) = P(z_t \mid c_t) \ |\det(dz_t / dy_t)|$

Kim, et al., FloWaveNet : A Generative Flow for Raw Audio. ICML 2019.

Training

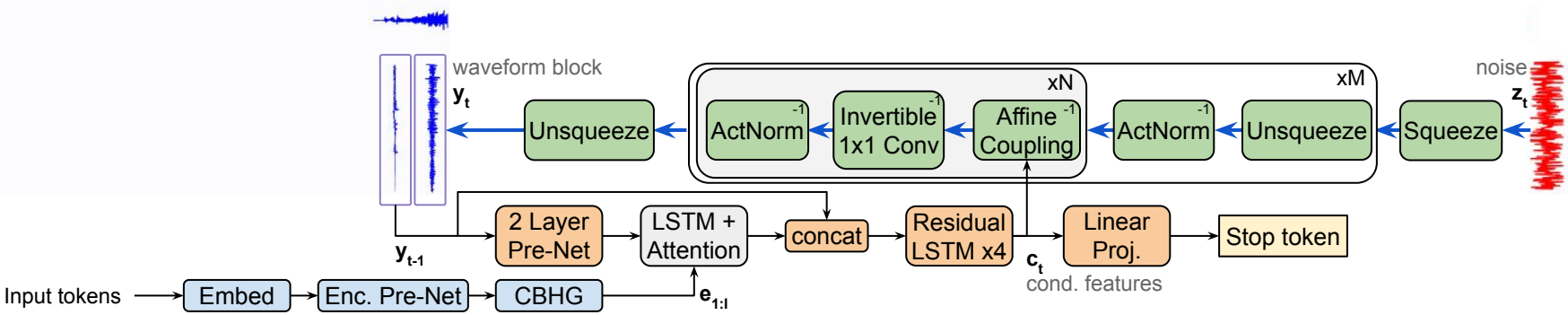


- Teacher forced conditioning
- At each step: transform waveform block y_t into noise z_t
- Flow loss: $-\log P(\mathbf{y}) = \sum_t -\log P(\mathbf{y}_t \mid c_t)$

  $= \sum_t -\log N(g^{-1}(\mathbf{y}_t; c_t); \mathbf{0}, \mathbf{I}) - \log \left|\det\left(dg^{-1}(\mathbf{y}_t; c_t) / d\mathbf{y}_t\right)\right|$

  spherical Gaussian — Jacobian determinant
- EOS *stop token* classifier loss: P(t is last frame)

Sampling



- Invert the flow network
  - take inverse of each layer, reverse order
- At each step
  - sample a noise vector
  - pass through flow to generate waveform block
  - autoregressive conditioning on previous output $\mathbf{y}_{t-1}$
- concatenate blocks $\mathbf{y}_t$ to form final signal
  - $\mathbf{y}$ = vstack($\mathbf{y}_t$)

$$\mathbf{z}_t \sim N(\mathbf{0}, \mathbf{I})$$

$$\mathbf{y}_t = g(\mathbf{z}_t; c_t)$$

Experiment configuration

- Systems
  - Tacotron-PN (postnet) + Griffin-Lim (à la Tacotron)
  - Tacotron + WaveRNN (à la Tacotron 2)
  - Tacotron + Flow vocoder
    - identical Tacotron model
    - fully parallel vocoder (similar flow architecture to Wave-Tacotron, 6 stages)
  - Wave-Tacotron

- Datasets - US English, single female speaker, sampled at 24 kHz
  - Proprietary
    - 39 hours training
    - average duration: 3.3 seconds
  - LJ Speech
    - 22 hours training
    - average duration: 10 seconds

Generation speed

Model	K	Vocoder	TPU	CPU
Tacotron-PN		Griffin-Lim, 100 iterations	0.14	0.88
Tacotron-PN		Griffin-Lim, 1000 iterations	1.11	7.71
Tacotron		WaveRNN	5.34	63.38
Tacotron		Flowcoder	0.49	0.97
Wave-Tacotron	13.3ms	–	0.80	5.26
Wave-Tacotron	26.6ms	–	0.64	3.25
Wave-Tacotron	40.0ms	–	0.58	2.52
Wave-Tacotron	53.3ms	–	0.55	2.26

