LPCNet: Improving Neural Speech Synthesis Through Linear Prediction

Jean-Marc Valin* (Amazon Web Services)
Jan Skoglund (Google LLC)

May 2019

*work performed while with Mozilla

Approaches to Speech Synthesis

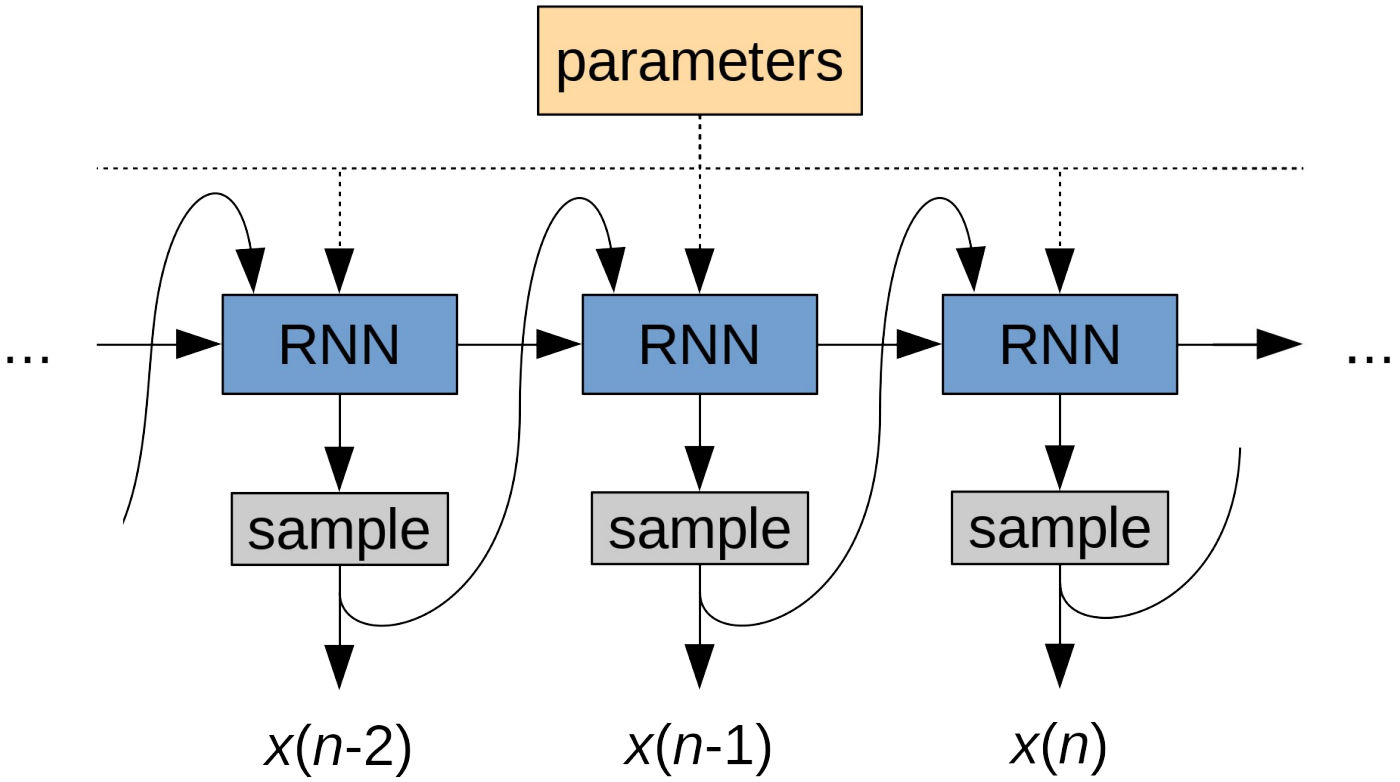
- “Old” DSP approach
  - Source-filter model
  - Synthesizing the excitation is hard
  - Acceptable quality at very low complexity
- New deep learning approach
  - Data driven
  - Results in large models (tens of MBs)
  - Very good quality at very high complexity
- Can we have the best of both worlds?

