Transformer Based Text Embeddings for Text to Image Generation

Andrew Peng, David Lin
University of California, Berkeley

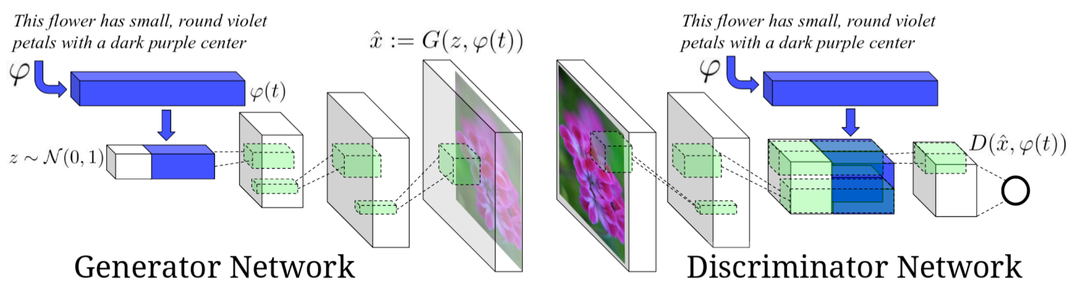
Introduction

We decided to see if we could improve text to image synthesis with Generative Adversarial networks by using transformer based encoders. The original paper used a RNN-CNN encoder that was pre-trained.

The goal of text to image generation is to generate pixel images from input sentence descriptions such as “This flower has large red petals and a yellow center”.

Recently deep convolutional networks have seen good results, specifically with dueling Generative Adversarial networks. We explore the effect of different text encodings on the generators output, using skip-thought vectors, transformer based encoders, and a bi-directional GRU encoder.

Model



Model overview

https://github.com/aelnouby/Text-to-Image-Synthesis/blob/master/images/pipeline.png

Dataset and Method

We used the Oxford 102 flowers dataset which has 102 classes of different flowers, with each class containing between 40 and 258 images. Each image has 10 sentence descriptions. The dataset was split into 80% train/val and 20% test. The data is normalized into the range [-1,1].

The model was implemented using PyTorch.

For our generator, we project the sentence embedding into a size 128 vector and concatenate it with a random vector sampled from a normal distribution. Then the image is upsampled using transpose convolutions.

For our discriminator, we use strided convolutions to downsample the image. After each convolution we use batch normalization and the leaky ReLU activation function. When we have a 4x4 spatial dimension we concatenate the projected sentence embedding and feed it through 2 last convolutional layers and finally a sigmoid activation function.

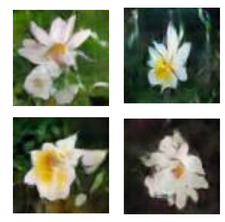
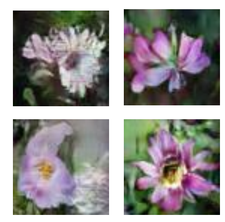
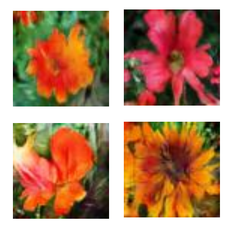
We tested four different embeddings to see which would perform the best, an end-to-end trained bi-directional RNN encoder, a pre-trained skip thought encoder, a BERT encoder, and an OpenAI GPT encoder. The RNN and skip thought encoders were trained for 600 epochs with the Adam optimizer and learning rate 0.0002. BERT was trained for 400 epochs, while GPT was trained for 200 epochs.

The inception scores are calculated using the test dataset on inceptionV3.

Encoding	Inception Score	Encoding Size	Batch size	Epochs
End-to-end	2.846	1024	64	600
Skip-thought	3.103	2400	64	600
BERT*	**3.226**	768	32	400
GPT*	2.646	768	32	200

*training unfinished

Results

Text	End to End	Skip thought	BERT
This flower is yellow-green with warped petals and small green leaves			
The petals on this flower are white with a yellow center			
This flower has petals that are yellow, white and purple and has dark lines			
The petals of the flower are narrow and extremely pointy, and consist of shades of red, orange			

Above: Images generated by our GAN with different encodings

Discussion

We adopted several methods in training in order to stabilize the loss including label smoothing, adding feature matching, and doubling the generator gradient updates when the discriminator became too powerful.

We also can visualize interpolation between embedded vectors to see how each embedding affects the generated images.

This one flower has large **red** petals with a brown center → This one flower has large **blue** petals with a brown center

Above: Skip Thought Embedding

Above: BERT Embedding

Conclusions

As you can see, transformer based encodings increase the inception score of the generated images. However, using inception score may not be the best way to benchmark GANs (see A Note on the Inception Score [8]). Visually, the transformer based encodings produced images that were on similar to the RNN based encodings.

References

1. Scott Reed et al. Generative Adversarial Text to Image Synthesis [arXiv:1605.05396]
2. Salimans, Goodfellow, Zaremba, Cheung, Radford, Chen. Improved Techniques for Training GANs [arXiv:1606.03498]
3. Devlin, Lee, Chang, Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding [arXiv: 1810.04805v1]
4. Kiros et al. Skip Thought Vectors. [arXiv:1506.06726v1]
5. DCGAN in PyTorch [https://github.com/pytorch/examples/tree/master/dcgan]
6. Xu et al. AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks
7. Inception Score for PyTorch [https://github.com/sbarratt/inception-score-pytorch]
8. Barratt, Sharma. A Note on the Inception Score. [arXiv:1801.01973v2]

Contact

apeng@berkeley.edu
lin.david@berkeley.edu

© Poster Template by Genigraphics® 1.800.790.4001 www.genigraphics.com