- Seconds to generate 5 seconds of speech
  - 90 input tokens, batch size 1
- Wave-Tacotron ~10x faster than real-time on TPU (2x on CPU)
  - slower as frame size K decreases (more autoregressive steps)
- ~10x faster than Tacotron + WaveRNN on TPU (25x on CPU)

Experiments: proprietary data

- Subjective listening tests rating speech naturalness
  - MOS on 5 point scale
- Tacotron + WaveRNN best
  - char / phoneme roughly par
- Wave-Tacotron trails by ~0.2 points
  - phoneme > char
  - network uses capacity to model detailed waveform structure instead of pronunciation?
- Large gap to Tacotron-PN and Tacotron + Flowcoder

Model	Vocoder	Input	MOS
Ground truth	–	–	4.56 ± 0.04
Tacotron-PN	Griffin-Lim	char	3.68 ± 0.08
Tacotron-PN	Griffin-Lim	phoneme	3.74 ± 0.07
Tacotron	WaveRNN	char	4.36 ± 0.05
Tacotron	WaveRNN	phoneme	$\mathbf{4.39 \pm 0.05}$
Tacotron	Flowcoder	char	3.34 ± 0.07
Tacotron	Flowcoder	phoneme	3.31 ± 0.07
Wave-Tacotron	–	char	4.07 ± 0.06
Wave-Tacotron	–	phoneme	4.23 ± 0.06

Ground truth

Tacotron-PN + Griffin-Lim

Tacotron + WaveRNN

Tacotron + Flowcoder

Wave-Tacotron

source: https://google.github.io/tacotron/publications/wave-tacotron/#architecture-comparison-on-single-speaker-proprietary-dataset

Experiments: LJ Speech

- Longer average utterance duration, halve training batch size to 128
- Similar trend to models trained on proprietary data
  Tacotron+WaveRNN >> Wave-Tacotron > others
- Larger gap between Tacotron+WaveRNN and Wave-Tacotron
  - need more training data for more end-to-end task?
  - and/or better tuning?

Model	Vocoder	Input	MOS
Ground truth	–	–	4.51 ± 0.05
Tacotron-PN	Griffin-Lim	char	3.26 ± 0.11
Tacotron	WaveRNN	char	$\mathbf{4.47 \pm 0.06}$
Tacotron	Flowcoder	char	3.11 ± 0.10
Wave-Tacotron	–	char	3.56 ± 0.09

