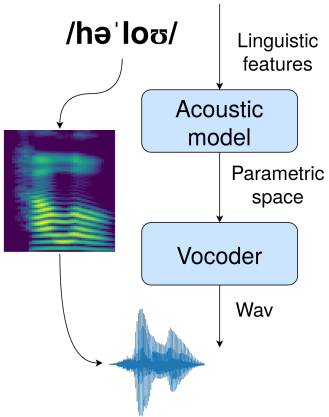
Yandex

SCHOOL OF DATA ANALYSIS

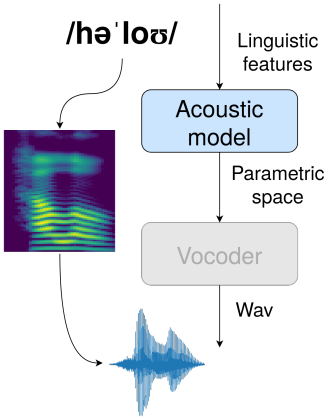
TTS: Acoustic Models

Vladimir Kirichenko

May 11th 2022



- Let's generate sound in two steps
- Firstly, generate a compact *parametrical representation* of the sound - e.g. mel-spectrogram - with *Acoustic Model*
- Secondly, convert this compact representation into a full-scale waveform with *Vocoder*



- Acoustic model (AM) is a component responsible for creating intermediate sound representation
- AM-s are responsible for high-level quality of sound:
  - intonation and expression
  - articulation
  - temp and phonemes duration

Alignment:

- How to get from text-level phonemes to time-level acoustic frames? (*phoneme* and *frame level*)
- How to control and vary speech speed?
- How to consider global and local text-to-sound dependencies?

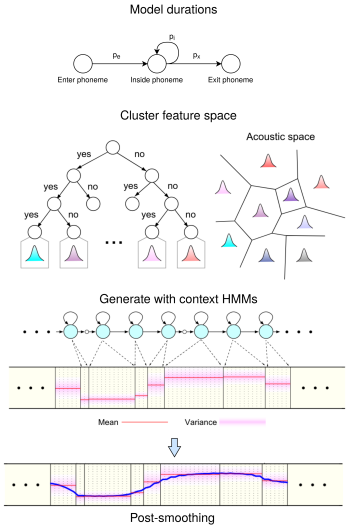
Continuity:

- How to convert discrete linguistic features into the continuous acoustic?
- How to prevent the acoustic features from being noisy?
- How to prevent the mode collapse?
- How to combine global and local factors in the acoustic features?

History of AMs

- HMMs
- Frame-wise FFNN-s
- RNNs
- RNN with frame- and phoneme-wise subnetworks
- (nowadays) Attention-based

HMMs



- Let's use HMM for generating quantized values of acoustic features
- We need a separate duration model to determine what size phonemes will be in parametric space
- With decision tree we can reduce input linguistic features variability
- To model acoustic features separately we need decorrelated parametric space (e.g. mfcc)
- We need a smoothing after HMM to produce continuous features

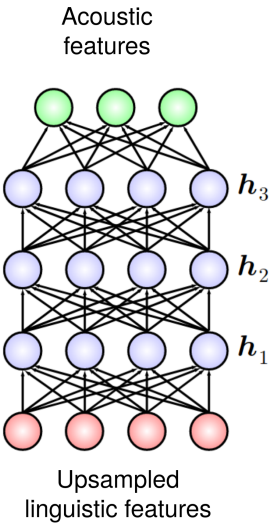
Pros:

- Fast
- Requires small amount of data to train (1-5 hours)

Cons:

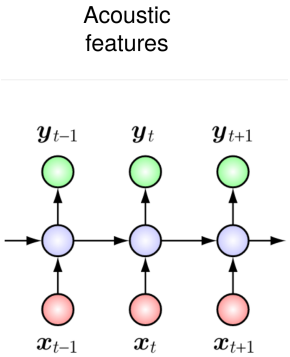
- Unable to handle complex features and internal dependencies in speech
- Requires additional clustering and duration prediction models
- Requires the parametric features to be decorrelated
- Requires post-smoothing

