Topics on Neural Speech Synthesis

Vassilis Tsiaras
University of Crete

November 2023

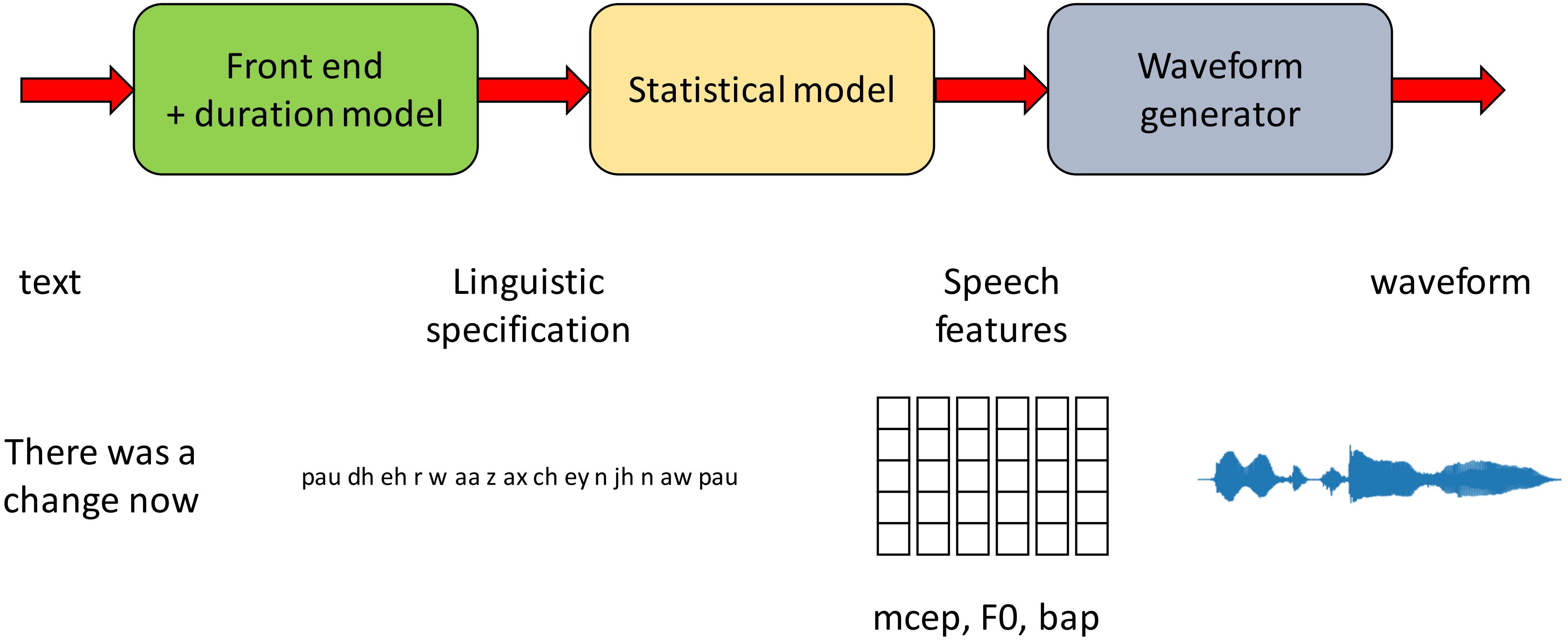
Outline

- Introduction
- Two stage pipeline systems
  - Acoustic models
    - Tacotron 2
    - Transformer TTS
    - FastSpeech 1 and 2
  - Neural vocoders
    - Sequential generation
      - WaveNet
      - WaveRNN
    - Parallel generation
      - Artifacts
- Do end-to-end systems reduce the artifacts in parallel generation?
- Neural audio codec
- Speech synthesis with language models

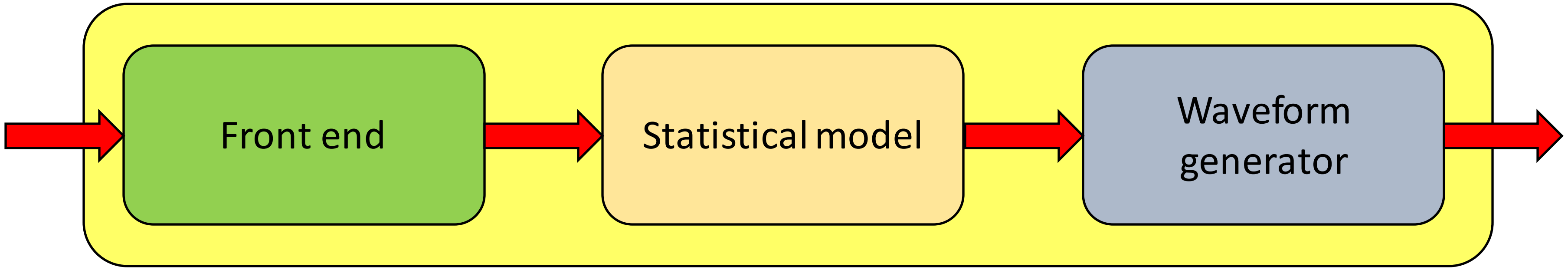
Introduction

- Traditional Text-To-Speech (TTS) systems are based on multi-stage, hand-engineered pipelines
- Neural network-based TTS models outperform conventional concatenative and statistical parametric approaches in terms of speech quality.
- Most popular neural network-based TTS systems have two-stage pipelines.
    1. Generate mel-scale spectrograms from text input.
    2. Synthesize speech from the generated mel-spectrograms using a separately trained neural vocoder.
- Also, there are End-to-end neural network-based TTS system.
    - Trained on <text, audio> pairs with minimal human annotation and effort.
- However, end-to-end models are harder to train and require a large number of high-quality recordings with transcriptions.

The classic three-stage pipeline of statistical parametric speech synthesis



The end-to-end problem

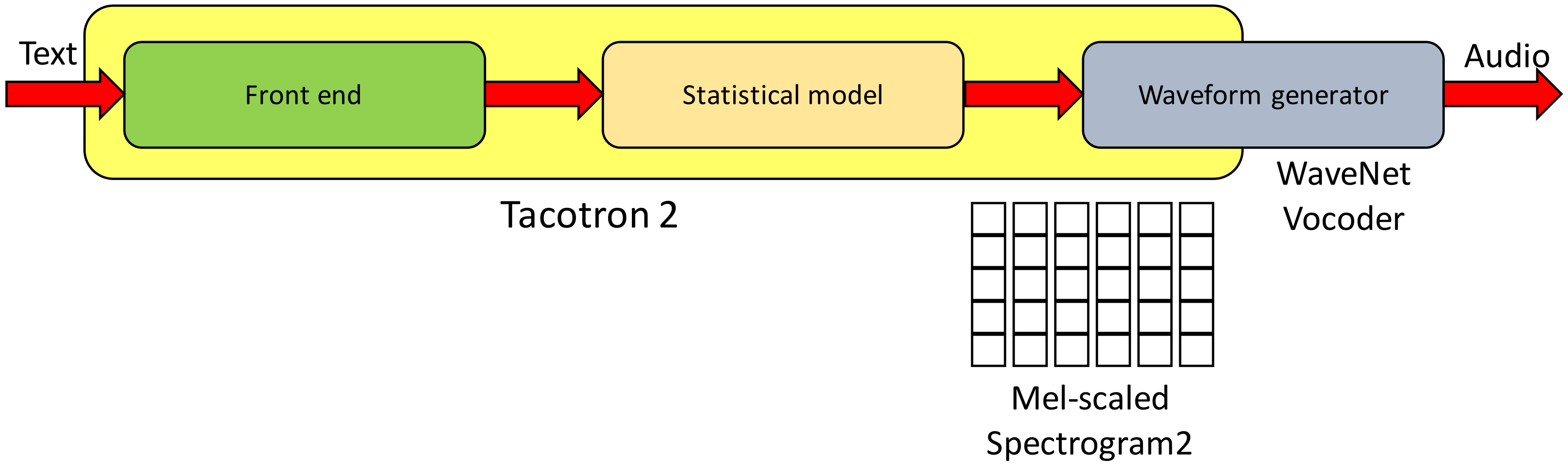


text

waveform

There was a
change now

Tacotron 2: A two stage pipeline



- **Tacotron 2** produces mel-scaled spectrograms, which are used as local conditioning to a WaveNet vocoder.

Outline

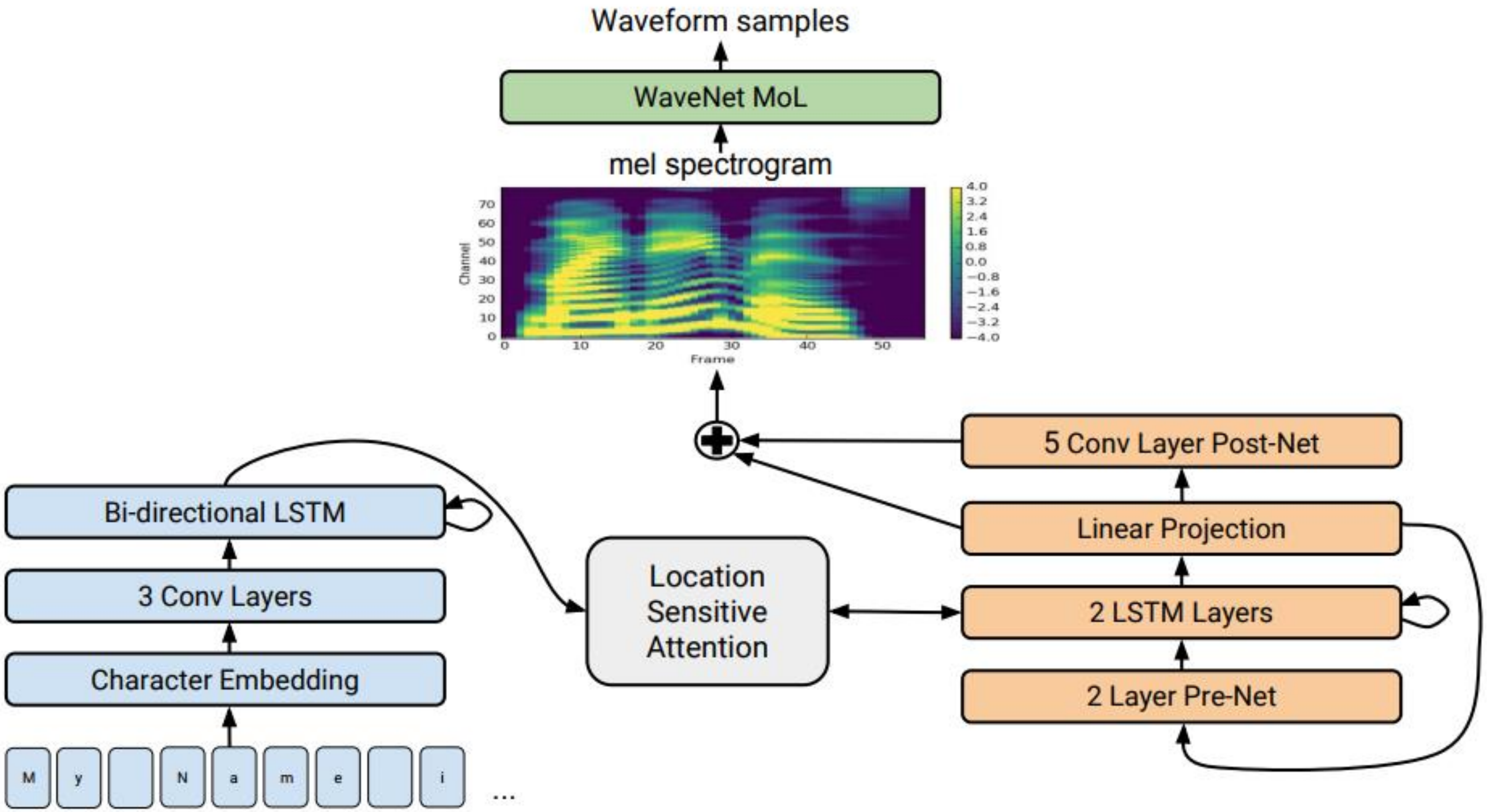
- Introduction

- Two stage pipeline systems
  - Acoustic models
    - Tacotron 2
    - Transformer TTS
    - FastSpeech 1 and 2
  - Neural vocoders
    - Sequential generation
      - WaveNet
      - WaveRNN
    - Parallel generation
      - Artifacts

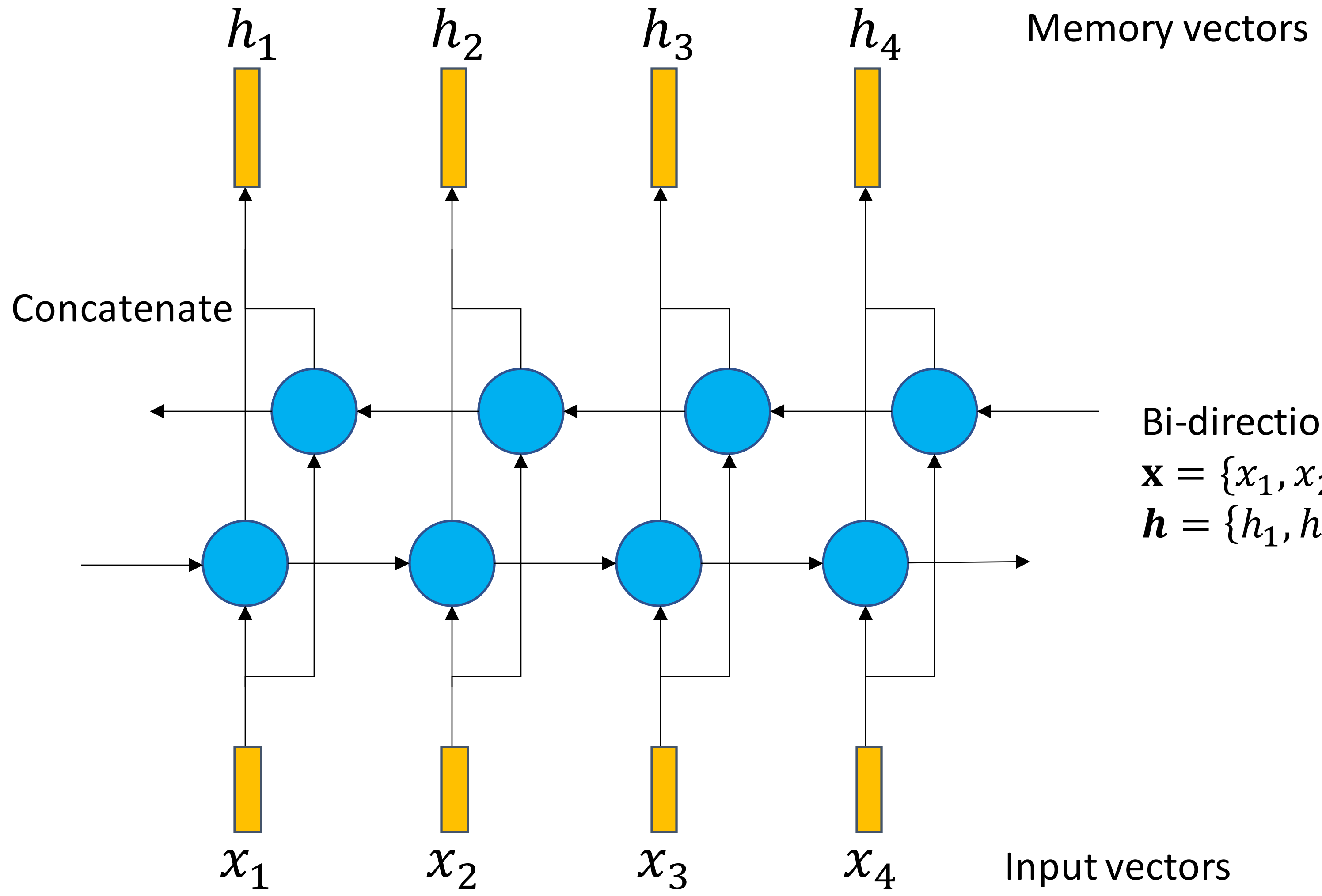
- Do end-to-end systems reduce the artifacts in parallel generation?

Tacotron 2 – A Sequence-to-Sequence model

- Tacotron 2 maps a sequence of letters to a sequence of 80-dimensional Mel-scaled spectrograms.
- Finally these spectrograms are converted to waveform using a WaveNet-like architecture.



