CS230

Generative Adversarial Networks for Story Generation (Natural Language Processing)

Andrew E. Freeman
Department of Statistics
Stanford University
aefree@stanford.edu

Anna C. Shors
Department of Statistics
Stanford University
ashors@stanford.edu

Anastasios Sapalidis
Department of Statistics
Stanford University
tsap@stanford.edu

1 Problem Description

We set out to investigate the generation of coherent stories in paragraph form. Natural Language Processing (NLP) has evolved significantly in recent years, and it is now possible to generate long form texts using language models.

Recently, several papers have investigated the use of Generative Adversarial Networks (GANs) in the generation of text. However, these methods either require the use of complex reinforcement learning and policy gradient methods and are thus unstable during training (15), or are limited to producing single sentences (4) (1). We aim to combine ideas from sequential conditional GANs (11) (9) with those of text GANs to develop a simpler GAN architecture that can generate full paragraphs of text.

2 Challenges

One challenge to this project is selecting the appropriate learning architecture. Traditionally, NLP uses a "picking" function to select the next word in a sentence. This function is not differentiable, which prevents it from being utilized in a GAN.

Another significant challenge will be generating coherent paragraphs. Much of the work in GANs for text generation has focused on generating single sentences rather than continuous paragraphs. Giving a model the ability to "remember" previously generated sentences will require modifications to existing GAN architectures.

3 Dataset

We propose to use the Wikipedia Movie Plot dataset available on Kaggle (8). This dataset comprises approximately 35,000 short movie summaries scraped from Wikipedia.

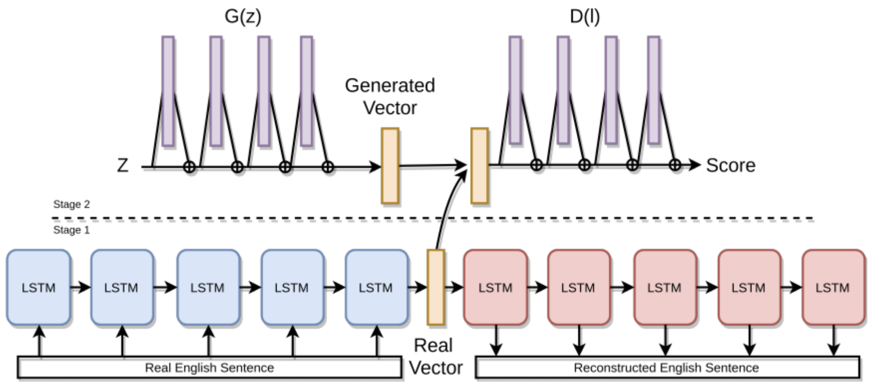

Figure 1: Proposed GAN + autoencoder architecture taken from (4)

4 Methods

We plan to expand on the architecture presented by Donahue and Rumshisky (4). The authors propose a method of training a GAN generator to output encoded sentence "context vectors", which can then be decoded using an autoencoder decoder. The autoencoder is also used to generate true sentence vectors. The true sentence vectors, as well as the generated sentence vectors, are fed as input into the discriminator, which is trained to discriminate the "real" sentence vectors from the "fake" ones. This architecture is outlined in Figure 1.

Because we wish to use this model to generate a series of fluent sentences, we need a way for the generator to condition on the previously generated sentence vectors when creating a new sentence. We propose using an attention mechanism (10) over the series of previously generated vectors to create a fixed-length representation of the story information which can be fed into the generator. Only after the generator has generated a full story will the output be given to the discriminator. The discriminator would then receive a series of sentence vectors as input and be trained to distinguish the fake stories from the real ones. After the GAN has finished training, we can decode the story by passing the generated sentence vectors into the decoder.

5 Evaluation

We intend to use a combination of perplexity and BLEU scores, as well as human evaluations, to measure the quality and fluency of the generated stories. Because metrics for natural language generation tasks often do not reflect the true quality of text, we will put substantial emphasis on the human evaluations. We plan to use a vanilla RNN language model as a baseline with which to compare our results.

References

[1] AKMAL HAIDAR, M., REZAGHOLIZADEH, M., DO-OMRI, A., AND RASHID, A. Latent Code and Text-based Generative Adversarial Networks for Soft-text Generation. *arXiv e-prints* (Apr. 2019), arXiv:1904.07293.

[2] BAMMAN, D., O'CONNOR, B., AND SMITH, N. A. Learning Latent Personas of Film Characters. *ACL 2013, Sofia, Bulgaria* (Aug. 2013).

[3] CHINTAPALLI, K. Generative adversarial networks for text generation — part 3: non-rl methods, Jun 2019.

[4] DONAHUE, D., AND RUMSHISKY, A. Adversarial Text Generation Without Reinforcement Learning. *arXiv e-prints* (Oct. 2018), arXiv:1810.06640.

[5] FAN, A., LEWIS, M., AND DAUPHIN, Y. Hierarchical Neural Story Generation. *arXiv e-prints* (May 2018), arXiv:1805.04833.

[6] FANG, L., ZENG, T., LIU, C., BO, L., DONG, W., AND CHEN, C. Outline to Story: Fine-grained Controllable Story Generation from Cascaded Events. *arXiv e-prints* (Jan. 2021), arXiv:2101.00822.

[7] HERRERA-GONZÁLEZ, B., GELBUKH, A., AND CALVO, H. Automatic Story Generation: State of the Art and Recent Trends. *Advances in Computational Intelligence 19th Mexican International Conference on Artificial Intelligence* (Oct. 2020).

[8] JUSTINR. Wikipedia movie plots. `kaggle.com/jrobischon/wikipedia-movie-plots/metadata`, 10 2018.

[9] LI, Y., GAN, Z., SHEN, Y., LIU, J., CHENG, Y., WU, Y., CARIN, L., CARLSON, D., AND GAO, J. StoryGAN: A Sequential Conditional GAN for Story Visualization. *arXiv e-prints* (Dec. 2018), arXiv:1812.02784.

[10] LUONG, M.-T., PHAM, H., AND MANNING, C. D. Effective Approaches to Attention-based Neural Machine Translation. *arXiv e-prints* (Aug. 2015), arXiv:1508.04025.

[11] MIRZA, M., AND OSINDERO, S. Conditional Generative Adversarial Nets. *arXiv e-prints* (Nov. 2014), arXiv:1411.1784.

[12] PRANAVPSV. Gpt2 genre-based story generator. `github.com/pranavpsv/Genre-Based-Story-Generator`, 2021.

[13] SAEED, A., ILIĆ, S., AND ZANGERLE, E. Creative GANs for generating poems, lyrics, and metaphors. *arXiv e-prints* (Sept. 2019), arXiv:1909.09534.

[14] SHREYDESAI. Implementation of adversarial text generation without reinforcement learning. `github.com/shreydesai/latext-gan`, 2019.

[15] YU, L., ZHANG, W., WANG, J., AND YU, Y. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. *arXiv e-prints* (Sept. 2016), arXiv:1609.05473.

[16] ZHU, Y., LU, S., ZHENG, L., GUO, J., ZHANG, W., WANG, J., AND YU, Y. Texygen: A Benchmarking Platform for Text Generation Models. *arXiv e-prints* (Feb. 2018), arXiv:1802.01886.