FFNNs



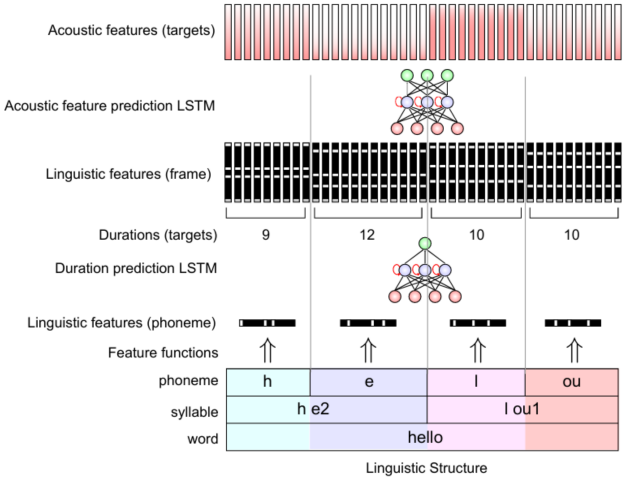
- Let's convert upsampled linguistic features (with some context) to an acoustic frame with DNN
- No need in clustering the linguistic features now
- We still need a duration model
- NN can generate smoother values at output
- Cannot model complex acoustic structures at acoustic feature space

RNNs



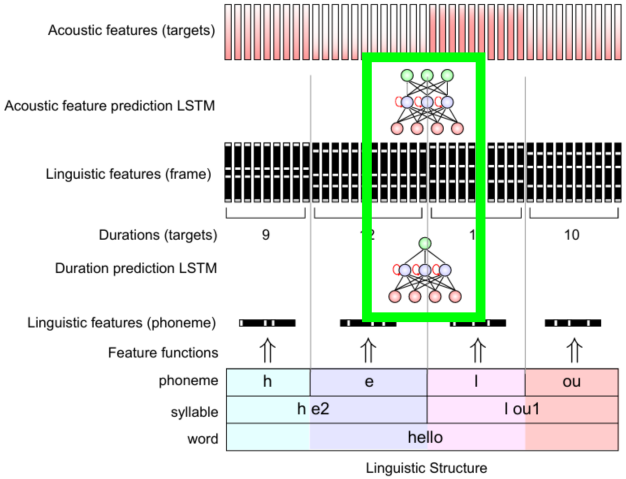
- To consider context we can replace FFNN with RNN (e.g. multi-layer LSTM)
- If we want streaming synthesis - we can use uni-directional RNNs
- Frame-level RNN can model complex acoustic structures (comparing to FFNN and HMM)

LSTM with Duration Model



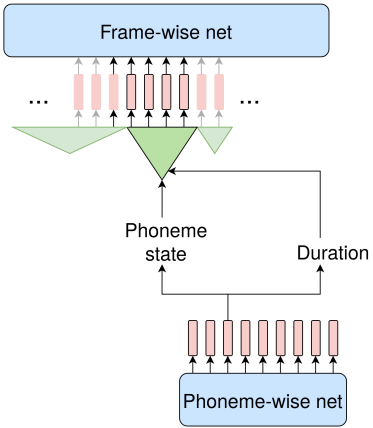
- LSTM RNN can be also utilized as duration predictor

LSTM with Duration Model



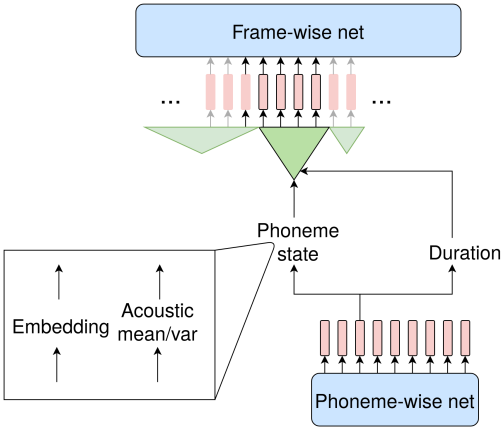
- LSTM RNN can be also utilized as duration predictor
- We can unite nets to get one whole AM network

Internal Upsampling



- Upsampling is differentiable
- So we can upsample output of LSTM phoneme-wise net and train all at once
- Loss will be combined: $\mathcal{L} = l_{acoustic} + l_{duration}$

Acoustic Features at Phoneme Level



1

- Also we can predict some phoneme-wise statistics of acoustic features to phoneme net outputs
- Predict mean/variance is easier than predict framewise values
- This information can helps the framewise net predict exact acoustic features