Encoder

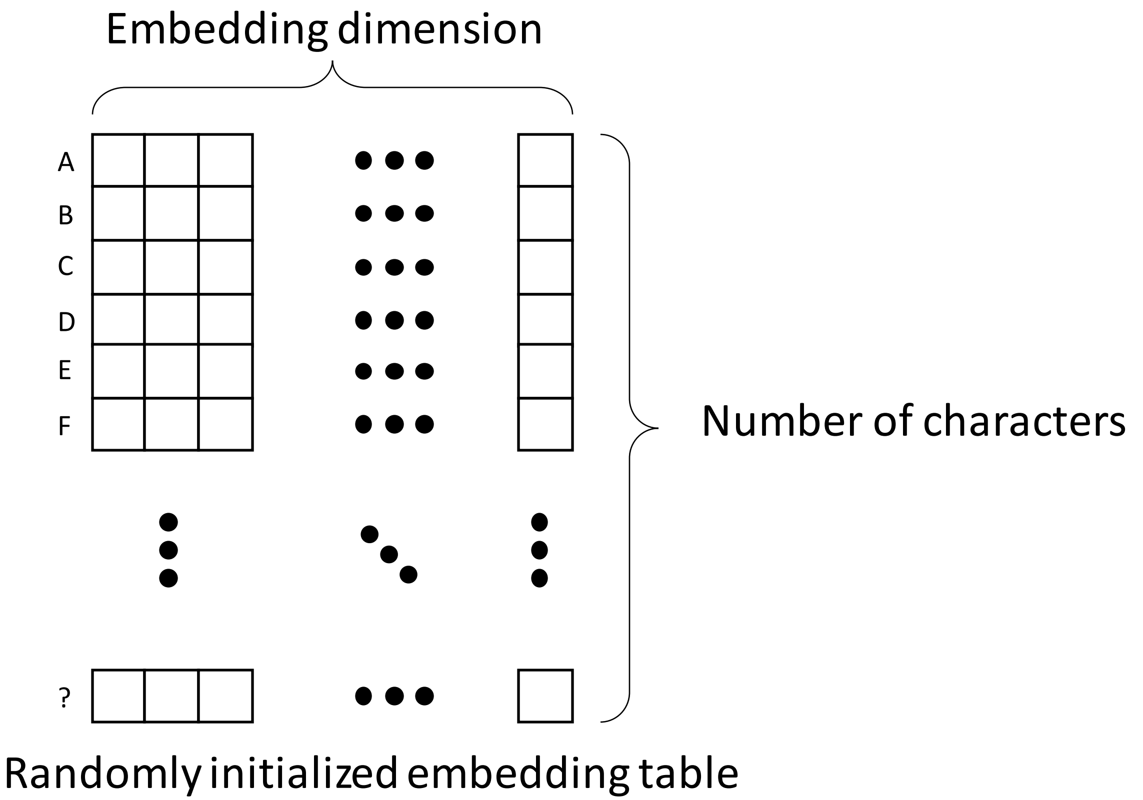


Bi-directional RNN based encoder takes the input sequence $\mathbf{x} = \{x_1, x_2, \dots, x_T\}$ and outputs the hidden states $\boldsymbol{h} = \{h_1, h_2, \dots, h_T\}$ (memory vectors).

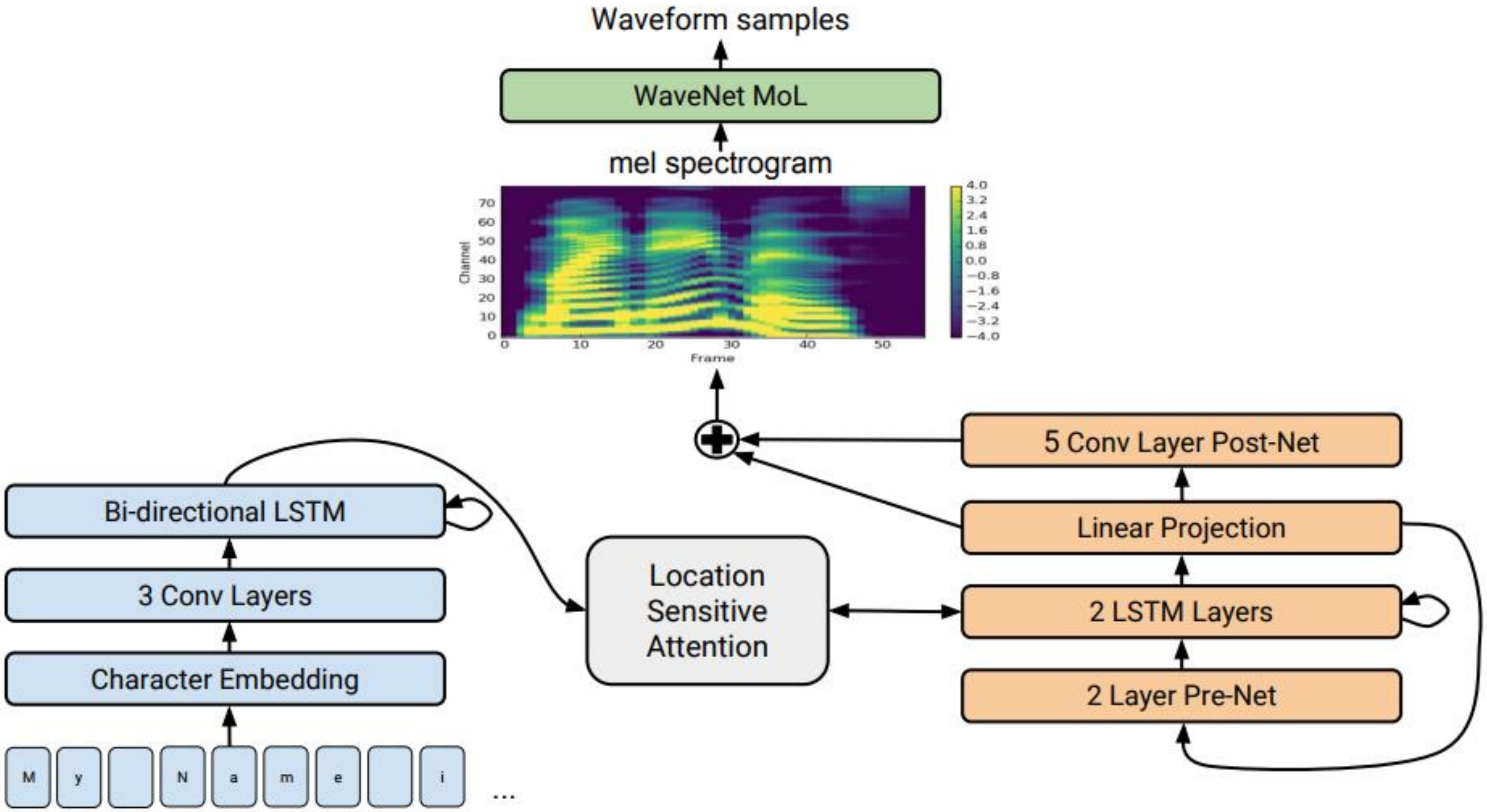
Note that a memory vector does not directly depend on the other memory vectors.
The RNNs in the implementation of the encoder extend the receptive field of a memory vector to the whole input sequence.
However, the use of RNNs prevent the parallel generation of the memory vectors.

Encoder: Character embedding

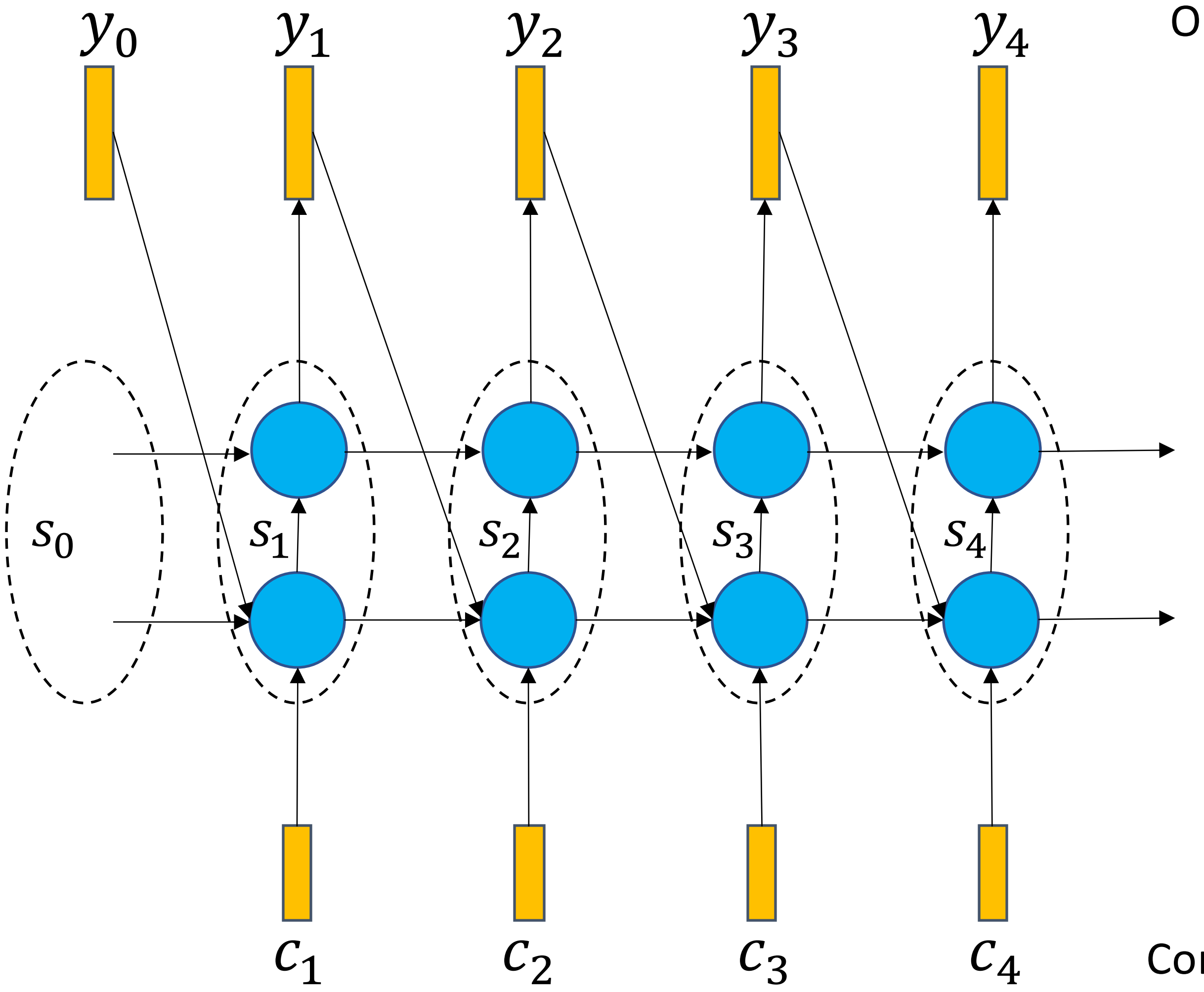
- Input characters are represented using a learned 512-dimensional character embedding.



Decoder: An autoregressive module



Decoder: Unrolling in time



Output vectors

$$s_i = Decoder(s_{i-1}, y_{i-1}, c_i)$$

$$y_i = OutputFunction(s_i)$$

Usually, a stack of two unidirectional RNNs

Context vectors which depend on the memory vectors of encoder

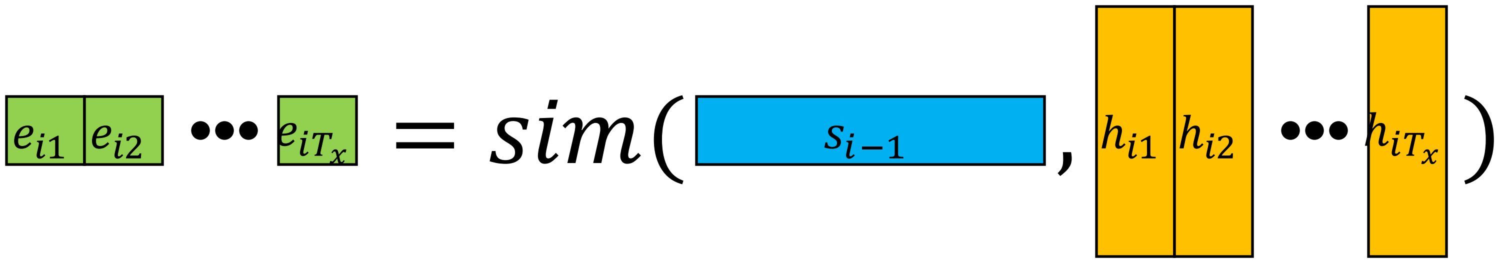
At each decoder output step i, inputs to the RNN cell are s_{i-1}, y_{i-1} and the context c_i and outputs the vectors s_i and y_i.

The attention mechanism

1. Alignment scores:

- The alignment model takes the encoded hidden states, h_j, and the previous decoder output, s_{i-1}, to compute a score, e_{ij}
- The score indicate how well the elements of the input sequence align with the current output at position, i

$$e_{ij} = sim(s_{i-1}, h_j), \qquad j = 1, \dots, T_x$$



2. Alignment weights or probabilities:

- The weights, a_{ij}, are computed by applying a softmax operation to the previously computed alignment scores:

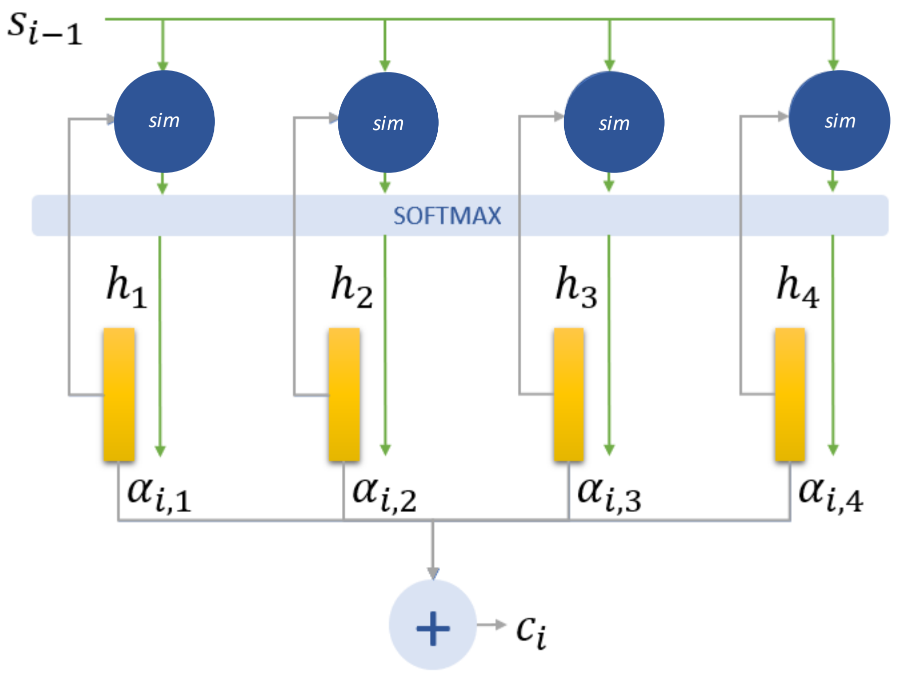
$$[a_{i1}, a_{i2}, \dots, a_{iT_x}] = softmax([e_{i1}, e_{i2}, \dots, e_{iT_x}]) = \frac{1}{Z}[\exp(e_{i1}), \exp(e_{i2}), \dots, \exp(e_{iT_x})], \qquad Z = \sum_{k=1}^{T_x} \exp(e_{ik})$$

The attention mechanism

3. Context vector

- A unique context vector, c_i, is fed into the decoder at each time step. It is computed by a weighted sum of all, T_x, encoder hidden states:

$$c_i = \sum_{j=1}^{T_x} a_{ij} h_j$$



Attention mechanisms

- Dot product attention (Luong)

  $$sim(s_{i-1}, h_j) = s_{i-1}^T h_j$$

- Multiplicative attention

  $$sim(s_{i-1}, h_j) = s_{i-1}^T W h_j$$

- Additive attention (Bahdanau)

  $$sim(s_{i-1}, h_j) = w^T \tanh(W_s s_{i-1} + W_h h_j + b)$$

- Location sensitive additive attention

  $$sim(s_{i-1}, h_j) = w^T \tanh(W_s s_{i-1} + W_h h_j + W_f f_{ij} + b)$$

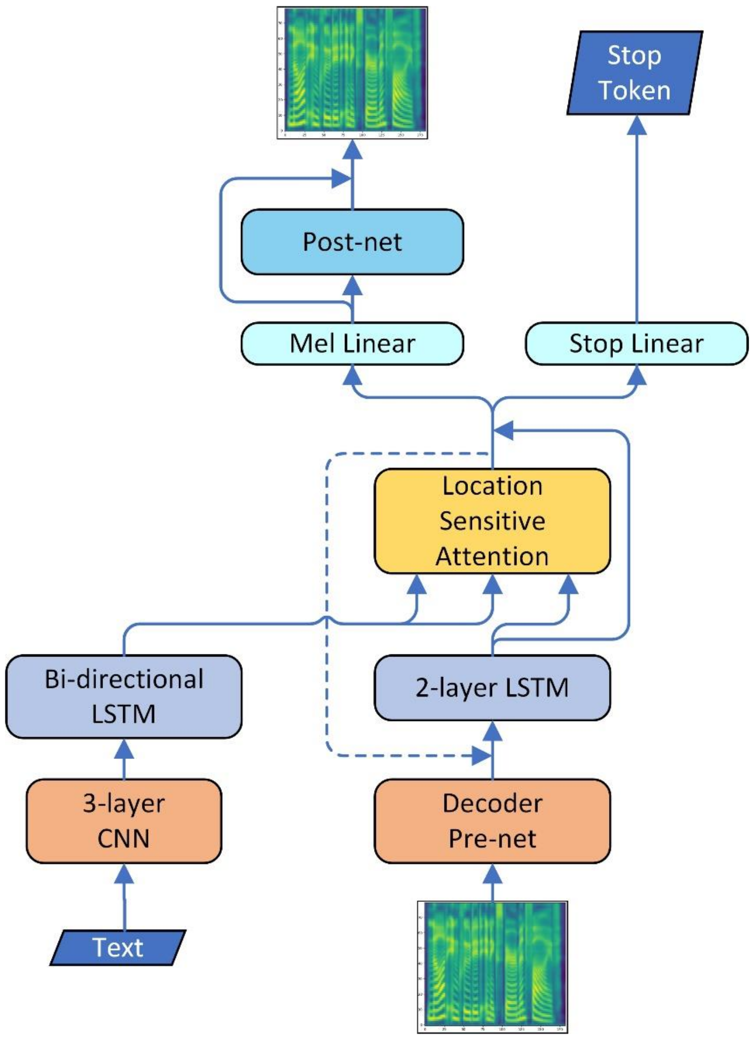
  $$\boldsymbol{f}_i = F * \boldsymbol{a}_{i-1}, \quad \text{where } \boldsymbol{a}_{i-1} = \left[a_{i-1,1}, \ldots, a_{i-1,T_x}\right]$$

