Regularization Techniques for RNN-Generated Melodies

Nalini M. Singh
Harvard-MIT Division of Health Sciences and Technology
Cambridge, MA 02139
`nmsingh@mit.edu`

Abstract

Recurrent neural networks are commonly used for sequence generation tasks including sentence and music synthesis. However, RNNs frequently suffer from overfitting during these tasks for two reasons. First, as with any neural networks with a large number of parameters, it is easy to train a network that learns specifics of the training set instead of underlying statistical trends. Second, sequence-generating RNNs suffer from an inconsistency in the training and sequence generation procedures. During training, RNNs learn a probability distribution that maximizes the log-likelihood of each token, given the current state of the RNN and the previous token. In contrast, during the sequence generation process, future tokens are generated based on previously generated tokens, instead of being generated based on previous tokens from actual sequences. Thus, it is possible for the RNN to learn a distribution that yields a high likelihood on the training sequence but that does not explain the generated sequence well. Further, errors can compound during the generation process as tokens are generated from previously generated, possibly skewed tokens. In order to generate better sequences, it is desirable to identify regularization techniques that best combat both of these overfitting issues.

This work characterizes the performance of three such regular techniques on a MelodyRNN [1] trained on the Yamaha Piano E-Competition dataset. First, dropout regularization [2] is a commonly-used approach to solving the first flavor of overfitting described above in which individual units within the network are randomly zero-ed out during training, preventing them from learning specifics of the training dataset. Second, S. Bengio et al. [3] describe a "scheduled sampling" approach to the training-generation mismatch issue in which RNNs are trained using a mix of both actual and generated previous tokens, making the training process more similar to the sequence generation process. Third, Lamb et al. [4] developed "professor forcing," which trains a discriminator to differentiate between true and generated samples while simultaneously training a generator to maximize data likelihood and fool the generator.

The goal of this project was to apply these three techniques to Google's Melody RNN for melody generation and evaluate their performance. In particular, this required implementing scheduled sampling and professor forcing within the Melody RNN codebase (dropout regularization was already supported), and comparing each of these methods on the basis of:

- Similarity in the distribution of training and generation hidden RNN states
- Trends in training and evaluation loss
- Log-likelihood of generated sequences

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

T-SNE plots of the distribution of hidden states of each RNN suggest that scheduled sampling and professor forcing yield hidden states that are more similar between training and generation than observed with the baseline and dropout strategies. Further, while scheduled sampling and professor forcing attain higher loss during training than the baseline and dropout strategies, those two methods attain slightly lower loss on the evaluation data set, suggesting that these methods effectively mitigate overfitting. Finally, the scheduled sampling strategy achieved the highest log-likelihood of generated sequences.

These results suggest that scheduled sampling is the best strategy for regularization during melody synthesis with the MelodyRNN network.

References

[1] Magenta: Music and art generation with machine intelligence. `https://github.com/tensorflow/magenta`, 2017.

[2] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958, 2014.

[3] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179, 2015.

[4] Alex M Lamb, Anirudh GOYAL, Ying Zhang, Saizheng Zhang, Aaron C Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In *Advances In Neural Information Processing Systems*, pages 4601–4609, 2016.

[5] Ferenc Huszár. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? *arXiv preprint arXiv:1511.05101*, 2015.

[6] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, pages 2852–2858, 2017.