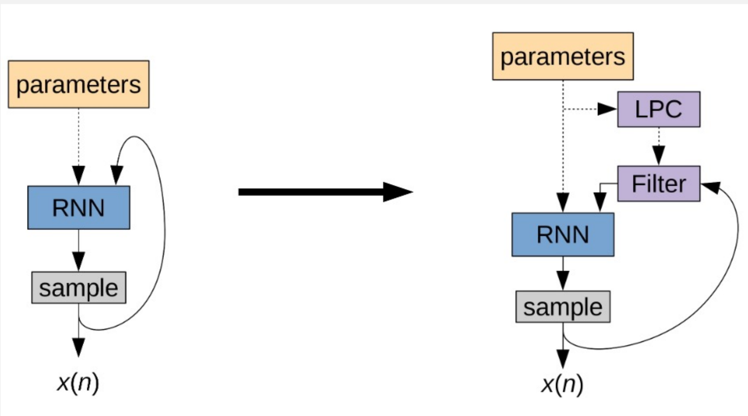
End-to-end LPCNet: A Neural Vocoder With Fully-Differentiable LPC Estimation

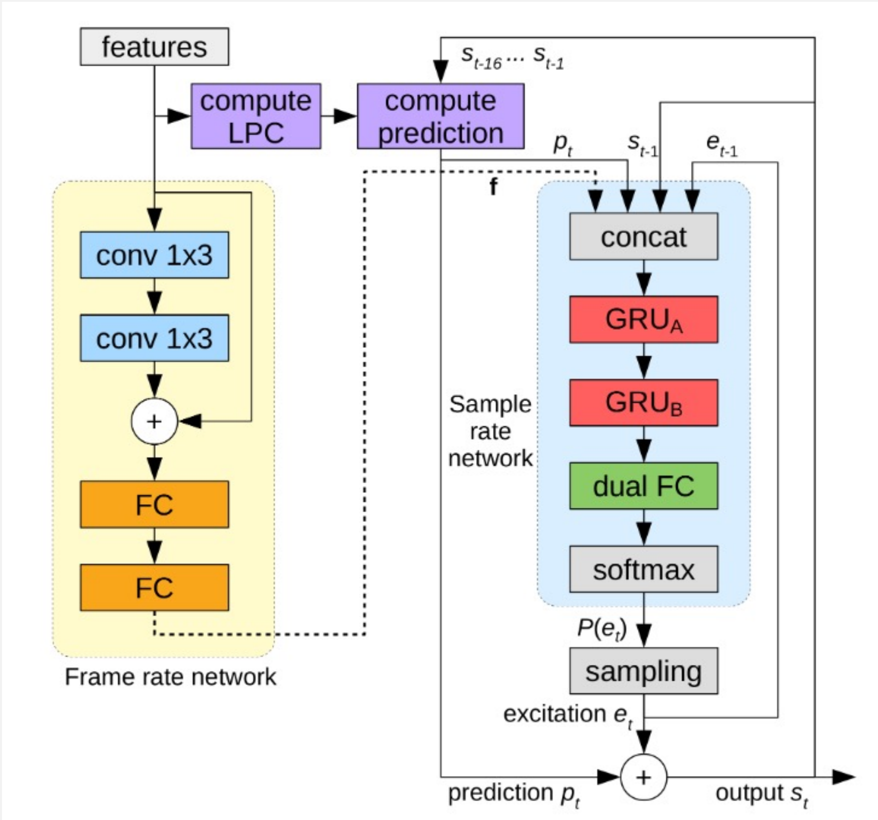
Krishna Subramani[1]*, Jean-Marc Valin[2], Umut Isik[2], Paris Smaragdis[1,2], Arvindh Krishnaswamy[2]

[1]UIUC, [2]Amazon Web Services

Speech Synthesis, LPCNet

- Use of Computers to synthesize intelligible speech
- DSP (Linear Prediction), Deep Learning (WaveRNN) => LPCNet





Features: Bark Cepstrum

Frame Rate Network, Sample Rate Network

- Signal (s_{t-1}), Excitation (e_{t-1}), Prediction (p_t)
- Output: Excitation (e_t)

Minimize cross-entropy loss

Why move ahead?