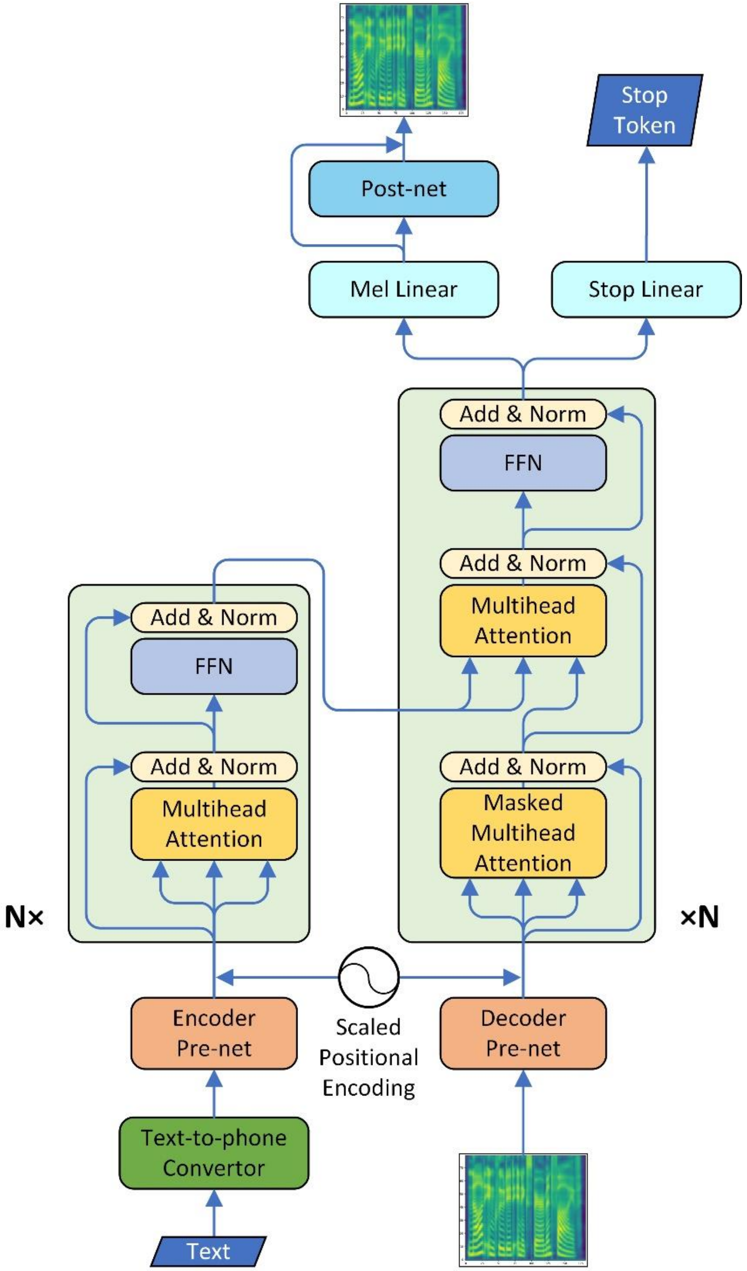
- Cumulative attention

  $$\boldsymbol{ca}_1 = \boldsymbol{a}_1, \qquad \boldsymbol{ca}_i = \boldsymbol{ca}_{i-1} + \boldsymbol{a}_i$$

- Monotonic attention

Tacotron 2 vs Transformer TTS

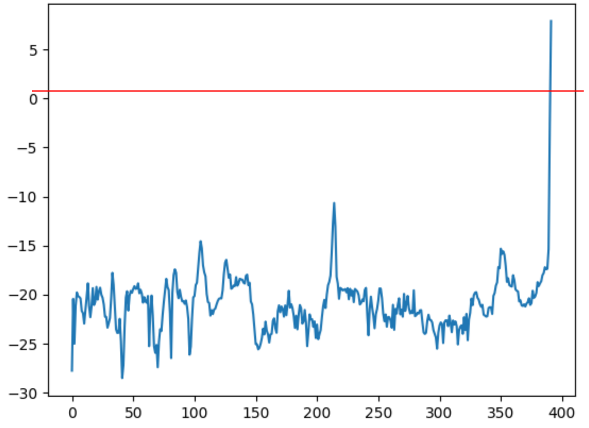




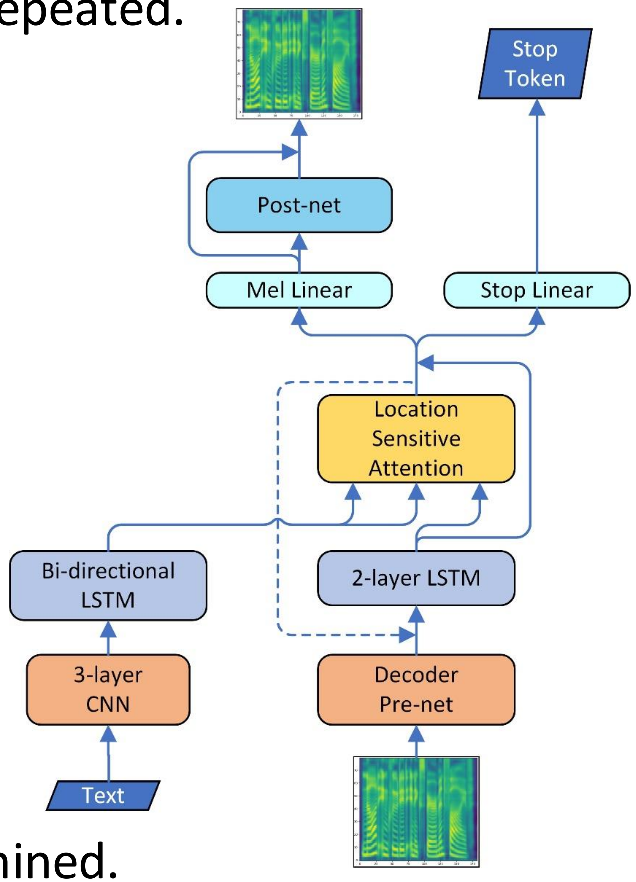
Issues with autoregressive acoustic models

- Tacotron and Transformer TTS face several challenges:
  - Slow inference speed for autoregressive generation.
  - Synthesized speech is usually not robust, with words being skipped and repeated.

Text: Bad speech synthesis



Stop token plot



- Lack of controllability since the generation length is automatically determined.
  - The voice speed and the prosody (such as word breaks) cannot be adjusted.

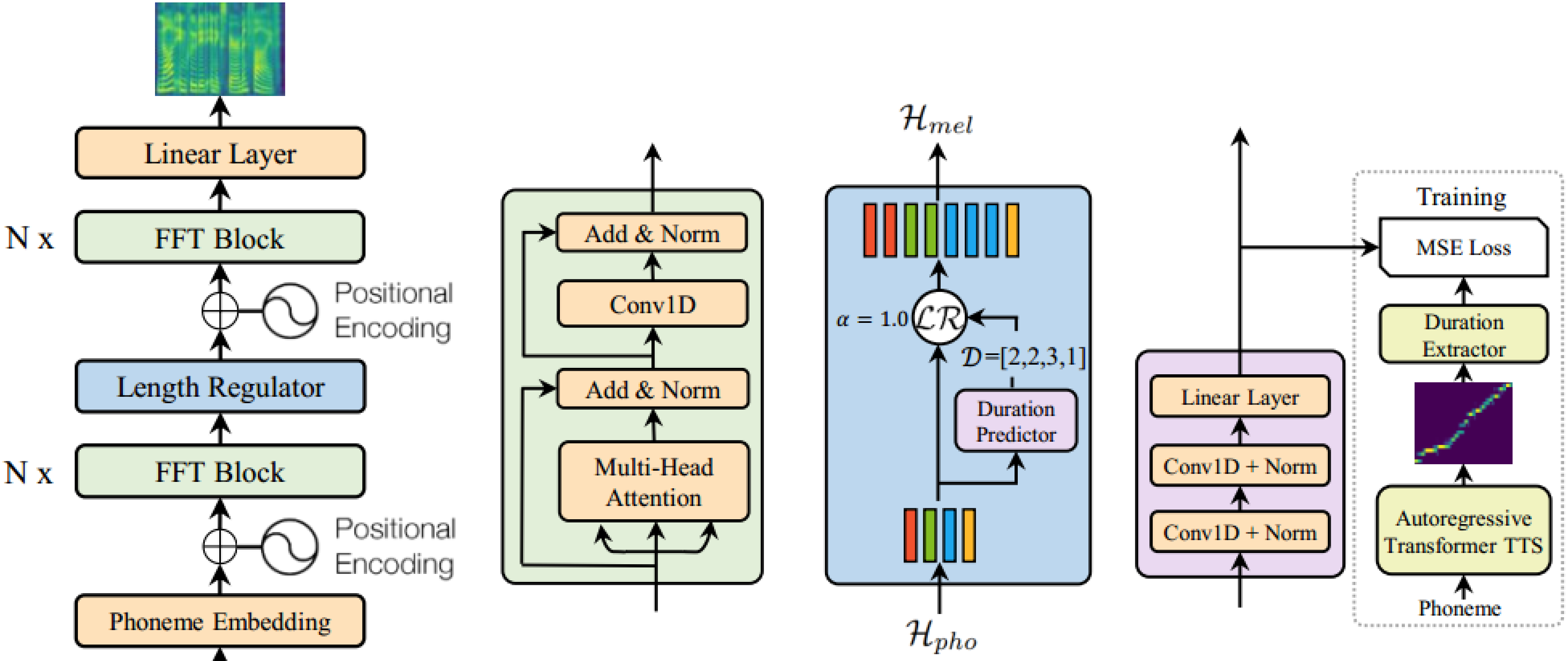
Non-autoregressive acoustic models

- Researchers from Microsoft and Zhejiang University propose FastSpeech.
  - Fast generation speed
  - Robustness
  - Controllability (?)
  - High quality (similar to autoregressive models, such as Tacotron 2 and Transformer TTS)

Text: Bad speech synthesis

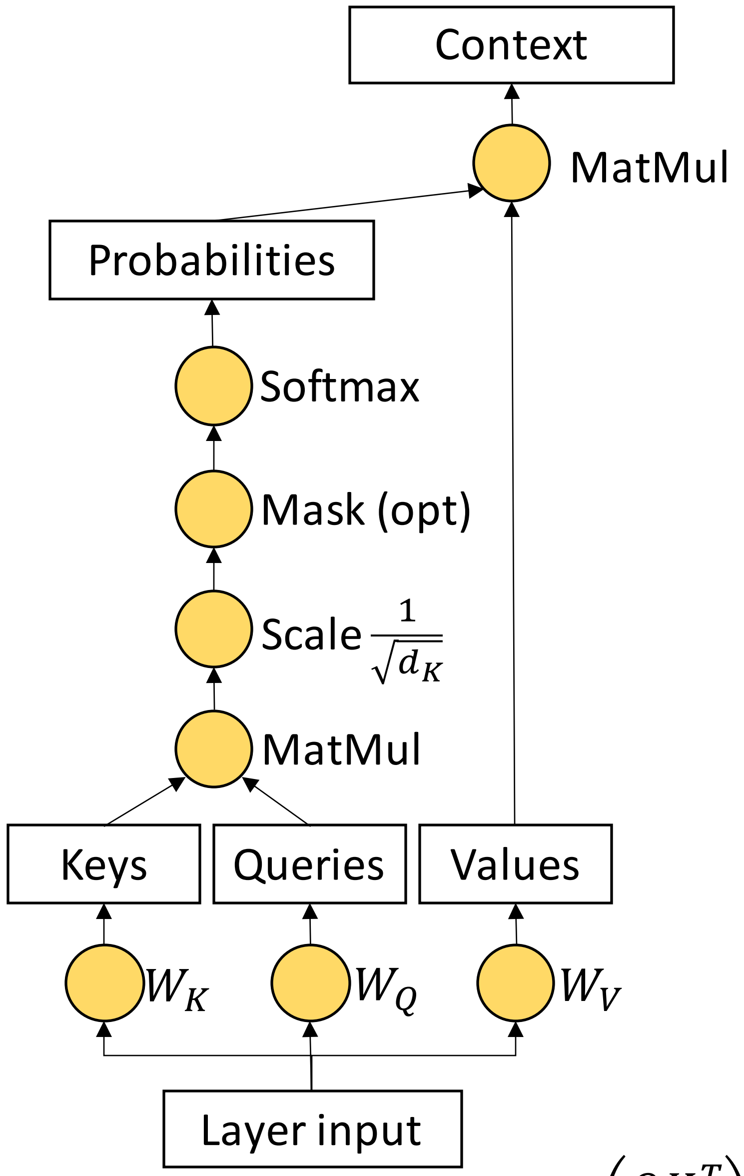
Created with TensorFlowTTS
with MelGAN vocoder

FastSpeech: Feed-Forward Transformer

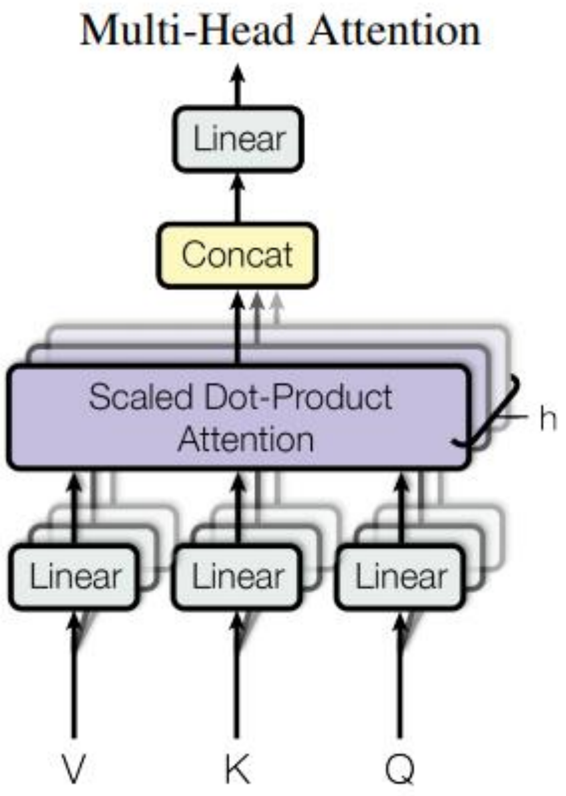


(a) Feed-Forward Transformer (b) FFT Block (c) Length Regulator (d) Duration Predictor