Pros:

- Quality of speech is acceptable for use in products (e.g. navigation, voice assistant, machine translation)
- Perf is better than FFNN (thick phoneme-wise part, thin frame-wise)
- Local features and details of speech are well-presented

Cons:

- LSTMs forget long context
- Poor at global speech structure
- Monotonic voice
- Need ground-truth durations (requires ASR or aligner)

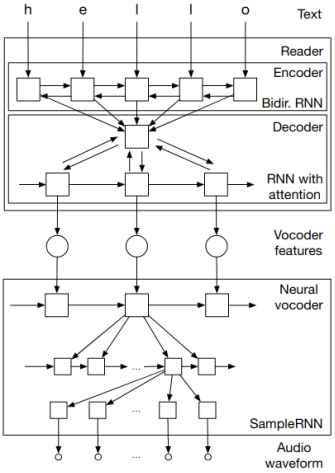
History of AMs

- HMMs required feature independence, smoothing afterward, did not consider complex speech structures
- FFNNs generated smooth speech, had problems with global intonation and non-trivial speech structures
- LSTM nets with phoneme-wise and frame-wise parts could handle complex local intonation and articulation details but failed to generate global expressive intonation contour

Attention-based Models

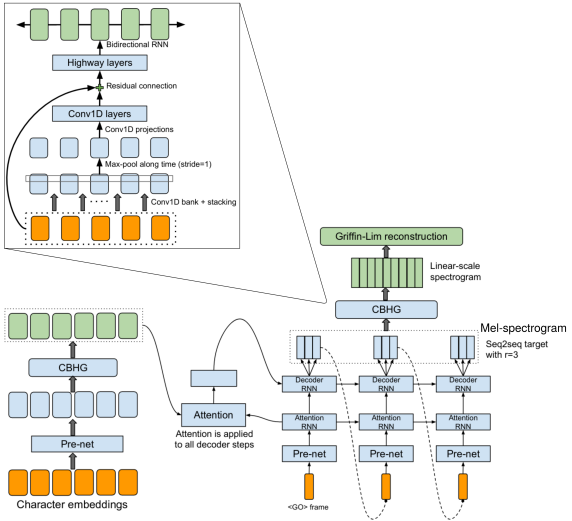
- LSTM-based models suffered from not-considering global intonation
- RNN is good for local features modeling, attention is good for both local and global ones
- Let's try seq2seq with attention for AMs!

Char2Wav



- LSTM bidirectional encoder and uni-directional decoder with attention
- Can work with phonemes or graphemes
- Autoregression
- Using neural vocoder SampleRNN

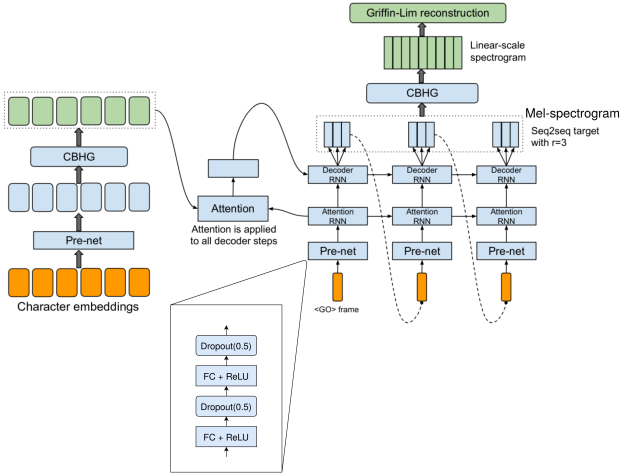
Tacotron



- Modified Machine Translation architecture
- CBHG blocks - 1d convolution + Highway + bi-directional GRU
- Makes mel-spectrogram and spectrogram (after CBHG Post-Net)
- Double loss: $\mathcal{L} = l_1(melspec) + l_1(spec)$
- Autoregression through Pre-Net block