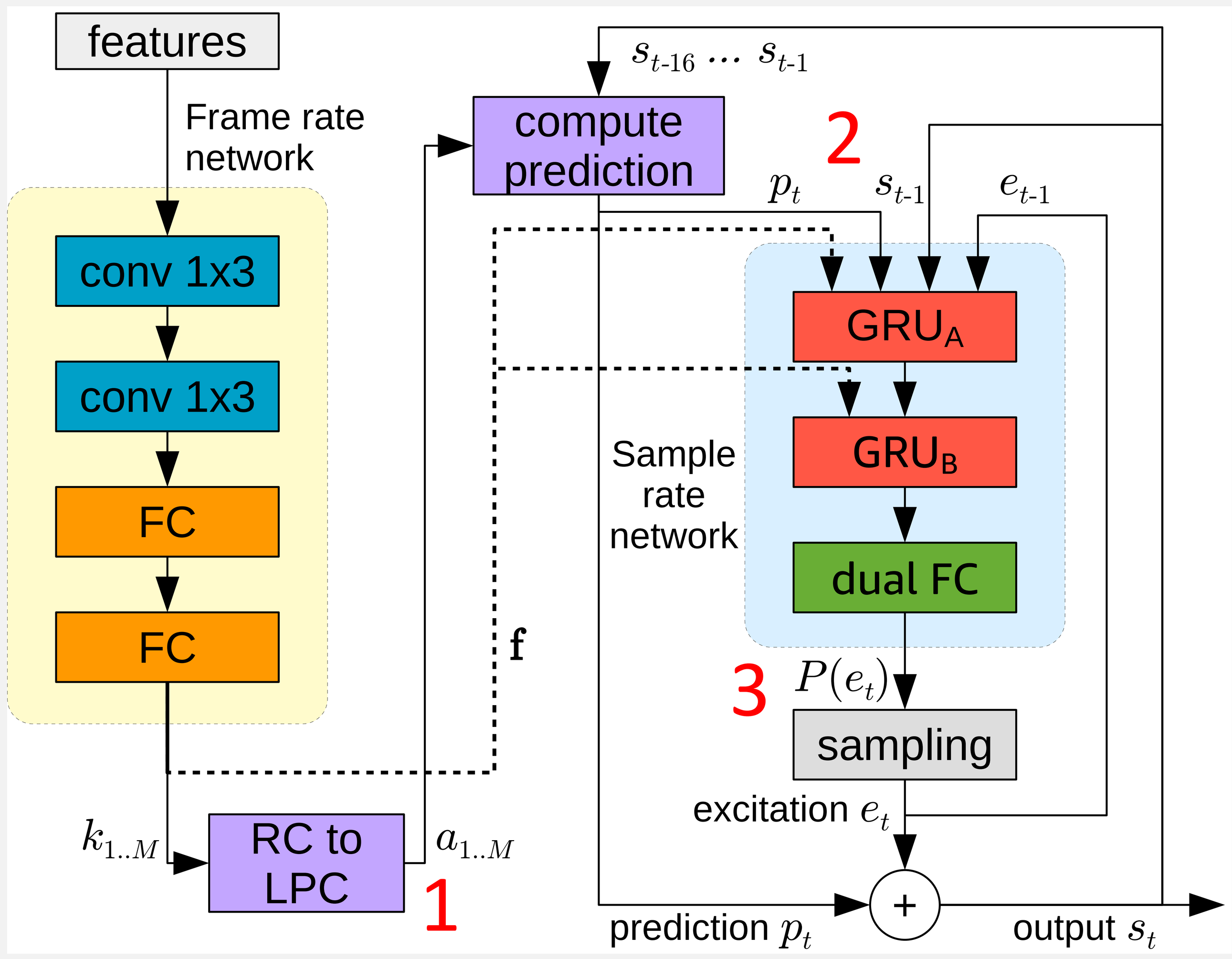
Explicit need of LPCs

- Need clean acoustic features which might not always be available

Why not "Learn" the LPCs?!