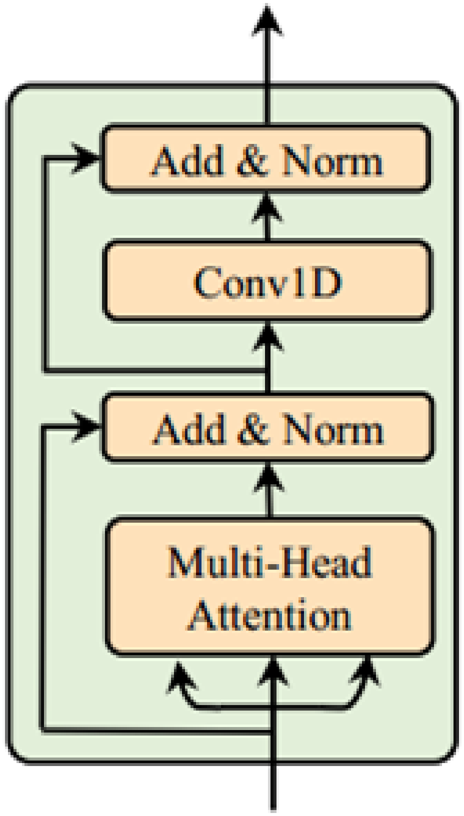
Scaled Dot-Product Multi-Head Attention



$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

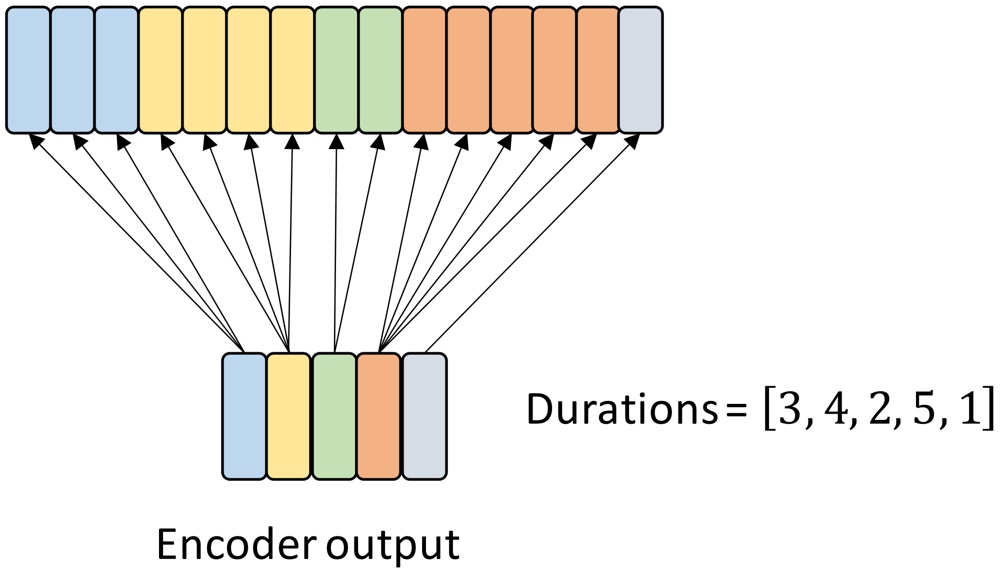


Attention heads=2

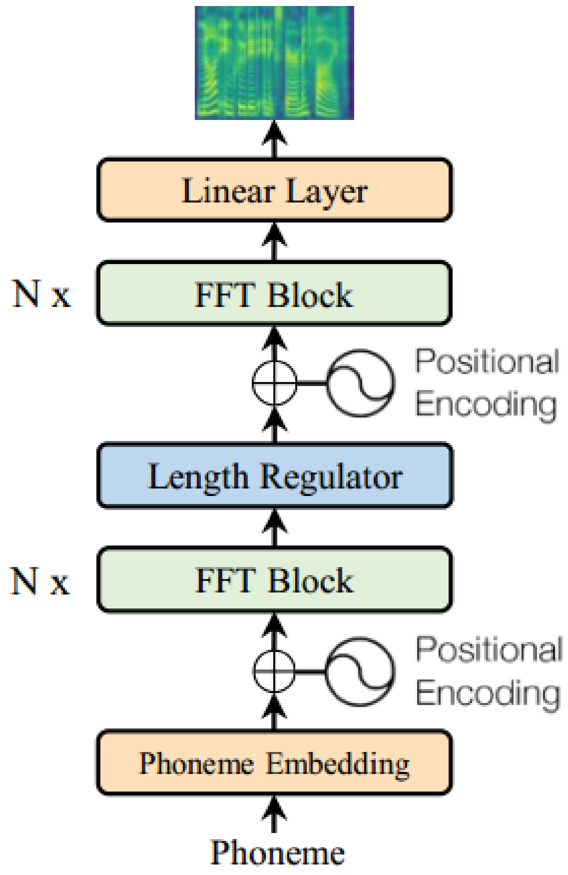


Length Regulator

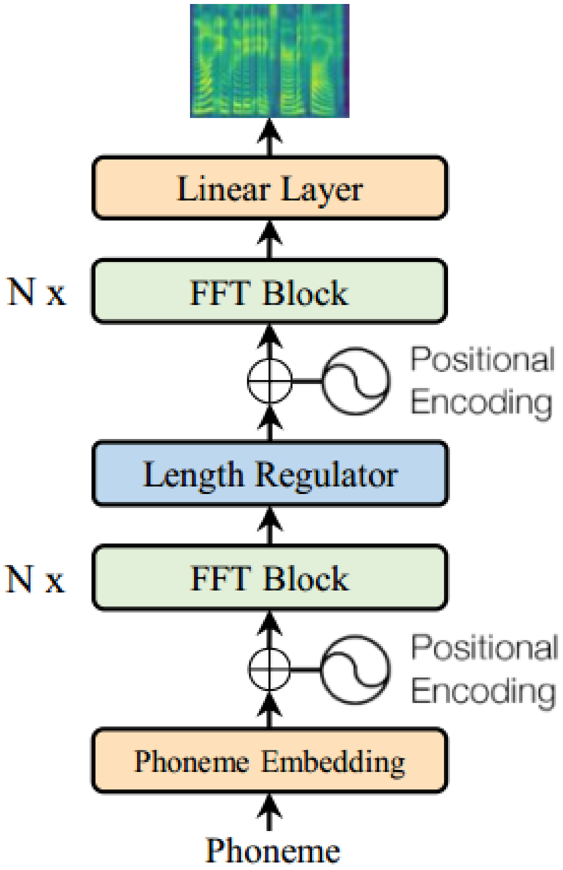
- Since the length of the phoneme sequence is smaller than that of the mel-spectrogram sequence, one phoneme corresponds to several mel-spectrograms.
- The number of mel-spectrograms that aligns to a phoneme is called phoneme duration.
- The length regulator expands the hidden sequence of phonemes according to the duration in order to match the length of a mel-spectrogram sequence.

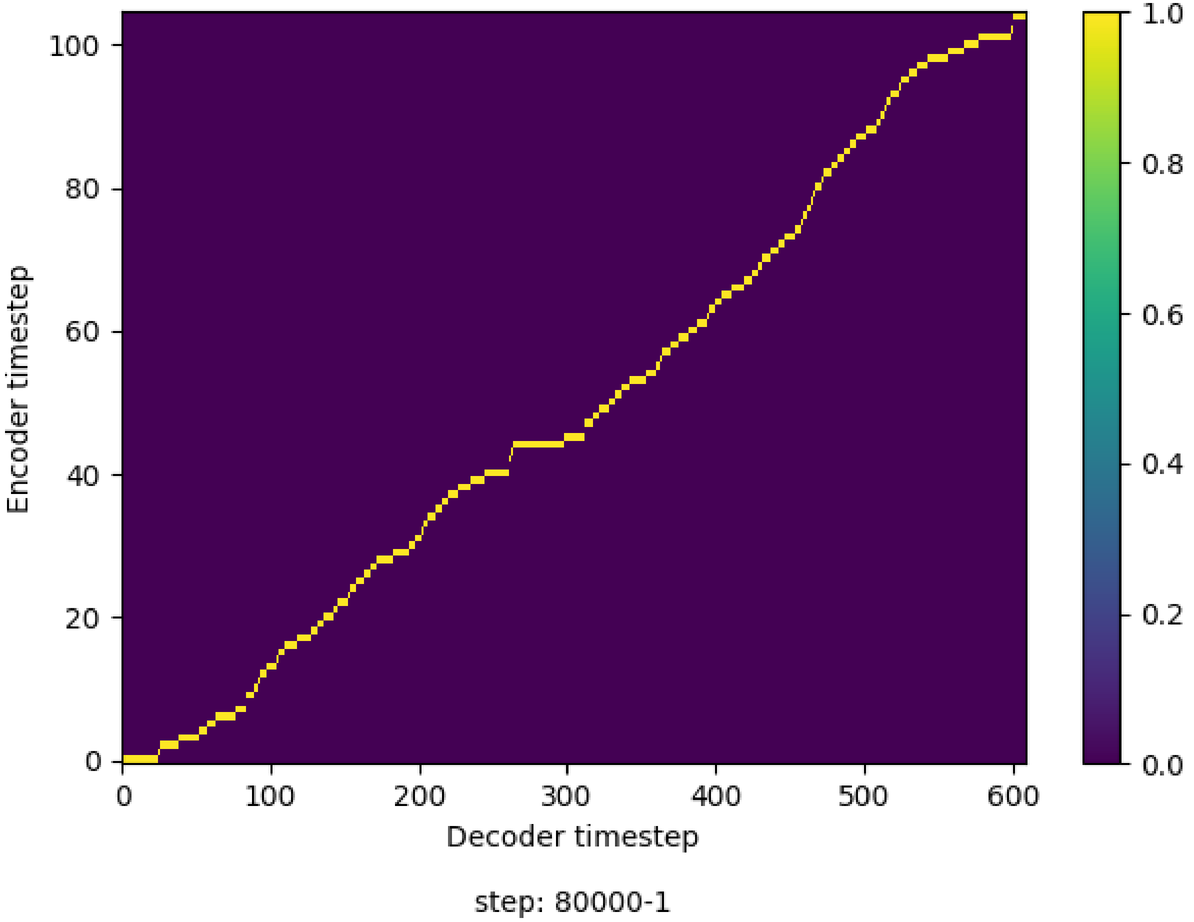


FastSpeech: Feed-Forward Transformer



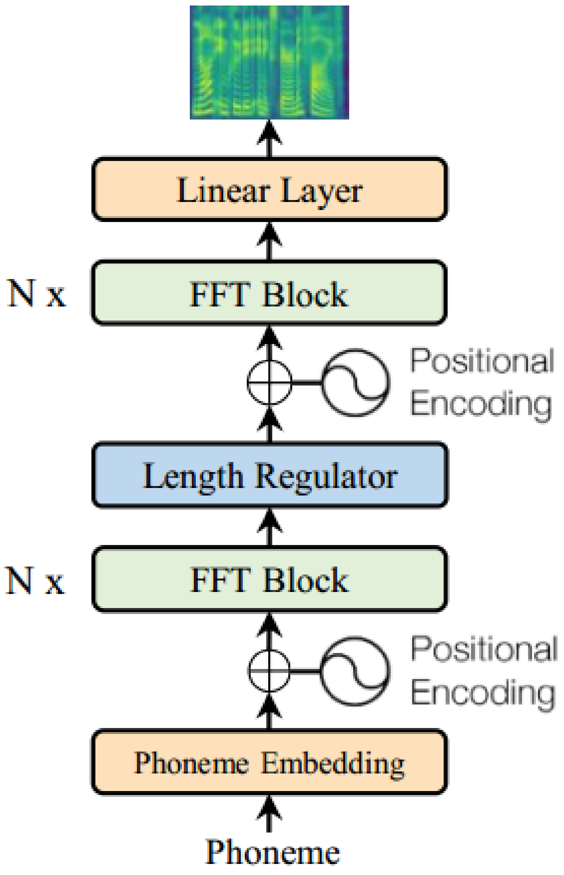
FastSpeech: Feed-Forward Transformer

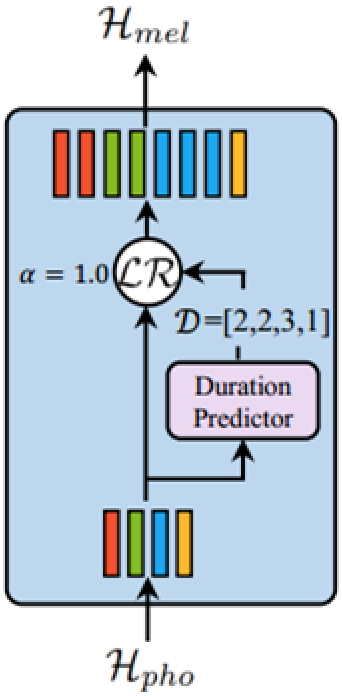




Loss functions

- Mel-loss: Usually the mean-absolute or the mean-square error between the ground-truth mels and the predicted-mels.
- Duration-loss: Mean square error between the ground-truth log-duration and the predicted log-duration.

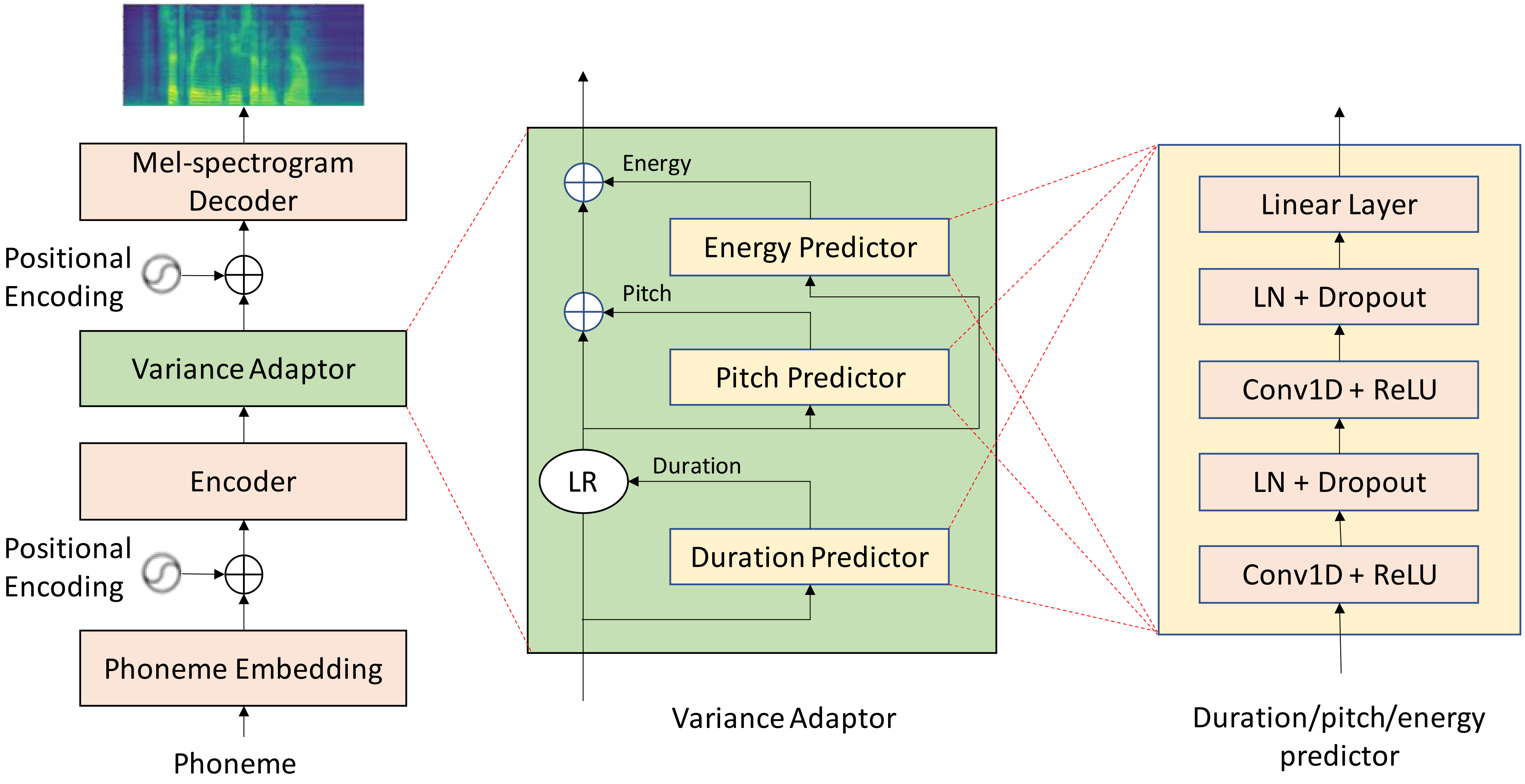




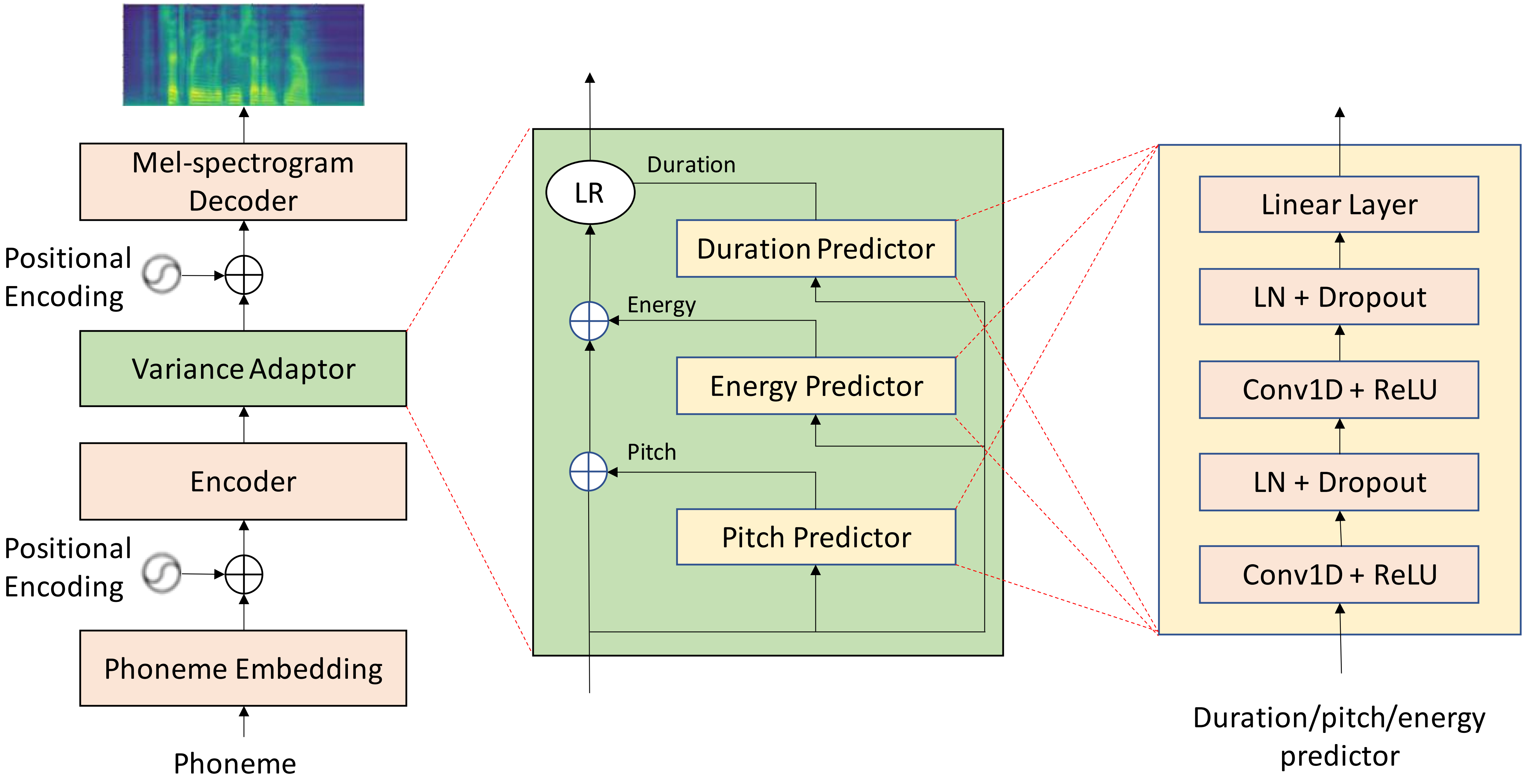
FastSpeech 2

- FastSpeech can generate mel-spectrograms with improved robustness and controllability, and can achieve comparable voice quality with previous autoregressive models.
- However, there are still some disadvantages to it:
  1. The two-stage teacher-student distillation pipeline is complicated;
  2. The duration extracted from the attention map of the teacher model is not accurate enough, and the target mel-spectrograms distilled from the teacher model suffer from information loss due to data simplification, both of which limit the voice quality and prosody.
- FastSpeech 2 addresses these issues.
  1. FastSpeech 2 is trained with ground-truth mel targets instead of the simplified output from a teacher.
  2. It uses the Montreal forced alignment tool to extract the phoneme duration.
  3. To reduce the information gap between the input (text sequence) and target output (mel-spectrograms), the input includes pitch, energy, and more accurate duration than the duration of FastSpeech.
     - during training, the duration, pitch, and energy are extracted from the target speech waveform.
     - during inference, these values are predicted by the predictors that were jointly trained with the FastSpeech 2 model.