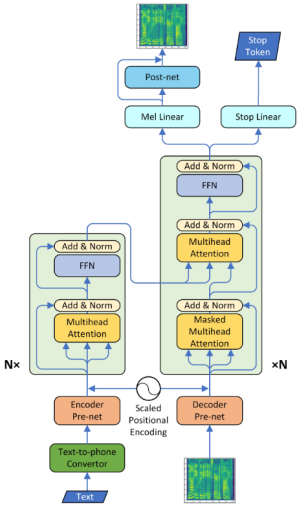
Limiting the Autoregression

Tacotron Pre-Net



- Melspectrogram slowly vary over time
- Need some regularization in AR loop
- Dropout in Tacotron Pre-Net works in inference

Transformer TTS

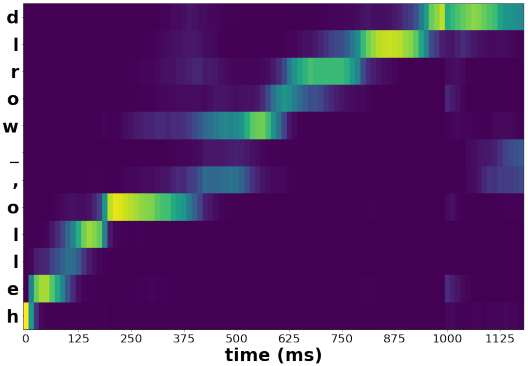


- Based on autoregressive transformer
- Pre-net and post-net
- Way more faster (than Tacotron) - x4-x5 speed up inference
- x10 or faster training
- Very unstable

Break!

10 min

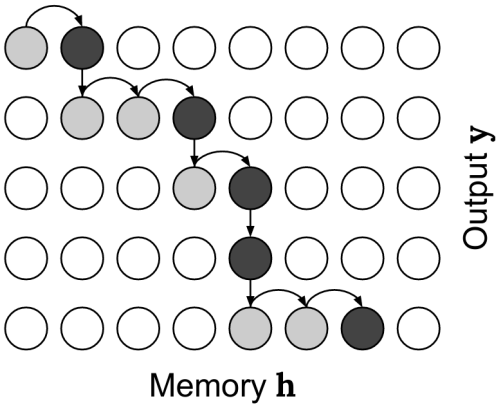
Attention in Synthesis



- Attention in TTS is very close to monotonic
- Modes come in ascending order
- General attention provides to much freedom and can fail at long and repeating sequences of words
- Such failures can cause strange sounds, moans, repeating and skipping words and phonemes
- We need some mechanism to prevent attention from falling apart

Monotonic Attention

Let's enforce monotonic attention at Inference!

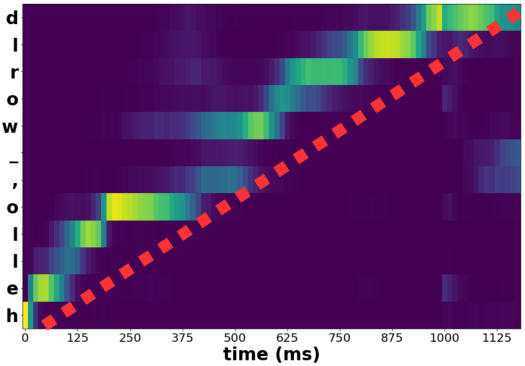


- Let s_i be decoder state, h_i - encoder state, $a(\cdot)$ - attention energy function

$$e_{i,j} = a(s_{i-1}, h_j)$$
$$p_{i,j} = \sigma(e_{i,j})$$
$$z_{i,j} \sim \text{Bernoulli}(p_{i,j})$$

- At each frame we continue sampling till $j^*: \; z_{i,j^*} = 1$
- At next frame $i+1$ we start from j^*
- Attention inference is linear and cannot go backward
- At the training phase we estimate $\alpha_{i,j} = \mathbb{E}(z_{i,j})$ over all possible j^*

Guided Attention

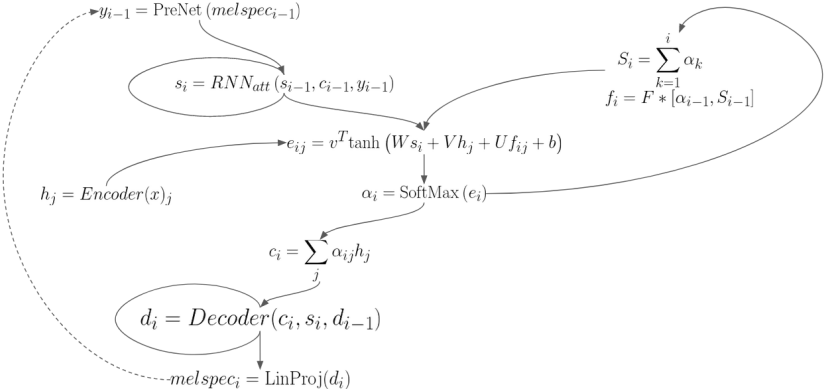


- Let's penalize attention for stepping away from diagonal
- Additional loss:

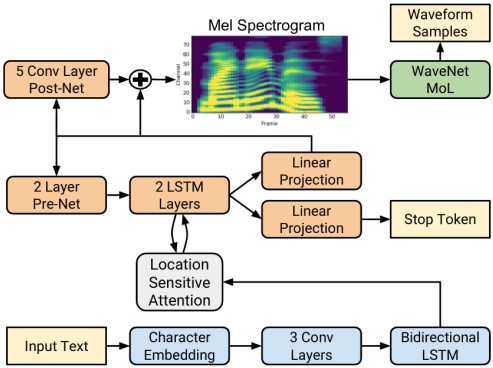
$$W_{nt} = 1 - \exp\left(-\left(\frac{n}{N} - \frac{t}{T}\right)^2 / 2g^2\right)$$

$$\mathcal{L}_{guided} = \sum_{n,t} \alpha_{nt} W_{nt}$$

Location-sensitive Attention



Tacotron2



- LSTM bidirectional encoder and uni-directional decoder with LSA
- Generates Mel-spectrogram
- Autoregression with dropout
- Double loss: MSE after decoder and MSE after Post-Net
- Wavenet neural vocoder

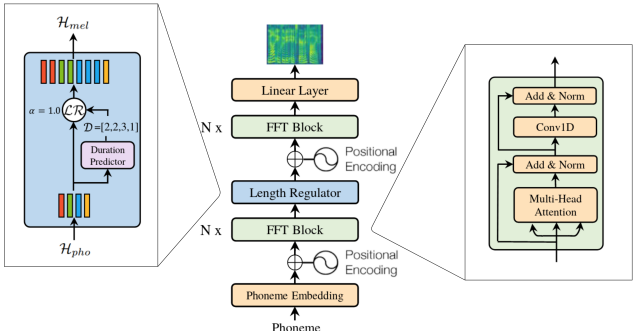
Attention-based Models

- Attention models can create nice global and local intonation detail
- Without any tweaks dot-product attention is unstable and works poor
- LSA boosts attention stability without (almost) decreasing the intonation quality
- Auto-regression loop requires regularization
- In terms of naturalness and intonation quality, Tacotron2 and its modifications is SotA for today

Upsampling + Attention

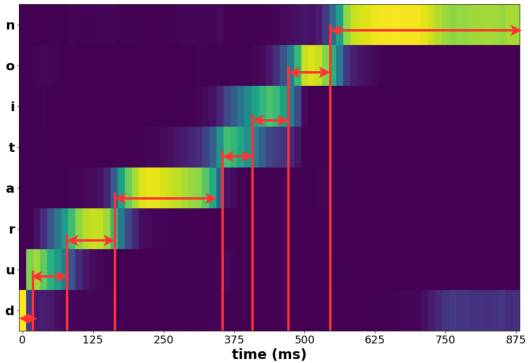
- Attention with auto-regression is slow
- Attention without auto-regression is unstable
- Let's give duration upsampling another chance!
- The main idea: let's split the information broadcast (to make nice global intonation) and dimensionality change (to get from phonemes to frames)

Feed-forward Transformers



- Fast Speech: N blocks of self-attention + duration prediction and upsampling
- Works with neural vocoders, generates mel-spectrogram
- As stable as LSTM+upsampling TTS
- As good in intonation as Tacotron
- Much faster (especially on GPU)