Ground truth

Tacotron-PN + Griffin-Lim

Tacotron + WaveRNN

Tacotron + Flowcoder

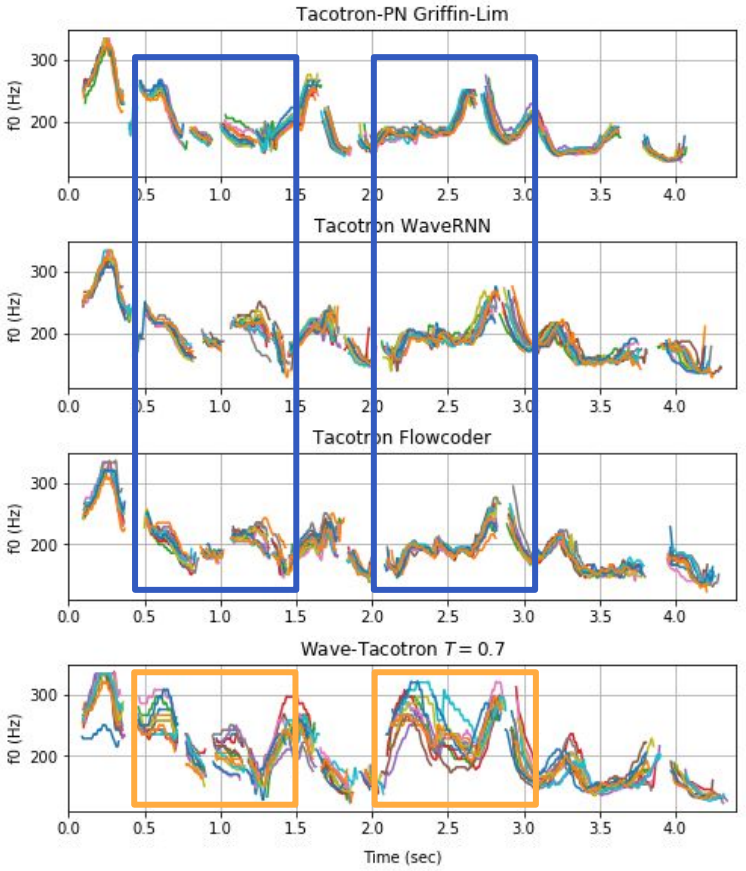
Wave-Tacotron

source: https://google.github.io/tacotron/publications/wave-tacotron/#architecture-comparison-on-single-speaker-ljspeech-dataset

Sample variability

- Baseline Tacostron generate very consistent samples
  - same prosody every time

- Wave-Tacotron has high variance
  - captures multimodal training distribution?
    - Tacotron regression loss collapses to single *prosody mode*?
  - similar pattern in Flowtron [Valle et al., 2021]



Valle, et al., Flowtron: an Autoregressive Flow-based Generative Network for Text-to-Speech Synthesis . ICLR 2021.

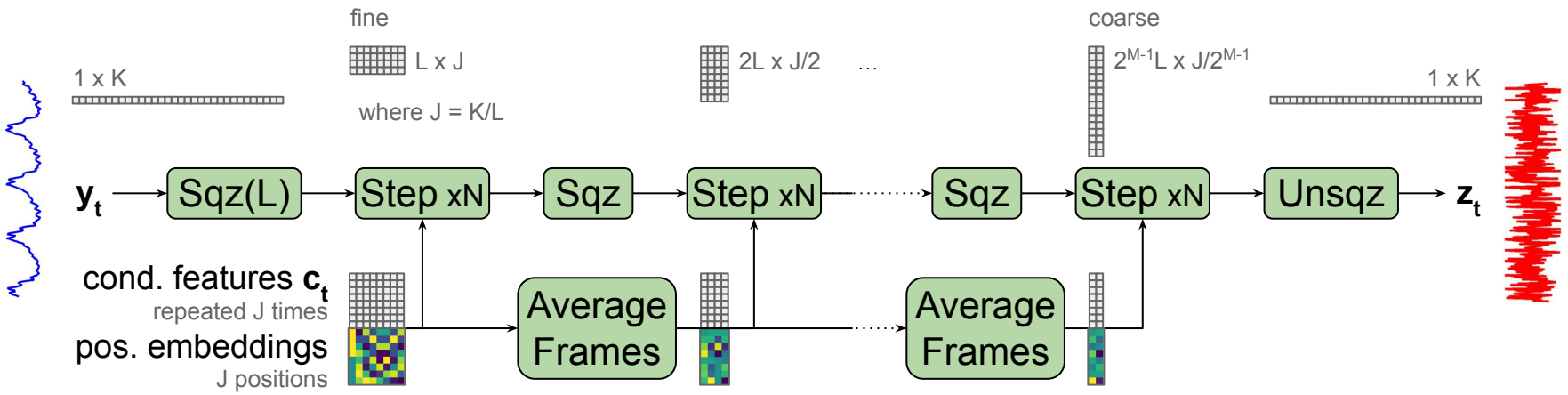
Summary

- Sequence-to-sequence text-to-speech synthesis without spectrograms
  - block-autoregressive normalizing flow
- End-to-end training, maximizing likelihood
- High fidelity output
  - trails Tacotron + WaveRNN baseline
  - higher sample variation, captures multimodal training data?
- ~10x faster than real-time synthesis on TPU