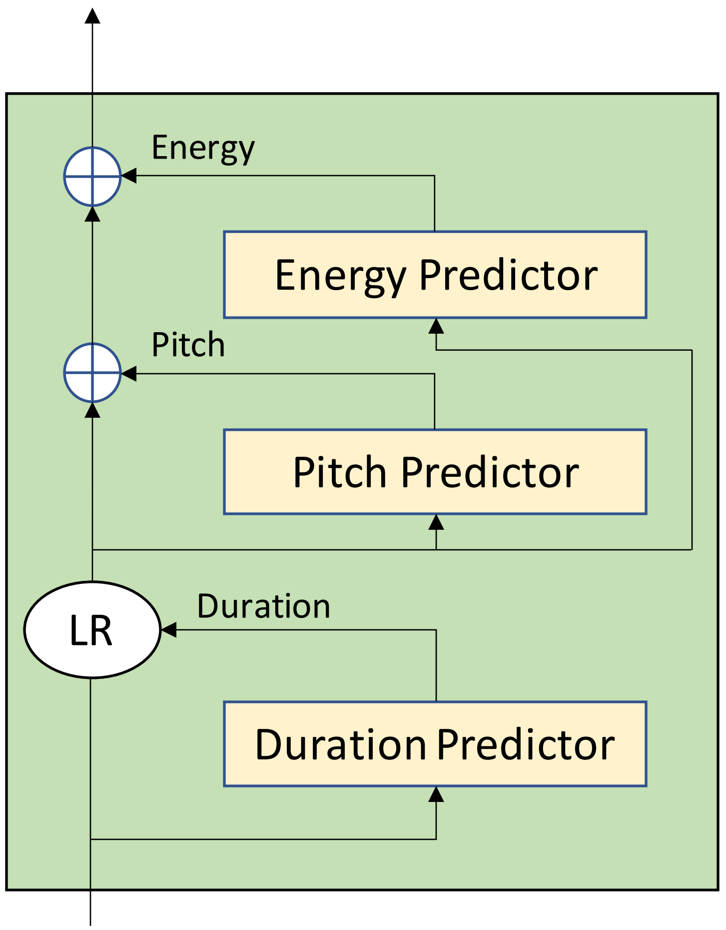
FastSpeech 2



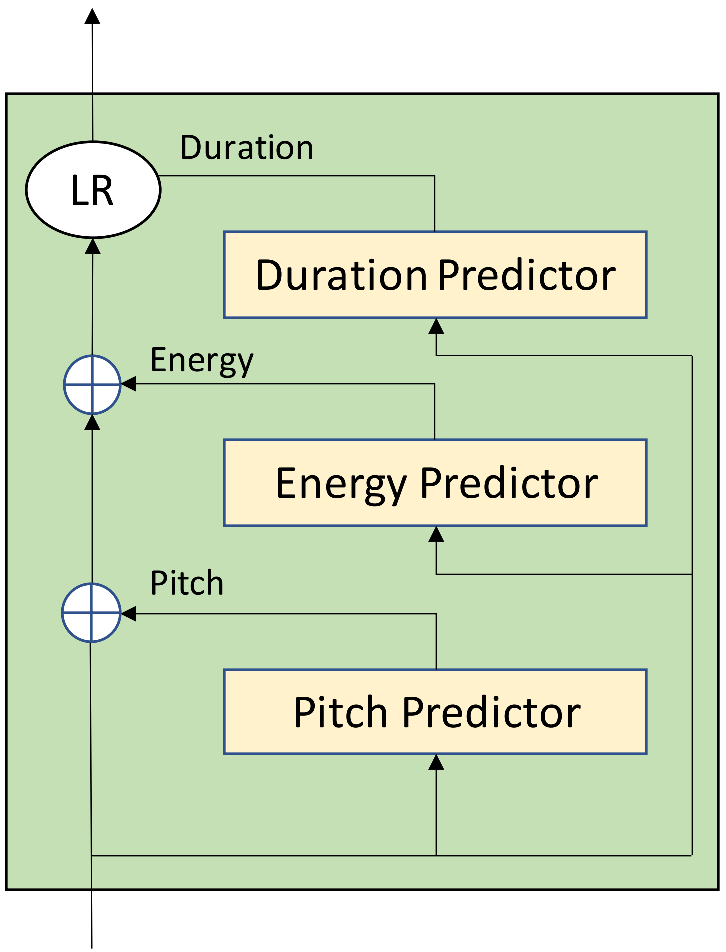
FastSpeech 2



Variance Adaptor



Frame level energy and pitch prediction



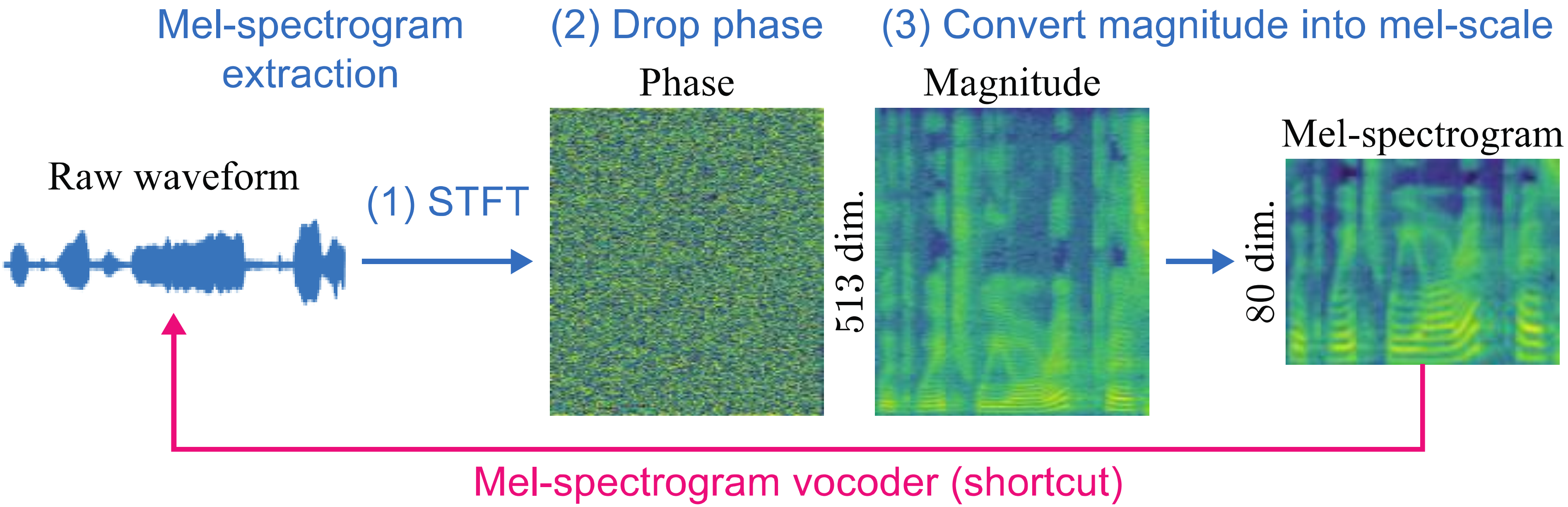
Phoneme level energy and pitch prediction

Phoneme level energy and pitch prediction instead of frame level results in better prosody

Outline

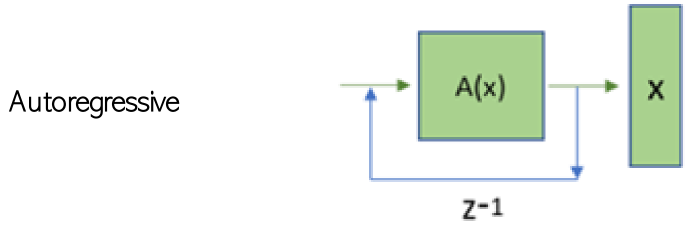
- Introduction
- Two stage pipeline systems
  - Acoustic models
    - Tacotron 2
    - Transformer TTS
    - FastSpeech 1 and 2
  - Neural vocoders
    - Sequential generation
      - WaveNet
      - WaveRNN
    - Parallel generation
      - Artifacts
- Do end-to-end systems reduce the artifacts in parallel generation?

Mel-spectrogram vocoders



Neural Vocoders

- Sequential generation of samples
  - Autoregressive



- Parallel generation of samples
  - GAN-based
  - VAE-based
  - Flow-based
  - Diffusion-based

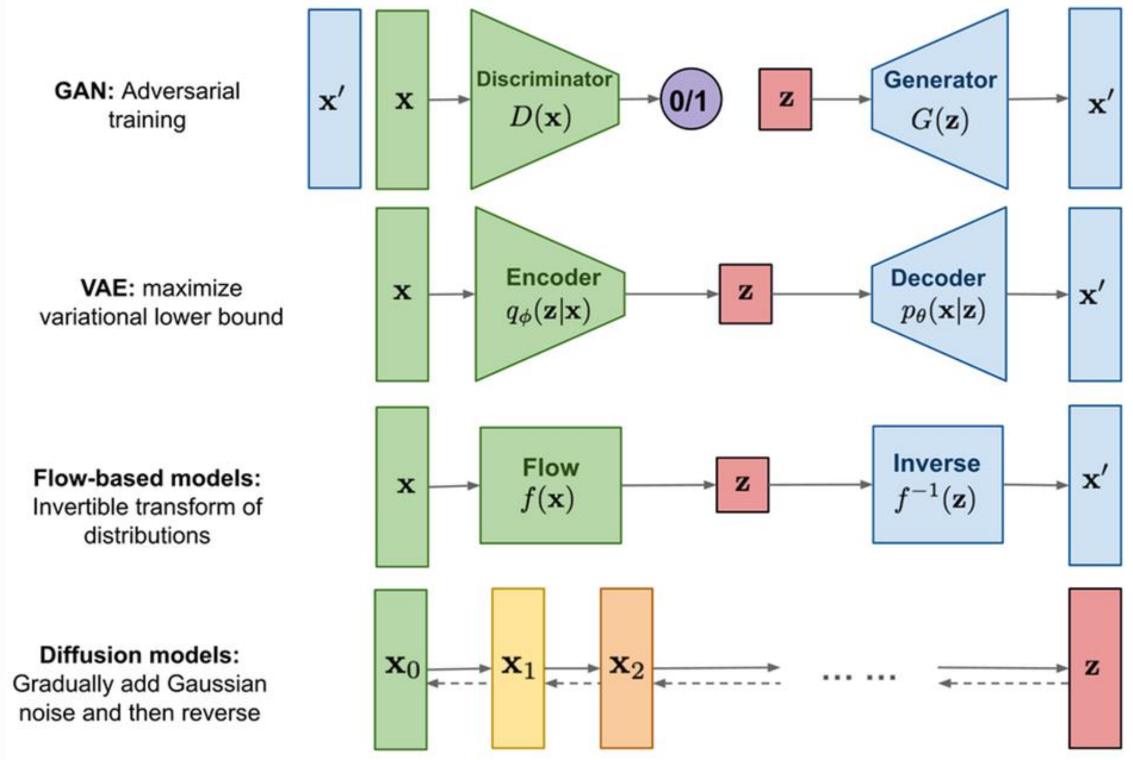
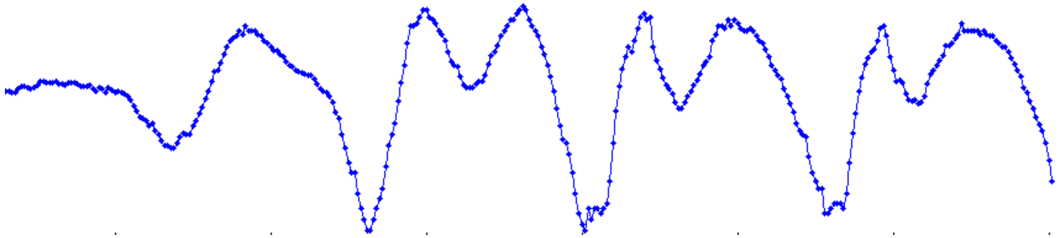


Fig. 1. Overview of different types of generative models.

Autoregressive Neural Vocoders

- Speech: x_0 x_1 x_2 x_3 x_4 ••• x_{T-1}



- The chain rule of probability

$$P(x_0 x_1 x_2 x_3 \ldots x_{T-1}) = P(x_0)\prod_{t=1}^{T-1} P(x_t | x_0 \ldots x_{t-1}) = P(x_0)\prod_{t=1}^{T-1} P(x_t | x_{<t})$$

- Autoregressive neural vocoders model the conditional probability

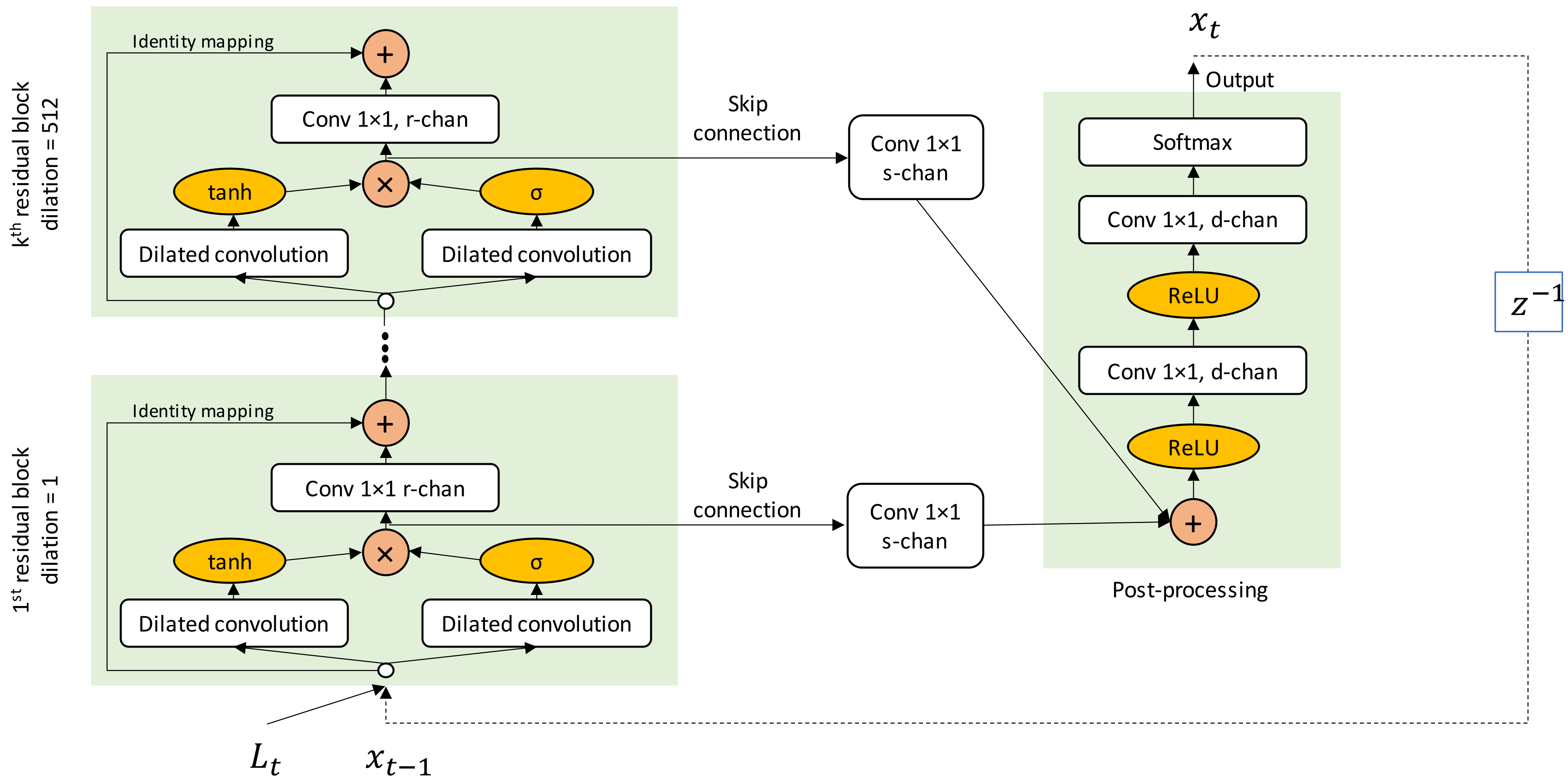
$P(x_t | x_{<t}) \longrightarrow$ Synthesis from this distribution

