Generating Yelp Reviews with GANs

Abhi Kulgod, Veeral Patel, Jayen Ram

Motivation

- Millions of users turn to Yelp as de facto site for choosing restaurants.[1]
- Because of this popularity, there is a monetary incentive for both good and bad actors to detect or generate bad reviews
- Good actor: Yelp filters out 25 percent of all submitted reviews it thinks are fake, biased, or unhelpful rants and raves.[2] This preserves integrity of their site.
- Bad actor: New restaurant may generate fake reviews to show a false sense of quality and obtain more customers
- Our generative adversarial network will tackle both of these problems. We will focus on the task of generating fake Yelp reviews since it's a much complex problem (especially for adversarial training).

Problem Setup

- Datasets of real and suspected fake Yelp reviews are used for adversarial training
- Limit generated review to 30 words and used massive dataset to help generate.
- For this problem, we want to generate reviews that won't sound like fake, generic bots; using adversarial training will allow us to optimize for sentence generation as opposed to prediction using normal RNNs

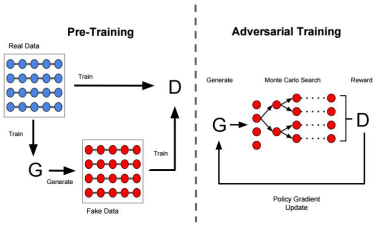
Dataset

- We used two datasets when training our LSTM and GAN setup:
  1. Yelp Review Dataset
     a. 5 million reviews from Yelp[3]
     b. from 11 metropolitan areas
     c. 200,000 were used for training
  2. Fake/Real Yelp Dataset[4]
     a. 200,000 reviews from Yelp
     b. labelled as suspected real and suspected fake
     c. from 2 metropolitan areas (Chicago and NYC)

Methods

Overview

- Model the Generator as an RL agent[5] where:
  - States and actions are represented by generated sequences and tokens
  - Reward is modeled by the Discriminator



Generator

- We first learn word embeddings using Word2Vec on the Yelp dataset
- G is an RNN with LSTM, pre-trained on real Yelp sequences
- Then begin adversarial training with loss:

$$J(\theta) = -\mathbb{E}[R_T|s_0] = -\sum_{y \in \gamma} G_\theta(y|s_0) \cdot Q_D^G(y, s_0)$$

Policy Gradient

- Given the discrete, non-differentiable input space, we use policy gradient methods to estimate updates to the policy, G:

$$\pi_\theta(s = Y_{1:t-1}, a = y_t) = G_\theta(y_t|Y_{1:t-1})$$

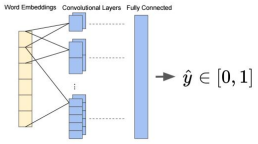
- Using the REINFORCE algorithm,

$$\nabla_\theta J(\theta) \approx \sum_{t=1}^{T} \mathbb{E}_{y_t \sim G_\theta}[\nabla_\theta \log G_\theta(y_t|Y_{1:t-1}) \cdot Q_D^G(y_t, Y_{1:t-1})]$$

Discriminator

- D is a CNN with parallel filters of sizes [1, 2, 3, 4, 5, 8, 10, 15, 20] to emulate looking at unigram, bigram, etc.[5]
- Pre-trained using samples from G and Yelp reviews

$$J(\phi) = -\frac{1}{m_d}\sum_{i}^{m_d} \log(D_\phi(x^{(i)})) - \frac{1}{m_g}\sum_{i}^{m_g} \log(1 - D_\phi(G_\theta(z^{(i)})))$$



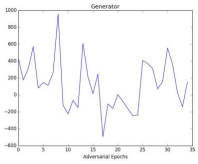
Optimization

- When generating sequences, we experiment with both Monte Carlo search and Beam search for generating our next token
  - MCS: sample from complete distribution modeled by G
  - Beam Search: samples from top-k most likely next tokens and we decrease beam width over our adversarial training (limit is 1000)
- We then normalize D's outputs in order to control the variance of the reward signal

Results

n-Gram Scores[6]

Algorithm	n=2	n=3	n=4
Random	0.03977	0.00001	0.0000
LSTM	0.37874	0.17603	0.06958
GAN - BS	0.26772	0.11074	0.03579
GAN - MCS	0.29007	0.13991	0.05451
GAN - NORM	0.35383	0.15388	0.05518



- great food. epic staff. i had the jambalaya and it was fantastic. fried plantain was excellent. <UNK> was very sweet and delicious.
- one of my favorite places for italian food every time (had a reservation available) but somehow nyc nice.

Discussion

- We faced challenges with tuning hyperparameters as there was high instability
- Applying GANs to real data with inconsistent content proved difficult, as opposed to modeling synthetic data as other papers have done
- Reviews are about a mix of restaurants from a mix of people--not one theme like generating Shakespeare or Obama speeches.
- Normalizing rewards has a stabilizing effect on generator

Future Work

- Incorporate new evaluation methods on quality of sentences (crowdsource, etc).
- Experiment and tune loss functions to increase GAN stability
- Conditional Generation to generate different kinds of reviews based on different start tokens

References

1. "An Introduction to Yelp Metrics as of December 31, 2017." *Yelp*, www.yelp.com/factsheet.
2. Grover, Joel, and Amy Corral. "Don't Fall for Fake Reviews Online." *NBC Southern California*, NBC Southern California, 30 Nov. 2017, www.nbclosangeles.com/news/local/Fake-Reviews-on-Yelp-Facebook-Google-447798103.html.
3. Yelp, Inc. *Yelp Dataset | Kaggle*, 6 Feb. 2018, www.kaggle.com/yelp-dataset/yelp-dataset.
4. *YelpZip Dataset*. odds.cs.stonybrook.edu/yelpzip-dataset/.
5. Yu, et al. "SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient." *SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient*, 25 Aug. 2017, arxiv.org/abs/1609.05473.
6. William, et al. "MaskGAN: Better Text Generation via Filling in the_____." *MaskGAN: Better Text Generation via Filling in the_____*, 1 Mar. 2018, arxiv.org/abs/1801.07736.