Neural Speech Synthesis

- WaveNet demonstrated impressive speech quality in 2016
  - Data-driven: learning from real speech
  - Auto-regressive: each sample based on previous samples
  - Based on dilated convolutions
  - Probabilistic: network output is not a value but a **probability distribution** for μ-law value
- Still some drawbacks
  - Very high complexity (tens/hundreds of GFLOPS)
  - Uses neurons to model vocal tract

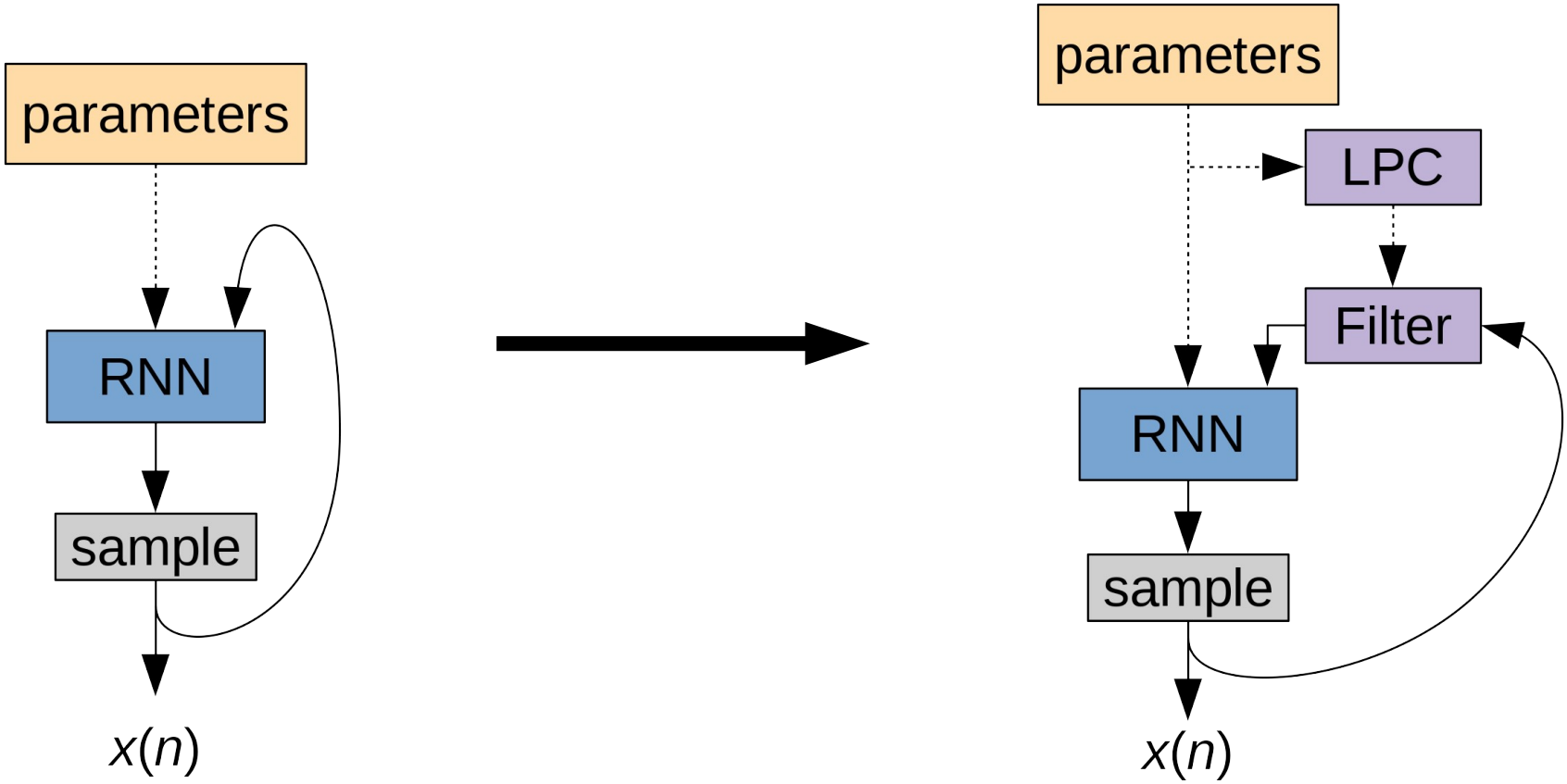
WaveRNN

- Replace dilated convolutions with RNN
- Addresses some of WaveNet's issues