- Learn LPCs directly from the input features
  - Possible better fit
- Not restricted to clean speech features
  - Arbitrary codec features (or more general neural features)
  - Opens up LPCNet to end-to-end tasks like TTS or speech coding

End-to-end LPCNet



1. Learning the LPCs

- Using subset of FRN features to avoid overhead
- Learn Reflection Coefficients instead of LPC for stability
- Levinson recursion to convert RC -> LPC

2. Differentiable Embedding Lookup

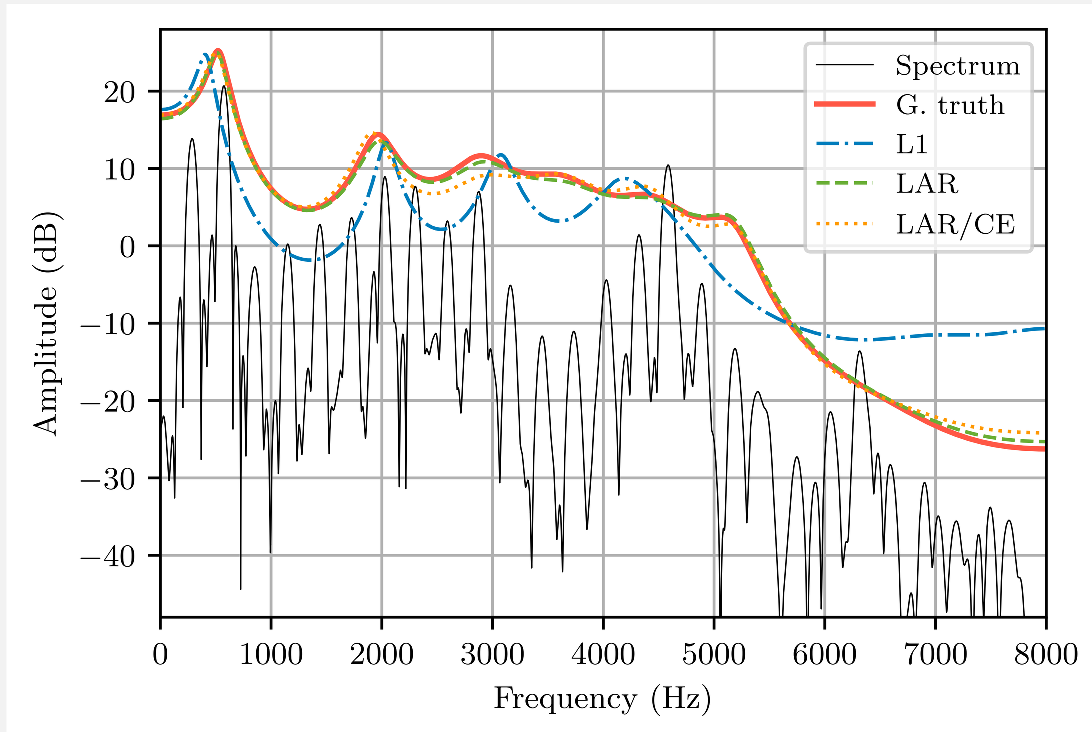
- μ-law quantized inputs prevent gradient backpropagation
- Linearly interpolate between adjacent embeddings

$$\mathbf{v}^{(i)}(x) = (1-f)\cdot \mathbf{v}_{\lfloor x \rfloor} + f\cdot \mathbf{v}_{\lfloor x \rfloor + 1}$$
$$f = x - \lfloor x \rfloor$$