- And in order to generate speech the conditions include linguistic or acoustic information

$P(x_t | x_{<t}, L_t) \longrightarrow$ Synthesis from this distribution

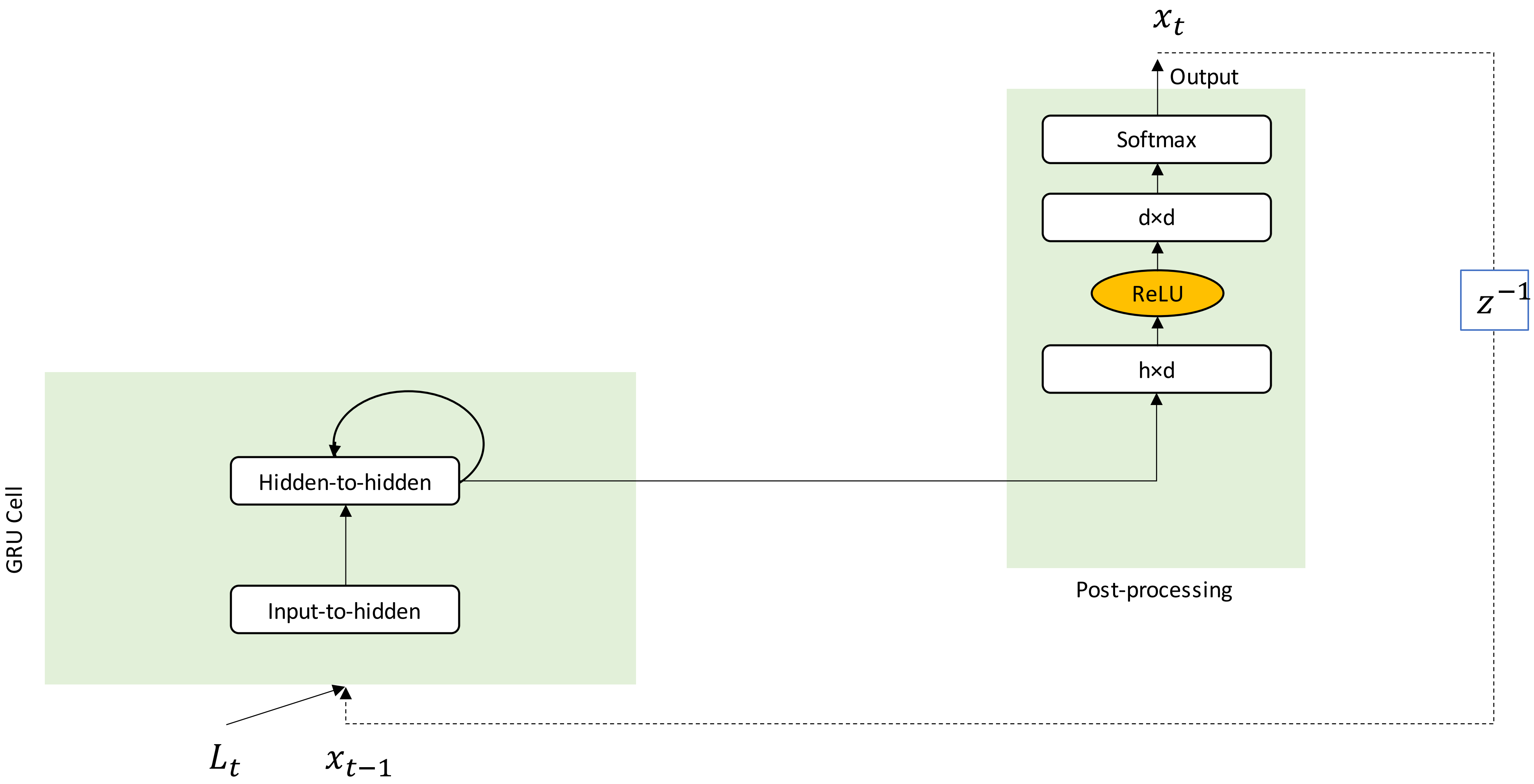
WaveNet

- WaveNets is an autoregressive model, which achieve state-of the art results in audio synthesis.



WaveRNN

- WaveNets achieves state-of the art results in audio synthesis.
- However, sampling from Wavenet is sequential and impractical.
- To increase the efficiency of sampling from these models, Kalchbrenner et al., proposed to substitute the layers of dilated convolutions of WaveNet with a single GRU layer



WaveRNN output

- WaveRNN assigns to an input vector x_t a probability distribution using the softmax function.

$$h(z)_j = \frac{e^{z_j}}{\sum_{c=1}^{256} e^{z_c}}, \quad j = 1, \ldots, 256$$

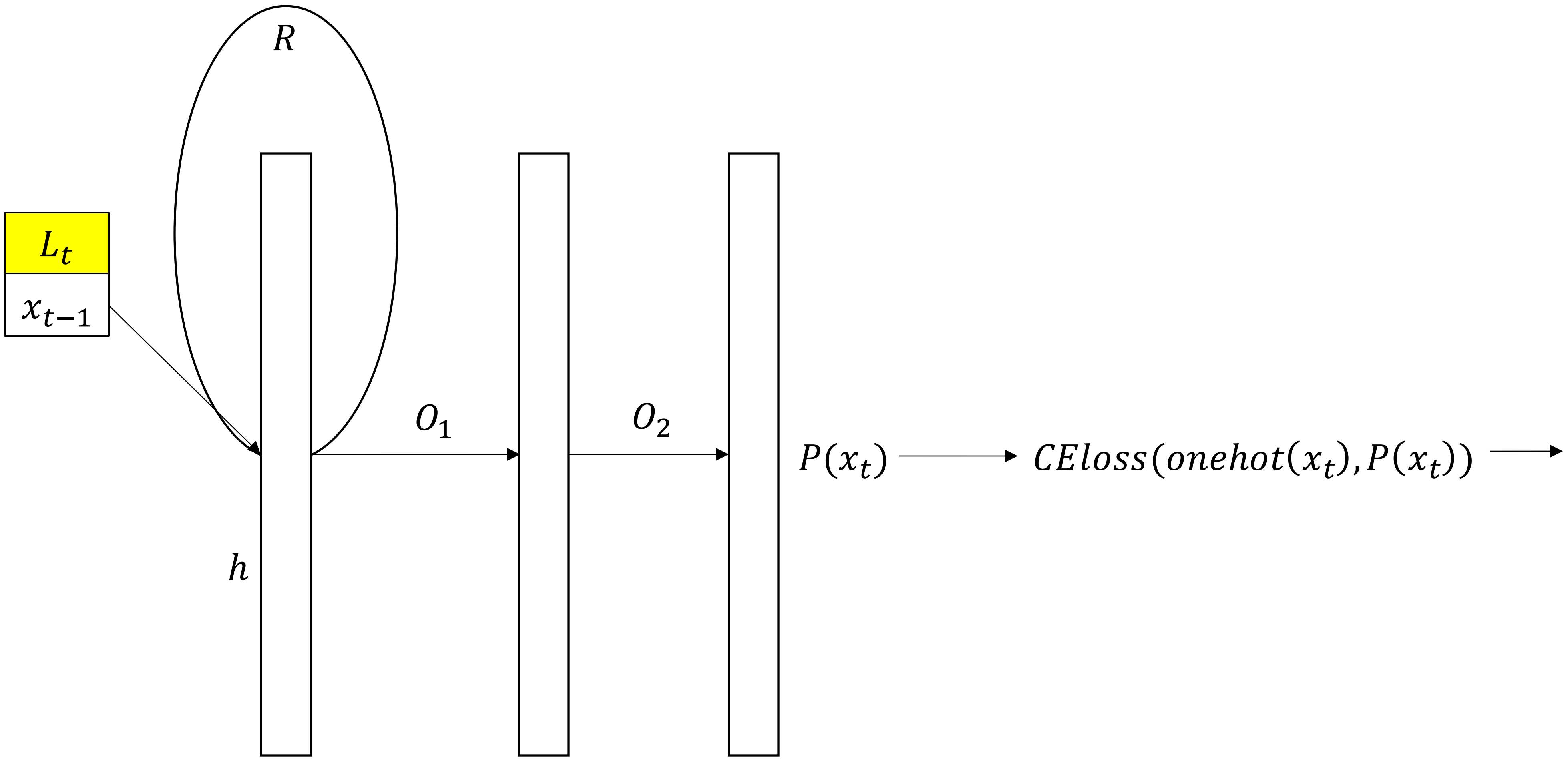
Channels

.6	.2	.1	.1	0
.2	.5	.1	.6	.1
.1	.2	.7	.2	.1
.1	.1	.1	.1	.8

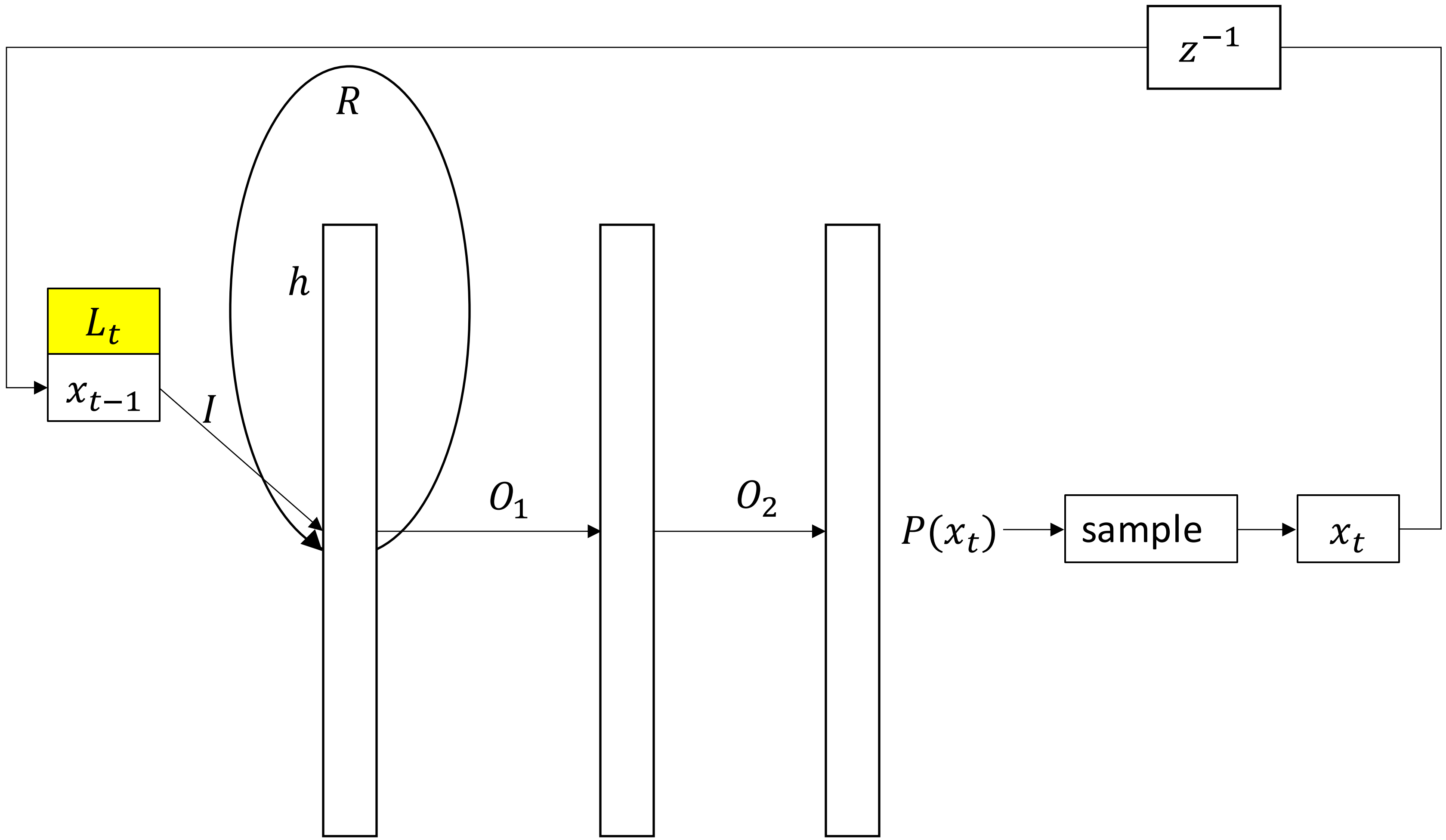
time

WaveRNN output: probabilities from softmax

WaveRNN training



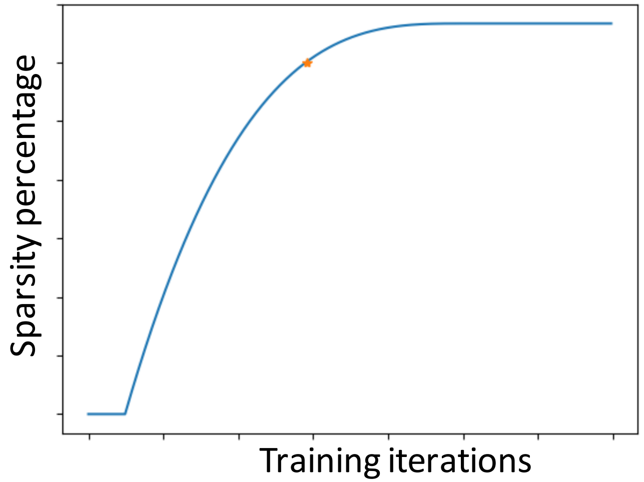
WaveRNN generation

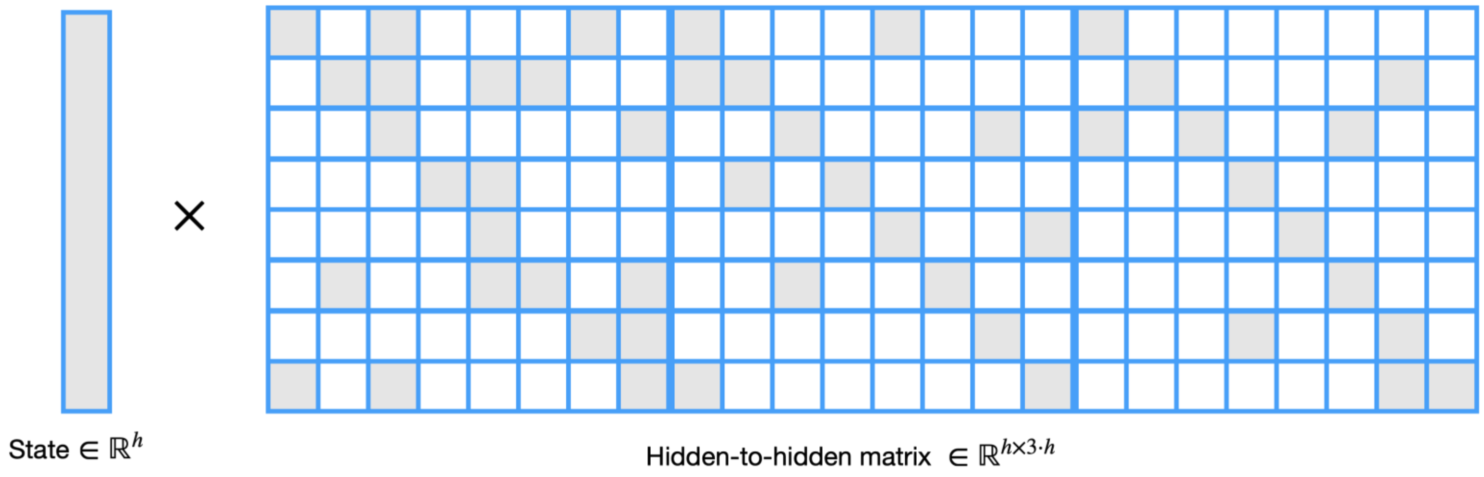


WaveRNN - Samples

Efficient WaveRNN

- Weight pruning
  - Progressively removes the weakest connections.
    1. Randomly initialize a neural network.
    2. Train the network until it converges.
    3. Prune a fraction of the network (the weights with the smallest absolute values).
    4. Repeat steps 2 and 3
  - Use of block or other structured sparsity for efficiency.