LPCNet: Bringing Back DSP

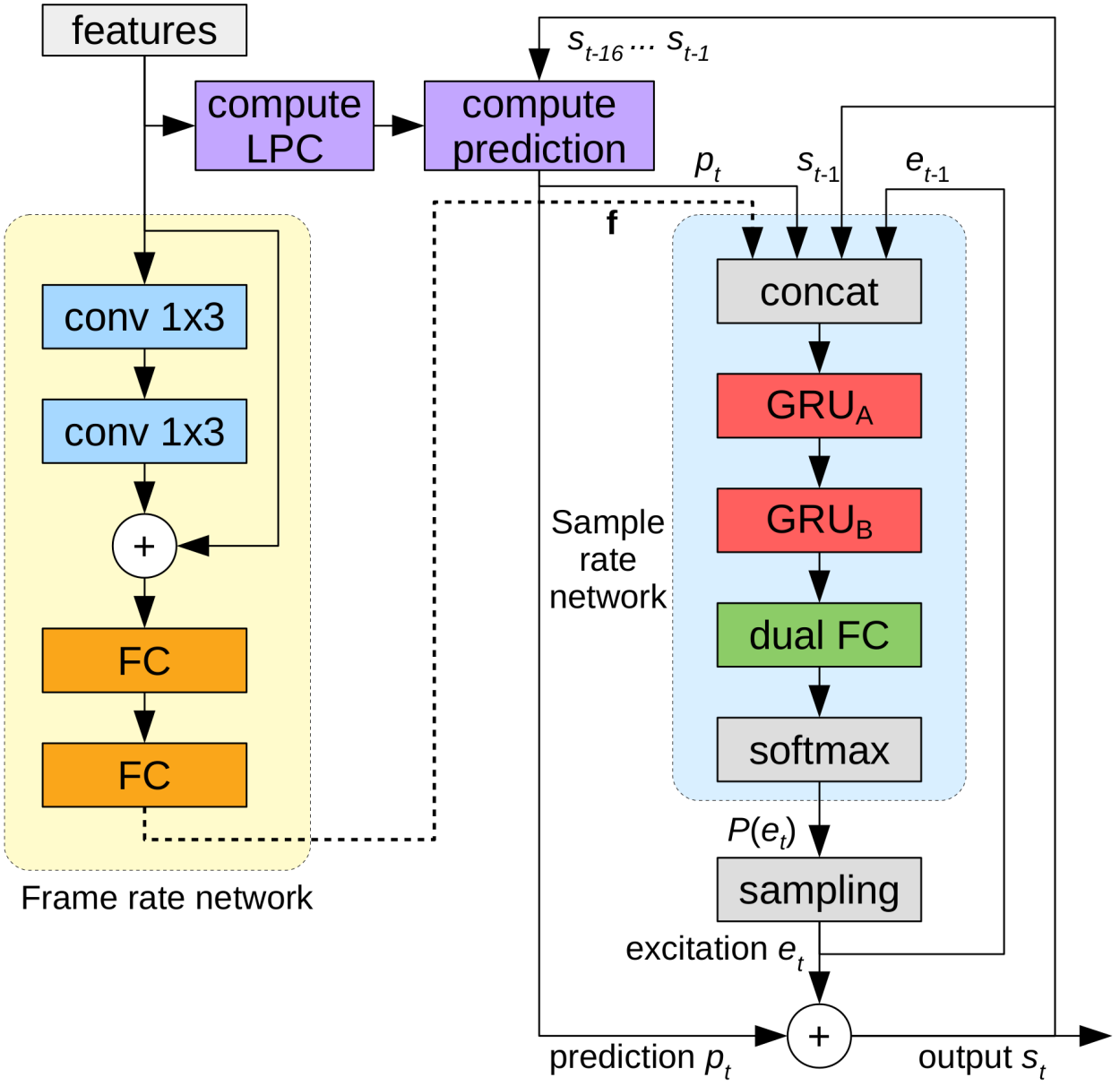
- Adding linear prediction to WaveRNN
  - Neurons no longer need to model vocal tract



Other Improvements

- Pre-emphasis
  - Boost HF in input/training data ($1 - \alpha z^{-1}$)
  - Apply de-emphasis on synthesis
  - Attenuates perceived μ-law noise for wideband
- Input embedding
  - Rather than use μ-law values directly, consider them as one-hot classifications
  - Learning non-linear functions for the RNN
  - Can be done at no cost by pre-computing matrix products