Sound examples:
https://google.github.io/tacotron/publications/wave-tacotron

Extra slides

Multiscale flow



- Squeeze K sample waveform block into K/L **frames**
  - base frame length L = 10 samples
- Squeeze after each stage
  - doubles dimension, halves frame rate
- M = 5 stages, each processes signal at different scale
  - N = 12 steps per stage
  - deep convnet: M N = 60 total steps

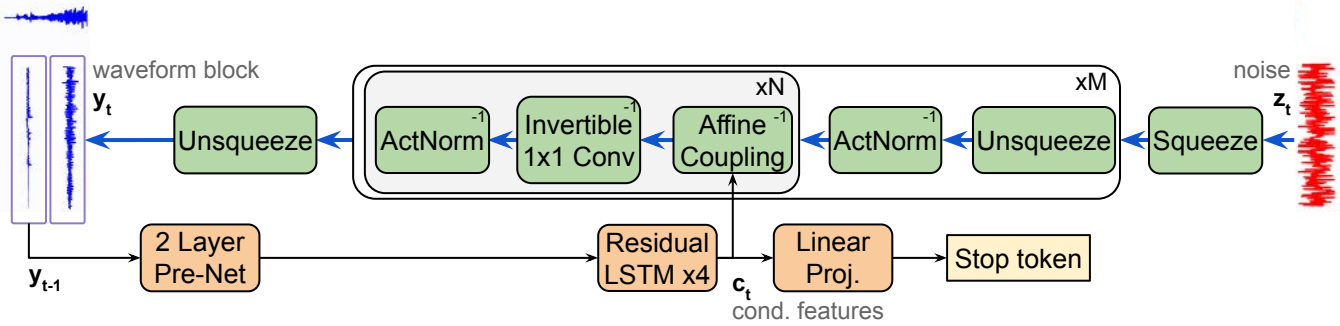
Dinh, et al., NICE: Non-linear independent components estimation. ICLR 2015.

Experiments: Ablations

- 2 layer decoder LSTM, 256 flow channels
- Optimal sampling temperature T = 0.7
  - $\mathbf{z}_t \sim N(\mathbf{z}_t; \mathbf{0}, T\mathbf{I})$
- Architecture details
  - pre-emphasis, position embedding
  - flow width
  - number of stages / multiscale architecture
- Varying block size K
  - quality starts degrading if K > 40 ms

Model	R	M	N	MOS
Base $T = 0.8$	3	5	12	4.01 ± 0.06
$T = 0.6$	3	5	12	4.12 ± 0.06
$T = 0.7$	3	5	12	4.16 ± 0.06
$T = 0.9$	3	5	12	3.77 ± 0.07
no pre-emphasis	3	5	12	3.85 ± 0.06
no position emb.	3	5	12	3.70 ± 0.07
no skip connection	3	5	12	–
128 flow channels	3	5	12	3.31 ± 0.07
30 steps, 5 stages	3	5	6	3.11 ± 0.07
60 steps, 4 stages	3	4	15	3.50 ± 0.07
60 steps, 3 stages	3	3	20	2.44 ± 0.07
$K = 320$ (13.33 ms)	1	5	12	4.05 ± 0.06
$K = 640$ (26.67 ms)	2	5	12	4.06 ± 0.06
$K = 1280$ (53.3 ms)	4	5	12	3.55 ± 0.07

Unconditional generation



- Remove encoder and attention, condition only on previous samples

  $P(\mathbf{y_t} \mid \mathbf{c_t}) = P(\mathbf{y_t} \mid \mathbf{y_{1:t-1}}) = P(\mathbf{y_t} \mid$ previous waveform blocks$)$

- Generates coherent syllables, occasional words

source: https://google.github.io/tacotron/publications/wave-tacotron/#unconditional-generation