3. Modified Loss Function

- LPCNet uses precomputed excitations,
  - Have to compute real-valued excitations on the fly
  - Interpolate excitation probabilities linearly (to propagate gradients)
- Naïve cross entropy minimization –> network cheats by forcing excitation to be large (μ-law spacing wider for larger excitation)
  - To overcome this, compute cross-entropy in linear domain

$$\mathcal{L} = -(p_i \log p_i + (1-p_i)\log(1-p_i)) + \alpha \left| \frac{\log(256)}{128}(p_t - s_t) \right|$$

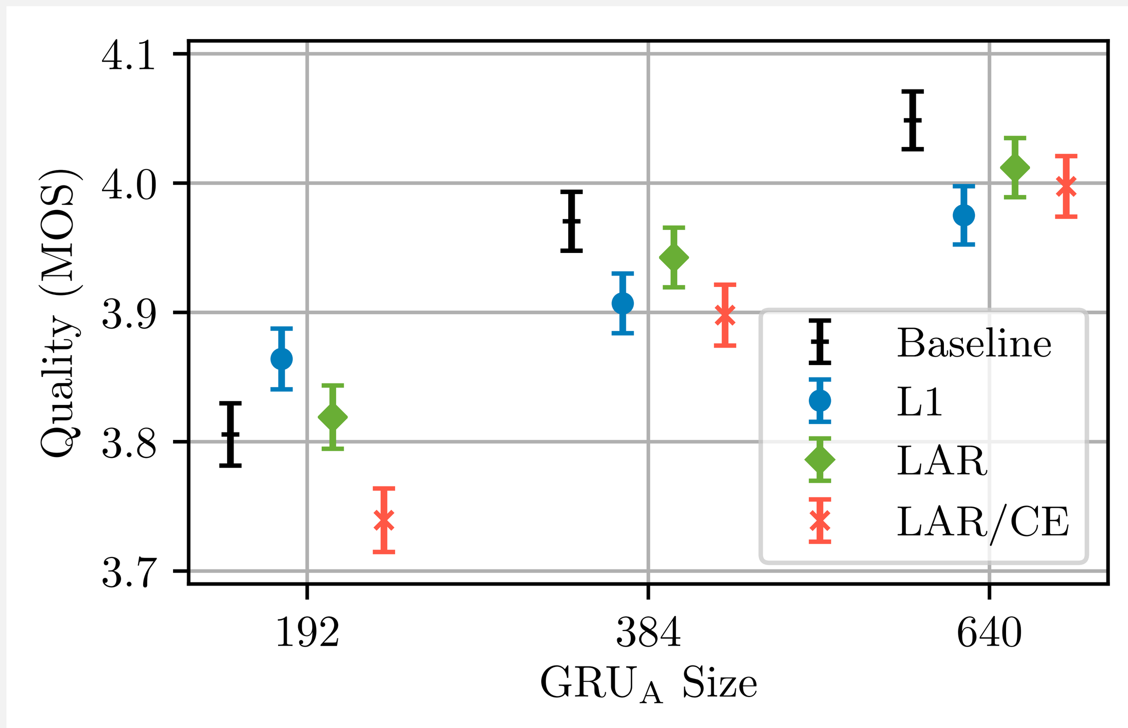


Model	LSD (dB)		
GRU$_A$ units	192	384	640
End-to-end L_1	3.58	3.64	3.64
End-to-end LAR	0.34	0.32	0.46
End-to-end LAR/CE	0.87	0.86	1.07