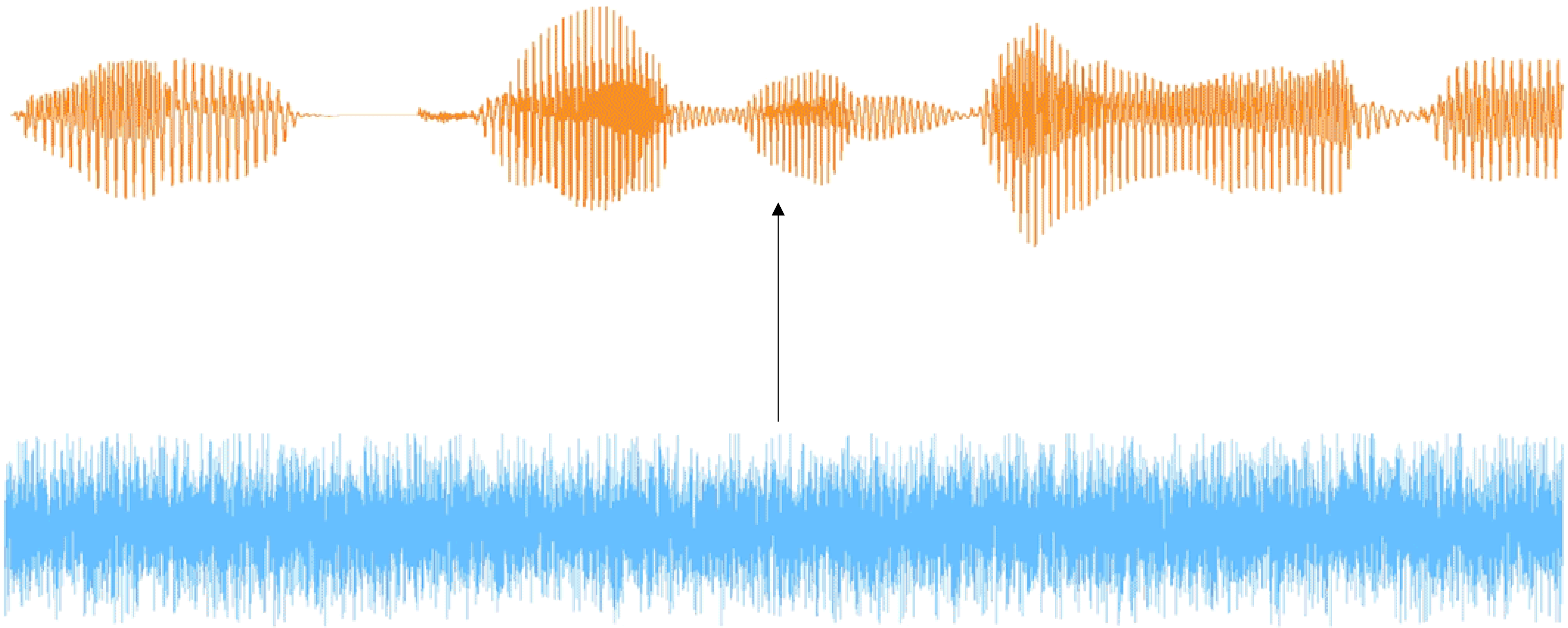




Parallel Vocoders

- During synthesis , parallel vocoders convert noise and conditioning acoustic features to speech.

Speech $x = [x_0, \dots x_{T-1}] \in X$ follows a probability distribution $p_X(x)$



Sample $z = [z_1, \dots, z_T] \in Z$ follows a simple probability distribution $p_Z(z)$.
E.g., samples z_t are independent of each other and they are drawn from a Logistic or a Gaussian distribution.

Samples from Parallel WaveNet

awb_a0001
bdl_a0001
clb_a0001
jmk_a0001
ksp_a0001
rms_a0001
slt_a0001

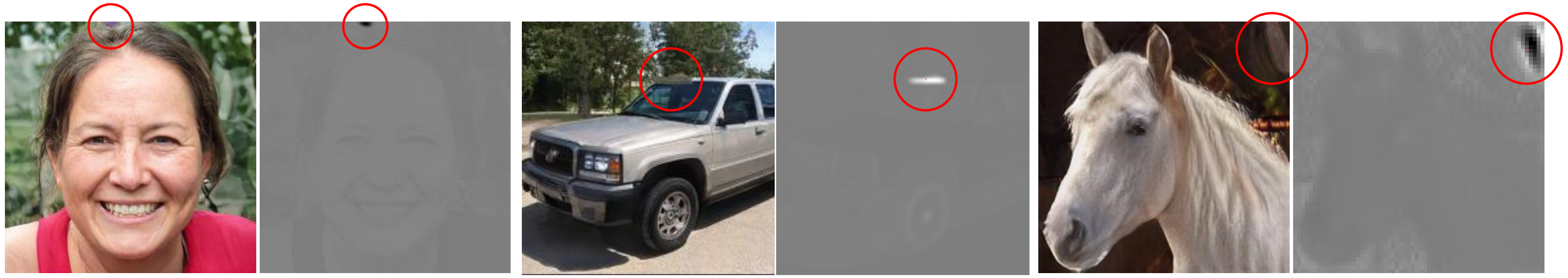
Kullback-Leibler 1/B*powerLoss

awb_a0001
bdl_a0001
clb_a0001
jmk_a0001
ksp_a0001
rms_a0001
slt_a0001

Kullback-Leibler 400/B*powerLoss

Known problems with two stage acoustic models

- Neural network-based TTS systems with two-stage pipelines, usually use mel-scale spectrograms as intermediate representation.
- The neural vocoder is trained separately using ground truth mel spectrograms.
- The distribution of Tacotron or FastSpeech mel-spectrograms differs from the distribution of the ground truth mel-spectrograms.
- This mismatch has no significant effect when the vocoder is autoregressive (WaveNet, WaveRNN).
- However, this mismatch triggers phase artifacts in parallel vocoders.
  - Tested with GAN, flow and diffusion vocoders (MelGAN, HifiGAN, DiffWave, WaveGrad, WaveGlow, WaveFlow).
  - Similar problem in image generation



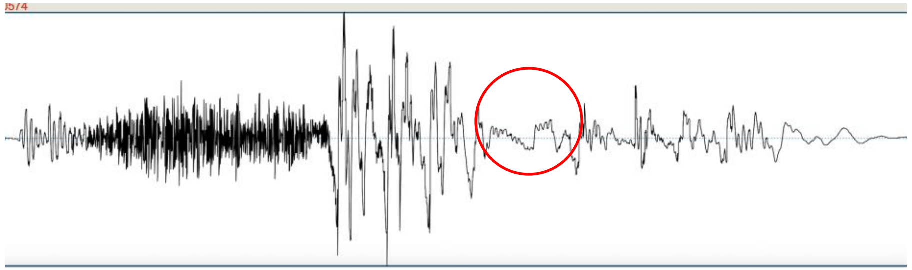
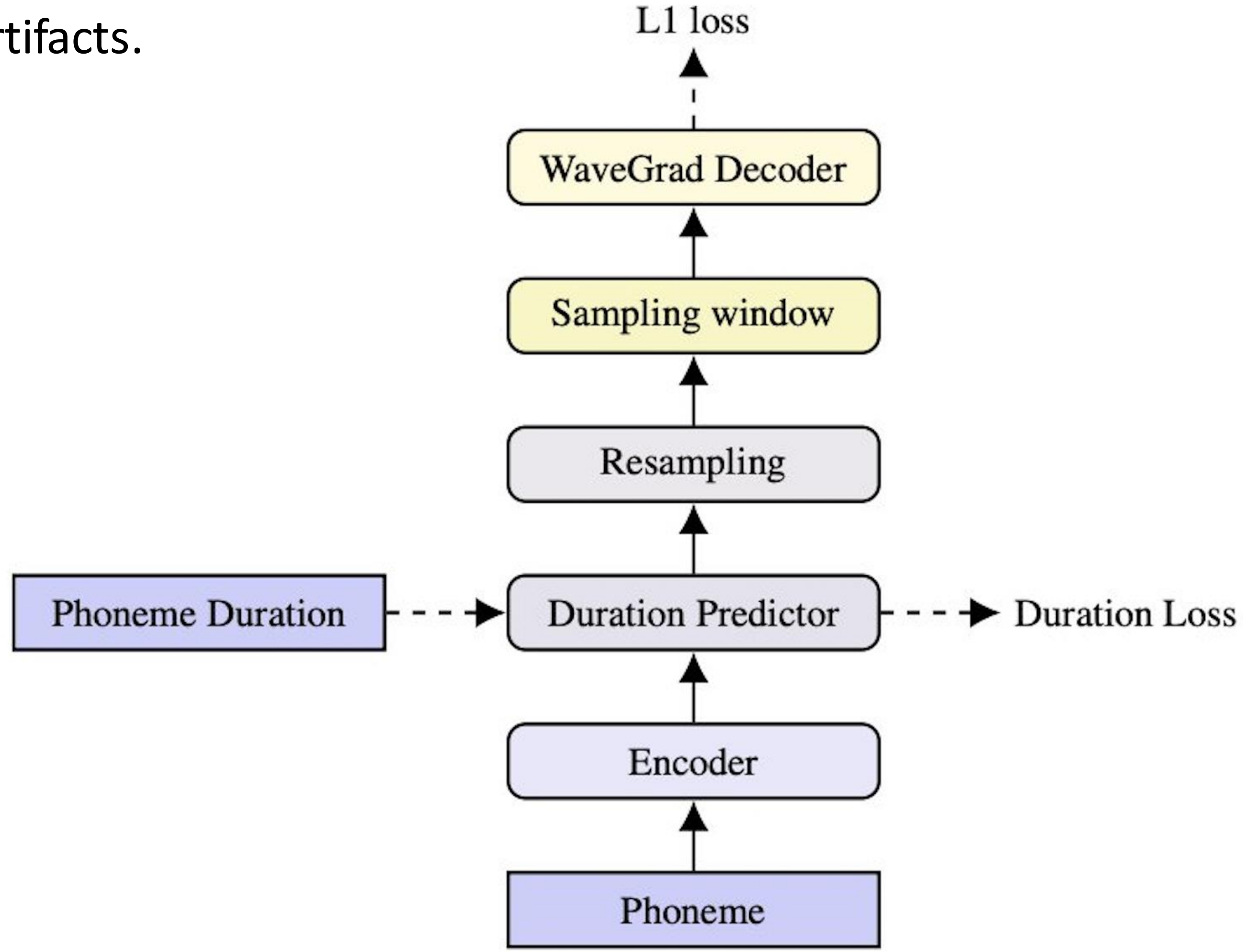


Image taken from Karras et al., Analyzing and Improving the Image Quality of StyleGAN

- Solutions:
  a) Use end-to-end models
  b) Use more robust intermediate representations

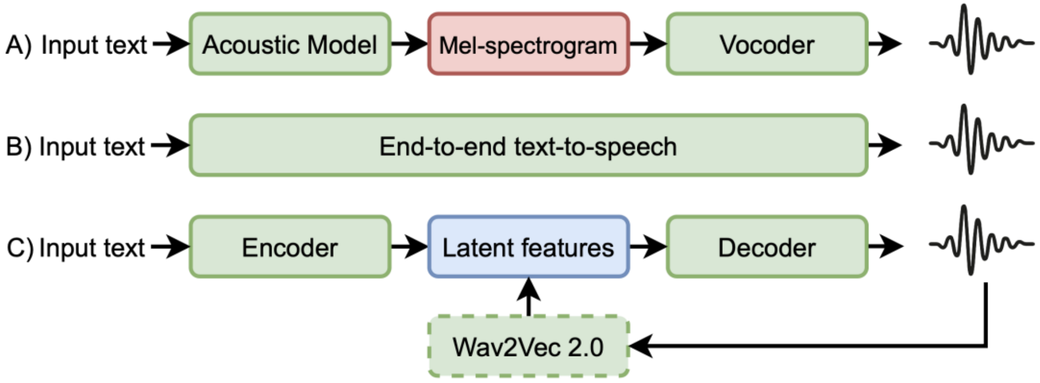
End-to-end systems

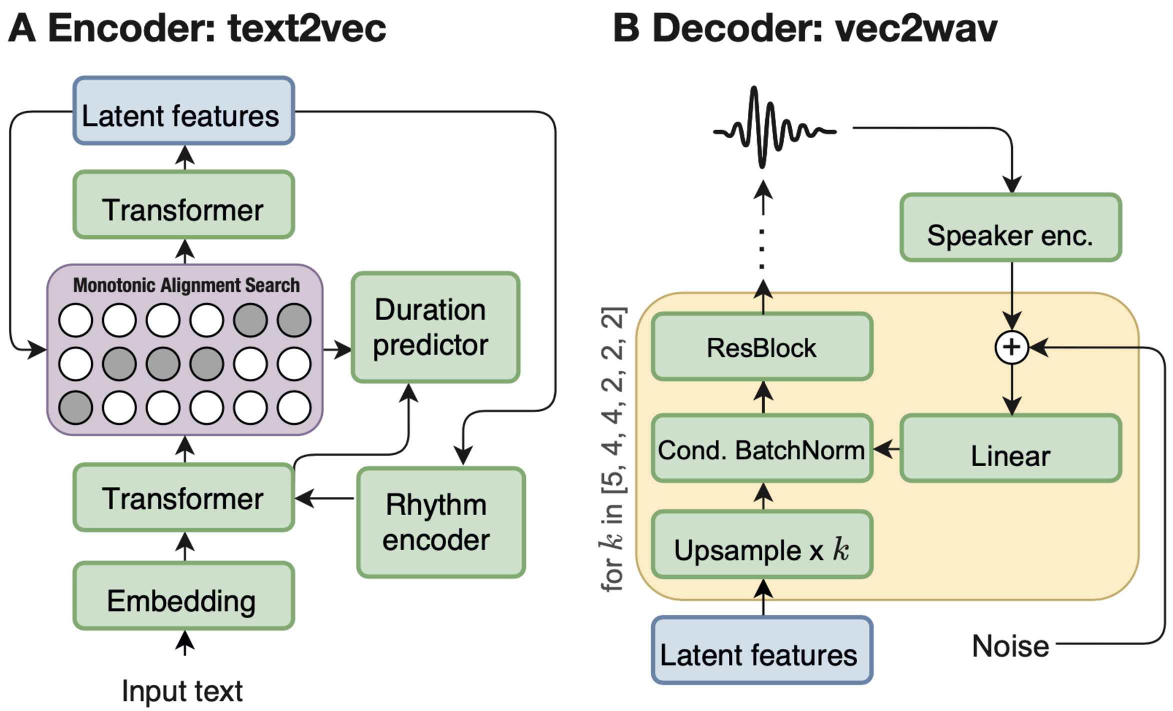
- WaveGrad 2 is a non-autoregressive generative model for text-to-speech synthesis.
- The model takes an input phoneme sequence, and through an iterative refinement process, generates an audio waveform.
- WaveGrad 2 significantly reduces the phase artifacts.



Using high level linguistic features

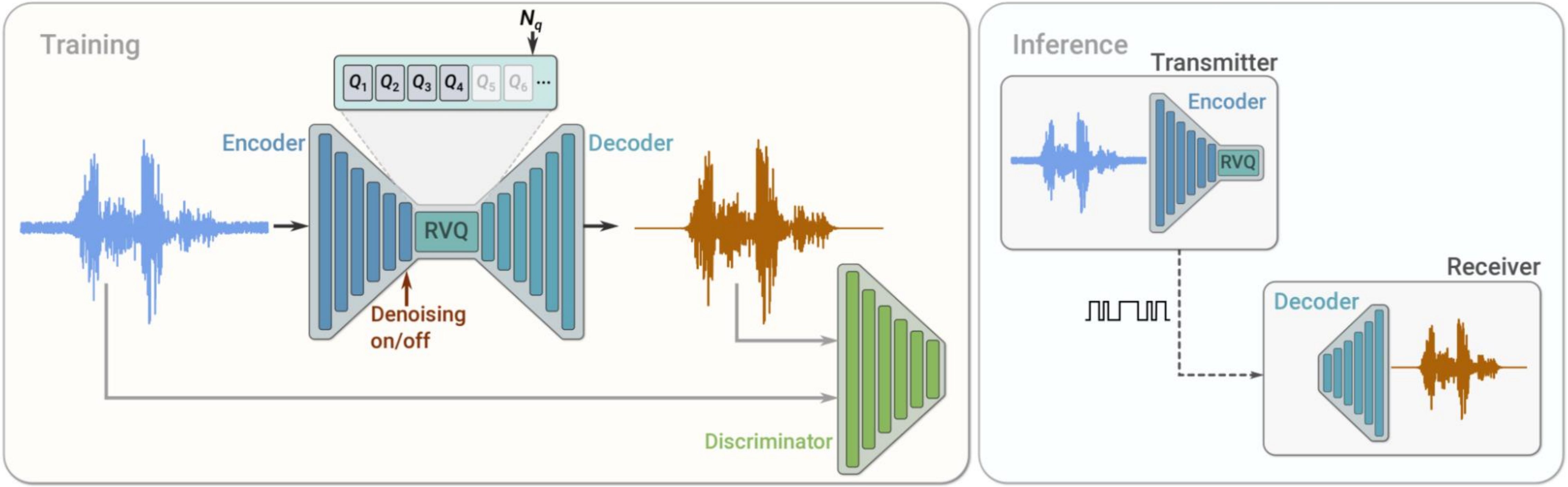
- WavThruVec is a two-stage architecture that uses high-dimensional WAV2VEC 2.0 embeddings as intermediate speech representation.
- Since these hidden activations provide high-level linguistic features, they are more robust to noise.
- Also, their distribution do not change from train to synthesis time.





