Reading And Special Problems Report

Name: Xu Ouyang

CWID: A20361390

Abstract

This is a report for reading and special problems. I continued implement the model that generates images from natural language descriptions named improved conditional Cycle GAN. Cycle GAN is to learn a mapping G: X → Y such that the distribution of images from G(X) is indistinguishable from the distribution Y using an adversarial loss (and vice versa). And introduce a cycle consistency loss to push F(G(X)) ≈ X (and vice versa). We try on kinds of dataset such as Flickr30k, Caltech-UCSD Birds and Oxford Flowers, and we hope our model can input description annotation and output image which is related to this annotation finally.

1 Introduction

There exists some promising researches of computer vision and image understanding in natural image statistics. One of research direction is exploit the inference and generative capabilities of deep neural networks. The problem of generating images from visual descriptions gained interest in the research community, but it is far from being solved. Previously study on generating images using defined distributions which were conditioned on classification labels.

In this paper, we illustrate how conditional cycle generative adversarial network can be used to translate descriptions into real images. We use the Oxford Flowers dataset along with five text descriptions per image we collected as our evaluation setting at the beginning of experiment. And then we try on Caltech-UCSD Birds and flickr-30k datasets. We also use skip vector which is a group of related models that are used to produce word embeddings in each description. We demonstrate its performance both on the training set categories and on the testing set.

2. Related work

Many researchers have recently exploited the capability of deep convolutional decoder networks to generate realistic images. Dosovitskiy et al. (2015) trained a deconvolutional network (several layers of convolution and upsampling) to generate 3D chair renderings conditioned on a set of graphics codes indicating shape, position and lighting. Yang et al. (2015) added an encoder network as well as actions to this approach. They trained a recurrent convolutional encoder-decoder that rotated 3D chair models and human faces conditioned on action sequences of rotations. Reed et al. (2015) encode transformations from analogy pairs, and use a convolutional decoder to predict visual analogies on shapes, video game characters and 3D cars.

Generative adversarial networks (Goodfellow et al., 2014) have also benefited from convolutional decoder networks, for the generator network module. Denton et al. (2015) used a Laplacian pyramid of adversarial generator and discriminators to synthesize images at multiple resolutions. This work generated compelling high-resolution images and could also condition on class labels for controllable generation. Radford et al. (2016) used a standard convolutional decoder, but developed a highly effective and stable architecture incorporating batch normalization to achieve striking image synthesis results.

In contemporary work Mansimov et al. (2016) generated images from text captions, using a variational recurrent autoencoder with attention to paint the image in multiple steps, similar to DRAW (Gregor et al., 2015). Impressively, the model can perform reasonable synthesis of completely novel (unlikely for a human to write) text such as “a stop sign is flying in blue skies”, suggesting that it does not simply memorize. While the results are encouraging, the problem is

highly challenging and the generated images are not yet realistic, i.e., mistakable for real. Their model can in many cases generate visually-plausible 64×64 images conditioned on text, and is also distinct in that their entire model is a GAN, rather only using GAN for post-processing.

Cycle GAN, an approach for learning to translate an image from a source domain X to a target domain Y in the absence of paired examples. Their goal is to learn a mapping G: X → Y such that the distribution of images from G(X) is indistinguishable from the distribution Y using an adversarial loss. Because this mapping is highly under-constrained, we couple it with an inverse mapping F: Y → X and introduce a cycle consistency loss to push F(G(X)) ≈ X (and vice versa). Qualitative results are presented on several tasks where paired training data does not exist, including collection style transfer, object transfiguration, season transfer, and photo enhancement, etc. Quantitative comparisons against several prior methods demonstrate the superiority of their approach.

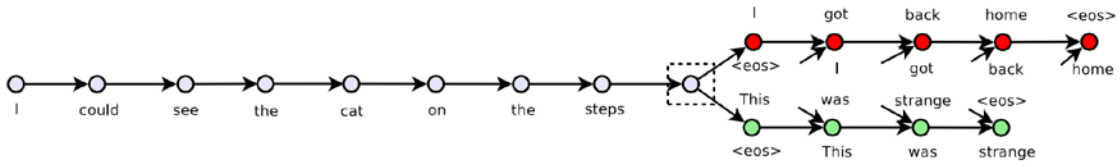
Building on ideas from these many previous works, we develop a simple and effective approach for text-based image synthesis using a character-level text encoder, improved conditional Cycle GAN. We propose a novel architecture and learning strategy that leads to compelling visual results. We focus on the case of fine-grained image datasets, for which we use the recently collected descriptions for Oxford flower images with 10 human-generated captions per image.