LPCNet: Complete Model



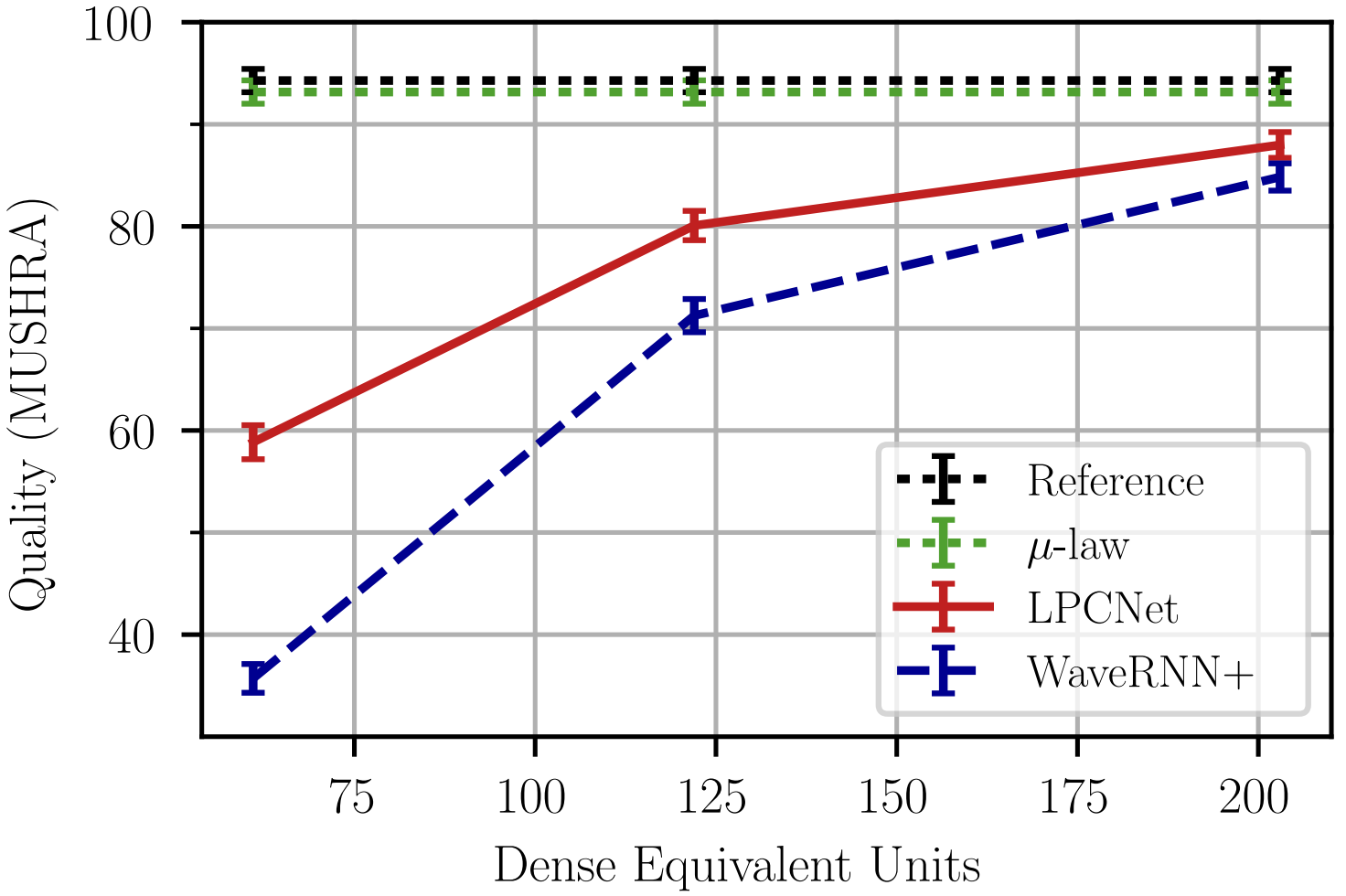
Training

- Inputs: signal (*t*-1), excitation (*t*-1), prediction (*t*)
- Output: excitation probability (*t*)
- Teacher forcing: use clean data as input
  - Need to avoid diverging due to imperfect synthesis not matching (perfect) training data
  - Inject noise in the input data
  - excitation = (clean signal) – (noisy prediction)
- Pre-emphasis and DC rejection applied to input
- Augmentation: varying gain and response

Complexity

- Use 16x1 block sparse matrices like WaveRNN
  - Add diagonal component to improve efficiency
  - 10% non-zero coefficients
  - 384-unit sparse GRU equivalent to 122-unit dense GRU
- Total complexity: 3 GFLOPS
  - No GPU needed
  - 20% of one 2.4 GHz Broadwell core
  - Real-time on modern phones with one core

Results



- Demo: https://people.xiph.org/~jm/demo/lpcnet/

Applications

- Text-to-speech (TTS)
- Low bitrate speech coding
- Codec post-filtering
- Time stretching
- Packet loss concealment (PLC)
- Noise suppression

Conclusion

- Bringing back DSP in neural speech synthesis
  - Improvement on WaveRNN
  - Easily real-time on a phone
- Future improvements
  - Use parametric output distribution
  - Add explicit pitch (as attention model?)
  - Improve noise robustness

Questions?

- LPCNet source code (BSD)
  - https://github.com/mozilla/lpcnet/