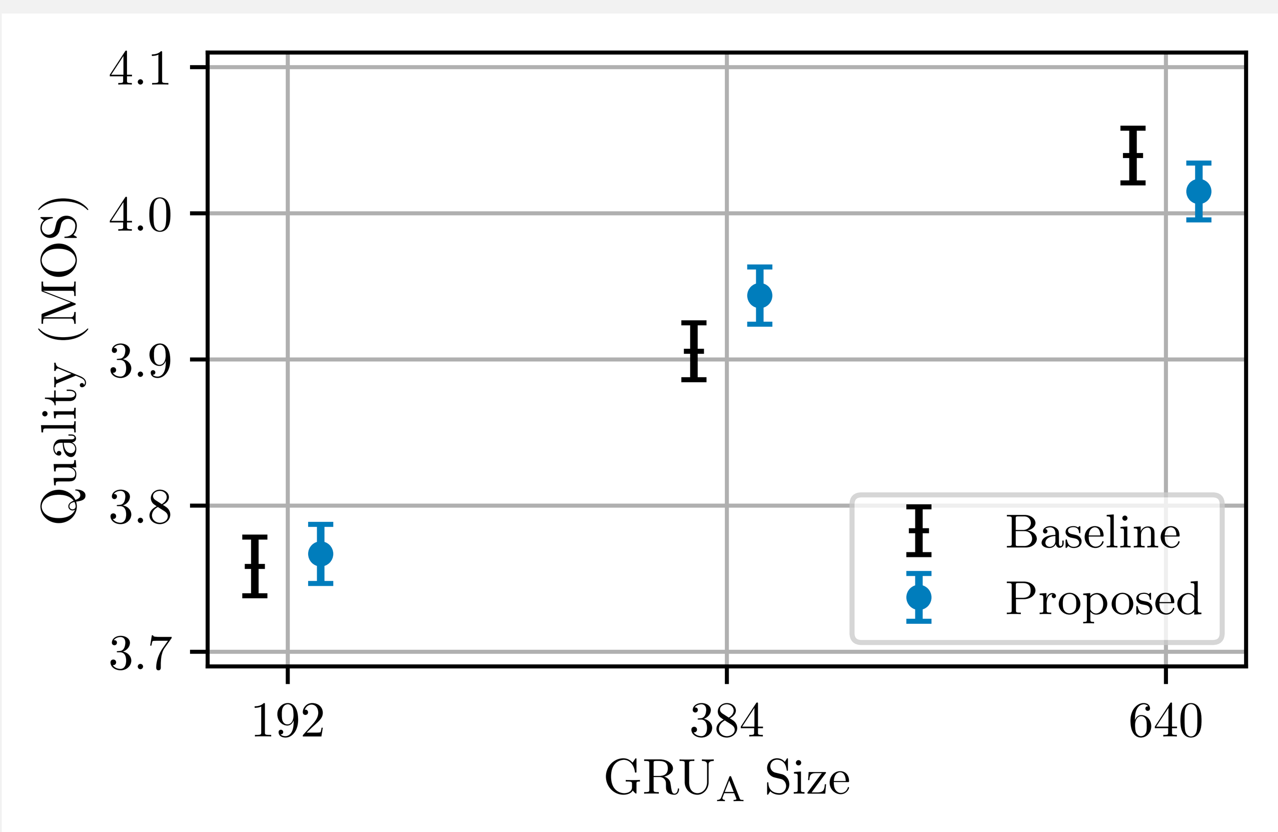
Need for Regularization

Prevent divergence + improve performance

- *L1*: Increase the weight on the second term in the loss
- Log-Area Ratio (*LAR*): Match network predicted LPCs to ground truth
  - Network predicts Reflection Coefficients (RCs)
  - Transform to Log-Area ratio (LAR) and match with ground truth LAR
- Log-Area Ratio matching with naïve cross-entropy minimization (*LAR/CE*)
  - Minimize μ-law cross-entropy to prevent divergence



Final Model



L1 + LAR Regularization

Conclusion

- End-to-end LPCNet that can learn the LPCs!
  - Not restricted to inputs from which we have to obtain the LPCs
- Improved Loss + Regularization for better performance
- Use LPCNet for broader range of applications
  - Speech Enhancement, TTS etc.

References

1. *WaveRNN*: https://arxiv.org/abs/1802.08435
2. *LPCNet*: https://arxiv.org/abs/1810.11846

*Work performed while at Amazon Web Services