LightGAN: An Adversarial Approach to Natural Language Generation at a Large Scale

Jonathan Booher [1](jaustinb), Nithin Kannan (nkannan), Enrique De Alba (edealba)
Department of Computer Science[1]

Motivation

Natural language generation is increasingly important in today's world of digital assistants. It is, however, difficult to have these systems produce language that makes sense. Traditional approaches like n-grams suffer from repeating corpus text and RNNs suffer from poor scaling as the vocabulary increases.

We therefore present a method that we call LightGAN. A GAN trained with a novel LSTM design origianlly from Microsoft that can address large vocabularies with minimal space requirements.

Data

- 467 Million tweets from 2009 from the SNAP group [1]
- Example raw data:

T 2009-06-30 23:59:51
H http://twitter.com/eboe
W Out for karaoke and shots. Text if you dare. http://plurk.com/p/15f43e

Preprocessing

- Remove timestamps and user information
- Remove non english language tweets
- Replace websites, emojis, and @s with special tokens
- Pad the lines to the max length and remove words that appear less than 5 times.
- All preprocessing done beforehand to ensure that is not the bound
- Reduced vocabulary size to 100,000
- Example processed data:

Out for karaoke and shots. Text if you dare. <url> <naw> <naw> ... <naw>

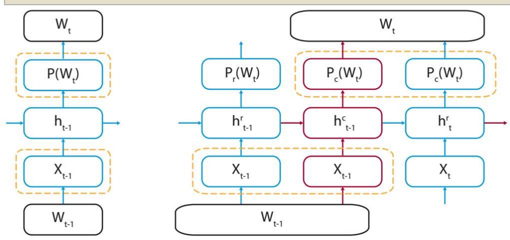
Method: LightRNN [2]

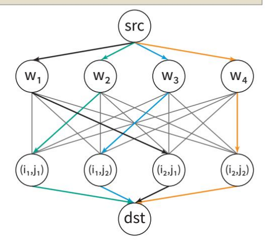
Key Idea:
- Allocate words into a 2D table
- Learn embeddings for each column and row
- A prediction for a row and column is a prediction for a word.
- Reallocate periodically to group similar words together in rows

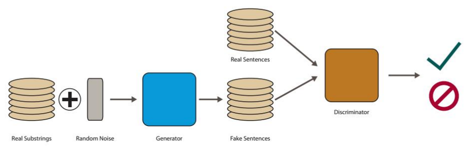
Savings:
- Table allows us to perform two softmaxes to ceil(sqrt(|V|)) instead of one to |V|
- Space savings of O(sqrt(|V|))

Drawback:
- Increased model complexity as operations are executed twice

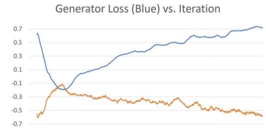
Word Allocation:
- Initially random
- Reallocate the words by solving a min cost max flow problem
- Have costs be proportional to the perplexity the model achieves on that word







Discriminator Loss (Orange) and Generator Loss (Blue) vs. Iteration



Method: The WGAN-GP Language Model

The Language Model:
- Frame as supervised learning problem: predict the next word
- Use RNNs for sequence prediction
- Pretrain the embedings and word allocation table

The WGAN [4]:
- Minimize the distance between the real and fake distributions
- Imporves the stability of traditional GAN
- Use same architecture for generator and discriminator

The GP [4]:
- An improved form of gradient clipping for GANs
- Penalize the gradients for being far from unit length

Results

Training was implemented using 'Curriculum Training'. Where the GAN is trained on increasingly large sequences [3]. Testing was accomplished using Beam Search with a beam width of 100.

RT <AT_TAG>
what wud you do
RT <AT_TAG> gdi fastfood
<AT_TAG> continually craazy
<AT_TAG> be oversleeping my scholl

Discussion

- Size of the dataset causes computability problems
- Attention and dropout in the generator greatly improved the stability of the model
- Model still has problems with longer sequences

Future Work

- Compare these results to those produced by a gan using traditional LSTM
- Train on different vocabulary sizes to see if the scaling affects accuracy
- Improve stability by working with different schedules for D and G

References

[1] J. Leskovec, A. Krevl. SNAP Datasets: Stanford Large Network Dataset Collection, June 2014.
[2] Xiang Li, Tao Qin, Jian Yang, Xiaolin Hu, Tie-Yan Liu: LightRNN: Memory and Computation-Efficient Recurrent Neural Networks. NIPS 2016: 4385-4393
[3] Ofir Press, Amir Bar, Ben Bogin, Jonathan Berant, Lior Wolf: Language Generation with Recurrent Generative Adversarial Networks without Pre-training. CoRR abs/1706.01399 (2017)
[4] Ishaan Gulrajani, Faruk Ahmed, Mart{\'{\i}}n Arjovsky, Vincent Dumoulin and Aaron C. Courville. Improved Training of Wasserstein GANs. CoRR abs/1704.00028 (2017)