Getting Duration from Tacotron



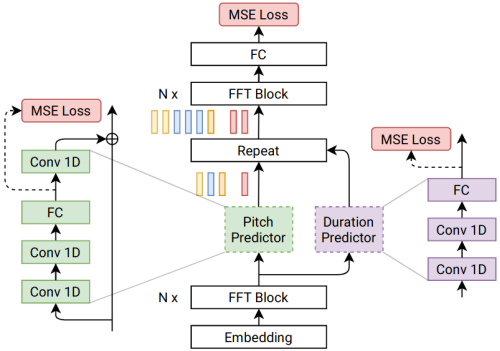
- Duration predictor quality is quite sensitive to the training data
- ASR labels are not very precise
- We can extract data from AR model (e.g. Tacotron alignment)

$$\{d_i\} = \text{argmax} \prod_{i=1}^{N} \left(\prod_{j=1}^{d_i} \alpha_{i, s_{i-1}+j} \right)$$

$$s_i = \sum_{k=1}^{i} d_k$$

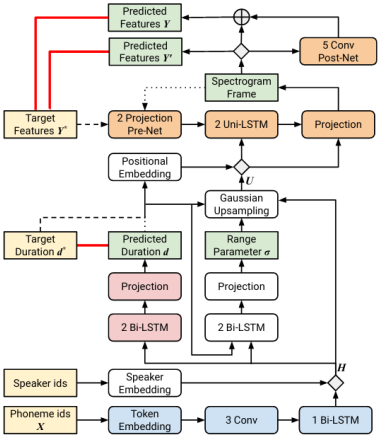
- We can use DP to find argmax

Acoustic Features at Phoneme Level 2.0



- We can predict phoneme means of f_0 in phoneme-wise subnet
- This makes the model more intonation-aware and increase the quality of speech
- If we use f_0-s and durations from other utterance or other speaker - we can copy intonation
- The model can be used as many-to-one VC
- This is also applicable to acapella songs conversion

Soft Upsampling



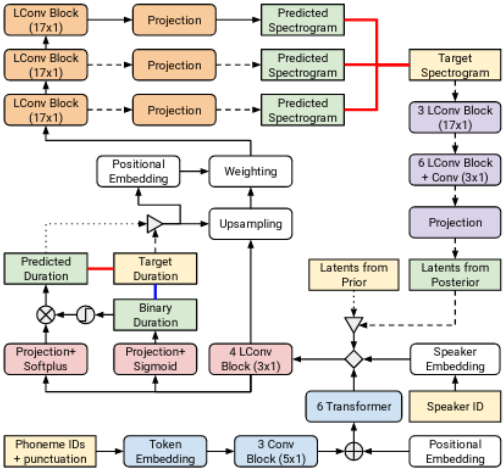
- At non-attentive tacotron attention was replaced with *Gaussian Upsampling*
- This is a "soft" way of upsampling phoneme-wise tensor H into frame-wise U:

$$c_i = \frac{d_i}{2} + \sum_{j=1}^{i-1} d_j,\ sigma_i = const \cdot d_i$$

$$w_{ti} = \frac{\mathcal{N}(t; c_i, \sigma_i^2)}{\sum_{j=1}^{N} \mathcal{N}(t; c_j \sigma_j^2)}$$

$$u_t = \sum_{i=1}^{N} w_{ti} h_i$$

FFT without T



- In parallel tacotron paper was proposed replacement of FFTs with LConv
- Dynamic LConv is a limited-field self-attention:

$$\mathrm{LConv}(X) = \mathrm{DepthConv}(X, f(X_{i-K,\dots,i+K}))$$
$$f(X) = \mathrm{SoftMax}(W \cdot X)$$

Upsampling + Attention

- Duration prediction with upsampling - stable phoneme-to-frame transform
- Global (FFT) or local (LConv) self-attention at phoneme and frame level is necessary to make a good intonation in duration-based models
- Attention-based AMs are useful to predict preciser ground truth duration (than ASR force-alignment)
- With softmax smoothing (instead of just upsample) phoneme-to-frame dependencies are modeled better
- We can additionally predict acoustic features statistics at phoneme level. That allows us to model intonation better and copy intonation from other records
- Self-attention + duration upsampling gives SotA in speech quality and speed for today

- LSTM + upsample - stable, average intonation fast
- Tacotron2 - SotA, slow, good intonation, can be unstable
- FastSpeech/FastPitch/Parallel Tacotron - FFT/LConv + upsample, fast, SotA intonation, stable

At the next lecture:

- Multi-speaker
- Multi-language
- Speech style and prosody control
- Using AMs for singing and voice conversion

End of Lecture