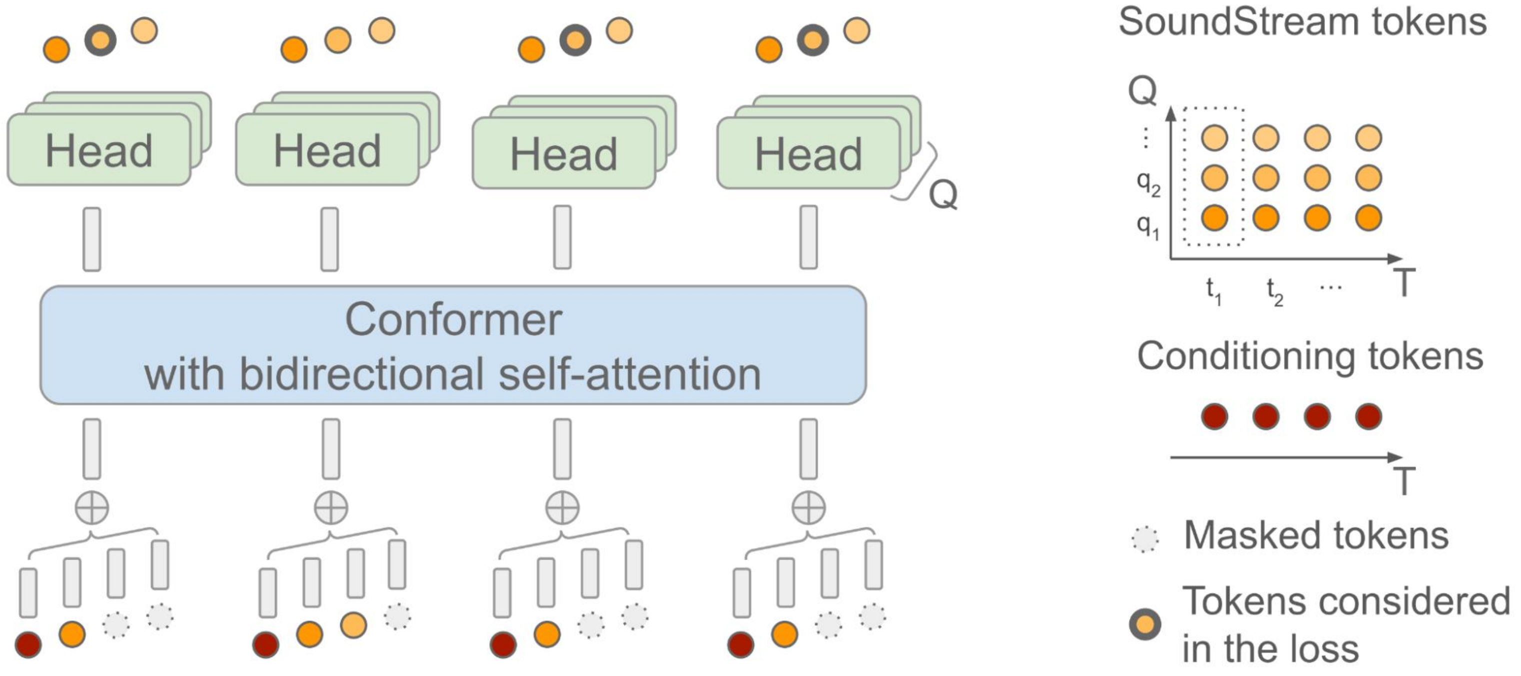
SoundStream: Neural audio codec

- During training, the encoder, quantizer and decoder parameters are optimized using a combination of reconstruction and adversarial losses, computed by a discriminator, which is trained to distinguish between the original input audio and the reconstructed audio.
- During inference, the encoder and quantizer on a transmitter client send the compressed bitstream to a receiver client that can then decode the audio signal.

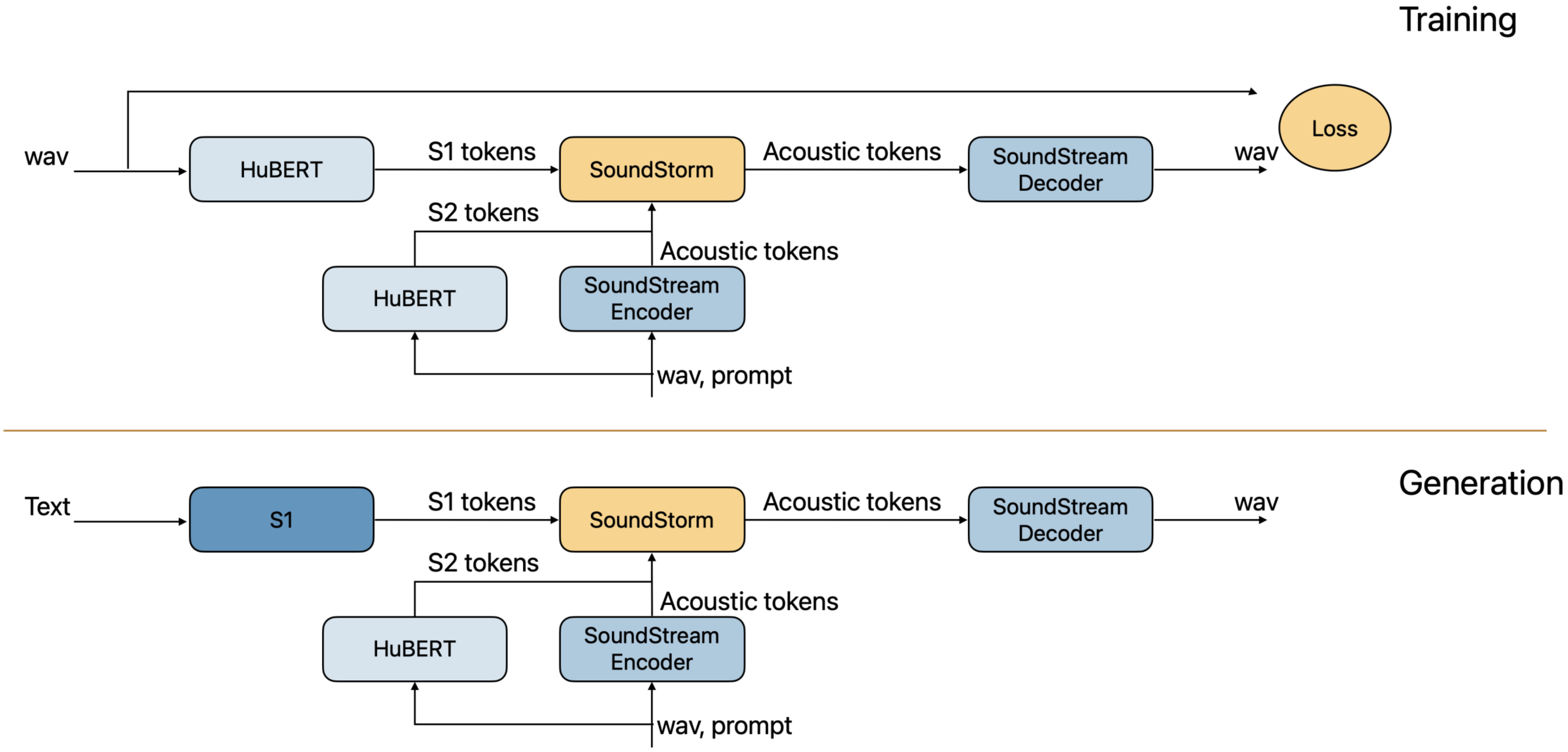


SoundStorm: Efficient parallel audio generation

- Resent generative models often rely on the fact that raw data is first converted to a compressed format as a sequence of tokens.
- In the case of audio, neural audio codecs (e.g., SoundStream) can efficiently compress waveforms to a compact representation, which can be inverted to reconstruct an approximation of the original audio signal.
- Such a representation consists of a sequence of discrete audio tokens, capturing the local properties of sounds (e.g., phonemes) and their temporal structure (e.g., prosody).



Language models for audio generation



References

1. van den Oord, Aaron; Dieleman, Sander; Zen, Heiga; Simonyan, Karen; Vinyals, Oriol; Graves, Alex; Kalchbrenner, Nal; Senior, Andrew; Kavukcuoglu, Koray, WaveNet: A Generative Model for Raw Audio, arXiv:1609.03499
2. Arik, Sercan; Chrzanowski, Mike; Coates, Adam; Diamos, Gregory; Gibiansky, Andrew; Kang, Yongguo; Li, Xian; Miller, John; Ng, Andrew; Raiman, Jonathan; Sengupta, Shubho; Shoeybi, Mohammad, Deep Voice: Real-time Neural Text-to-Speech, eprint arXiv:1702.07825
3. Arik, Sercan; Diamos, Gregory; Gibiansky, Andrew; Miller, John; Peng, Kainan; Ping, Wei; Raiman, Jonathan; Zhou, Yanqi, Deep Voice 2: Multi-Speaker Neural Text-to-Speech, eprint arXiv:1705.08947
4. Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan O. Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, John Miller, Deep Voice 3: Scaling Text-to-Speech with Convolutional Sequence Learning, ICLR 2018.
5. Äron van den Oord et al., Parallel WaveNet: Fast High-Fidelity Speech Synthesis, arXiv:1711.10433
6. Griffin D. and Lim J., "Signal Estimation from Modified Short-Time Fourier Transform". IEEE Transactions on Acoustics, Speech and Signal Processing. 32 (2): 236–243, 1984.
7. Wei Ping, Kainan Peng, Jitong Chen, ClariNet: Parallel Wave Generation in End-to-End Text-to-Speech, eprint arXiv:1807.07281, 2018.
8. Jose Sotelo, Soroush Mehri, Kundan Kumar, Joao Felipe Santos, Kyle Kastner, Aaron Courville, Yoshua Bengio, Char2Wav: End-to-End Speech Synthesis, ICLR workshop, 2017.
9. Soroush Mehri, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron Courville, Yoshua Bengio, SampleRNN: An Unconditional End-to-End Neural Audio Generation Model, arXiv:1612.07837v2, 2016.

References

10. Yuxuan Wang and RJ Skerry-Ryan and Daisy Stanton and Yonghui Wu and Ron J. Weiss and Navdeep Jaitly and Zongheng Yang and Ying Xiao and Zhifeng Chen and Samy Bengio and Quoc Le and Yannis Agiomyrgiannakis and Rob Clark and Rif A. Saurous, Tacotron: Towards End-to-End Speech Synthesis,, Interspeech 2017.
11. Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, RJ Skerry-Ryan, Rif A. Saurous, Yannis Agiomyrgiannakis, Yonghui Wu, Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions, ICASSP 2017.
12. Yaniv Taigman, Lior Wolf, Adam Polyak, Eliya Nachmani, VoiceLoop: Voice Fitting and Synthesis via a Phonological Loop, eprint arXiv:1707.06588, 2018.
13. Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, Yoshua Bengio, Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, eprint arXiv:1406.1078, 2018.
14. Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio, Neural Machine Translation by Jointly Learning to Align and Translate, arXiv:1409.0473v7, 2014.
15. Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, Yoshua Bengio, Attention-Based Models for Speech Recognition, arXiv:1506.07503v1, 2015.
16. Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, Tie-Yan Liu, FastSpeech: Fast, Robust and Controllable Text to Speech, arXiv:1905.09263, 2019
17. Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, Tie-Yan Liu, FastSpeech 2: Fast and High-Quality End-to-End Text to Speech, arXiv:2006.04558, 2020

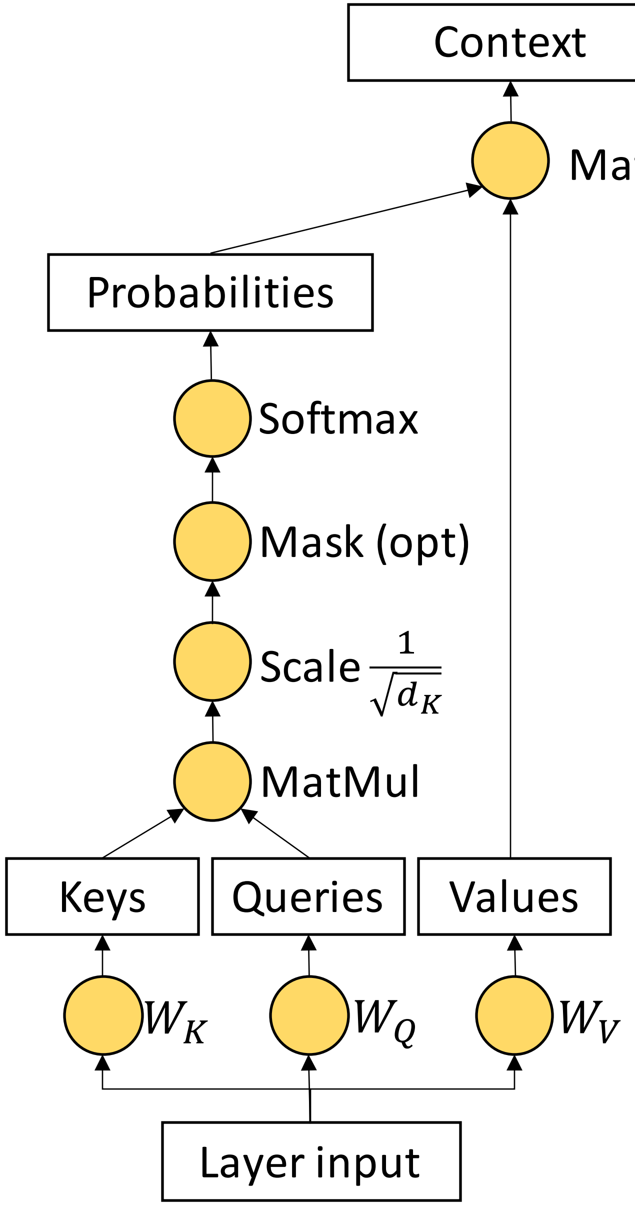
References

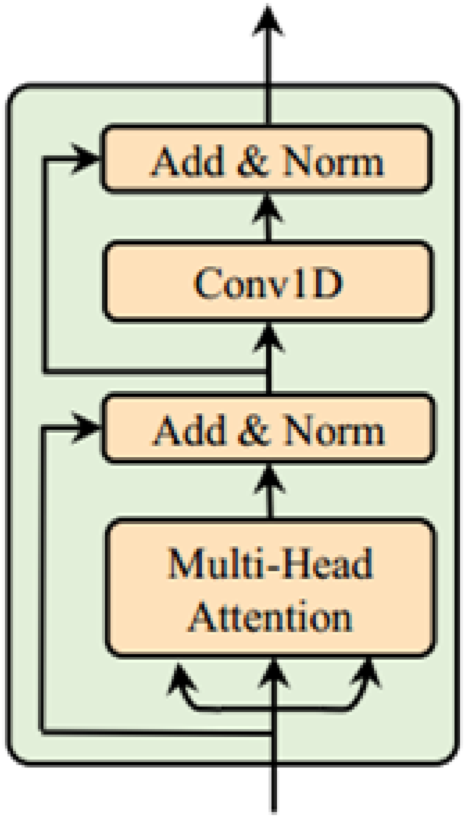
18. Nanxin Chen, Yu Zhang, Heiga Zen, Ron J. Weiss, Mohammad Norouzi, William Chan, WaveGrad: Estimating Gradients for Waveform Generation, arXiv:2009.00713, 2020
19. Nanxin Chen, Yu Zhang, Heiga Zen, Ron J. Weiss, Mohammad Norouzi, Najim Dehak, William Chan, WaveGrad 2: Iterative Refinement for Text-to-Speech Synthesis, arXiv:2106.09660, 2021
20. Hubert Siuzdak, Piotr Dura, Pol van Rijn, Nori Jacoby, WavThruVec: Latent speech representation as intermediate features for neural speech synthesis, arXiv:2203.16930, 2022
21. Kevin Shih, Rafael Valle, Rohan Badlani, Adrian Łacucki, Wei Ping, Bryan Catanzaro, RAD-TTS: Parallel Flow-Based TTS with Robust Alignment Learning and Diverse Synthesis
22. Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, Timo Aila, Analyzing and Improving the Image Quality of StyleGAN, arXiv:1912.04958, 2020
23. Xu Tan, Tao Qin, Frank Soong, Tie-Yan Liu, A Survey on Neural Speech Synthesis, arXiv:2106.15561, 2021
24. Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, Marco Tagliasacchi, SoundStream: An End-to-End Neural Audio Codec, 2021
25. Zalán Borsos, Matt Sharifi, Damien Vincent, Eugene Kharitonov, Neil Zeghidour, Marco Tagliasacchi, SoundStorm: Efficient Parallel Audio Generation, 2023

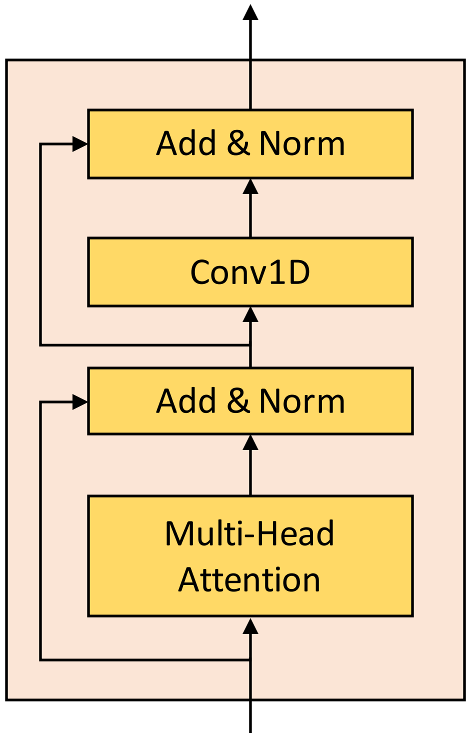
Appendix: The Griffin-Lim algorithm

- The Griffin and Lim's algorithm recovers an audio signal given only the magnitude of its Short-Time Fourier Transform (STFT), also known as the spectrogram.
- It is an iterative algorithm that attempts to find the signal having an STFT such that the magnitude part is as close as possible to the modified spectrogram.
- **Algotithm:**
- Input: s # spectrogram
- $x = random(n\_samples)$ # initialize the reconstructed audio
- for $i = 1: n\_iterations$
    - $c_1 = \mathrm{STFT}(x)$
    - $a = angle(c_1)$
    - $c_2 = s \cdot e^{ja}$
    - $x = \mathrm{ISFTF}(c_2)$ # reconstructed audio
- The Griffin-Lim algorithm converges after 30 to 50 iterations.
- The Griffin-Lim produces characteristic artifacts and lower audio quality than WaveNet.
- Nevertheless, the Griffin-Lim spectrogram inversion is efficient and allows back propagation of derivatives since it is differentiable.
- Therefore, it could be the initial choice when debugging a new end-to-end system.

Scaled Dot-Product Multi-Head Attention







$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$