3. Background

3.1 Skip-Thought Vectors

Skip-Thought Vectors was to come up with a useful embedding for sentences that was not tuned for a single task and did not require labeled data to train. They took inspiration from Word2Vec skip-gram and attempt to extend it to sentences.

Skip-thought vectors are created using an encoder-decoder model. The encoder takes in the training sentence and outputs a vector. There are two decoders both of which take the vector as input. The first attempts to predict the previous sentence and the second attempts to predict the next sentence. Both the encoder and decoder are constructed from recurrent neural networks (RNN). Multiple encoder types are tried including uni-skip, bi-skip, and combine-skip. Uni-skip reads the sentence in the forward direction. Bi-skip reads the sentence forwards and backwards and concatenates the results. Combined-skip concatenates the vectors from uni- and bi-skip. Only minimal tokenization is done to the input sentences. A diagram indicating the input sentence and the two predicted sentences is shown below.



3.2 Generative adversarial networks

Generative adversarial networks (GANs) consist of a generator G and a discriminator D that compete in a two-player minimax game: The discriminator tries to distinguish real training data from synthetic images, and the generator tries to fool the discriminator. Concretely, D and G play

the following game on V(D,G):

$$\min_G \max_D V(D, G) = E_{x\sim p_{data}(x)}[\log D(x)] + E_{x\sim p_z(z)}[\log(1 - D(G(z)))]$$

Goodfellow et al. (2014) prove that this minimax game has a global optimium precisely when pg = pdata, and that under mild conditions (e.g. G and D have enough capacity) pg converges to pdata. In practice, in the start of training samples from D are extremely poor and rejected by D with high confidence. It has been found to work better in practice for the generator to maximize log(D(G(z))) instead of minimizing log(1 − D(G(z))).

3.3 Cycle GAN

A Generative Adversarial Network (GAN) consists of two neural networks, a generator $G_{X\to Y}$ and a discriminator D_Y, which are iteratively trained in a two-player minimax game manner. The adversarial loss $L(G_{X\to Y}, D_Y)$ is defined as

$$L(G_{X\to Y}, D_Y) = \min_{\Theta g} \max_{\Theta d} \{E_y[\log D_Y(y)] + E_x[\log(1 - D_Y(G_{X\to Y}(x)))]\} \quad (1)$$

where Θg and Θd are respectively the parameters of the generator $G_{X\to Y}$ and discriminator D_Y, and $x \in X$ and $y \in Y$ denotes the training data in source and target domain respectively. In Cycle GAN, X and Y are two different image representations, and the Cycle GAN learns the translation $X \to Y$ and $Y \to X$ simultaneously. Different from “pix2pix”, training data in Cycle GAN is unpaired. Thus, they introduce Cycle Consistency to enforce forward-backward consistency which can be considered as “pseudo” pairs of training data. With the Cycle Consistency, the loss function of Cycle GAN is defined as:

$$L(G_{X\to Y}, G_{Y\to X}, D_X, D_Y) = L(G_{X\to Y}, D_Y) + L(G_{Y\to X}, D_X) + \lambda L_c(G_{X\to Y}, G_{Y\to X}) \quad (2)$$

where $L_c(G_{X\to Y}, G_{Y\to X}) = ||G_{Y\to X}(G_{X\to Y}(x)) - x||_1 + ||G_{X\to Y}(G_{Y\to X}(y)) - y||$ is the Cycle Consistency Loss. In our implementation, we follow the network architecture of Cycle GAN to train our conditional Cycle GAN except for the modifications described in the next subsections.

3.4 Conditional GAN

Generative adversarial nets can be extended to a conditional model if both the generator and discriminator are conditioned on some extra information y. y could be any kind of auxiliary information, such as class labels or data from other modalities. We can perform the conditioning by feeding y into the both the discriminator and generator as additional input layer.

In the generator the prior input noise pz(z), and y are combined in joint hidden representation, and the adversarial training framework allows for considerable flexibility in how this hidden representation is composed. In the discriminator x and y are presented as inputs and to a discriminative function (embodied
again by a MLP in this case).

The objective function of a two-player minimax game would be as:

$$\min_G \max_D V(D, G) = E_{x\sim p_{data}(x)}[\log D(x|y)] + E_{z\sim p_z(z)}[\log(1 - D(G(z|y)))]. \quad (3)$$

4. Implement details

Our approach is to train an improved conditional Cycle GAN which includes two GANs.

4.1 Network architecture

The architecture of network is as follows:

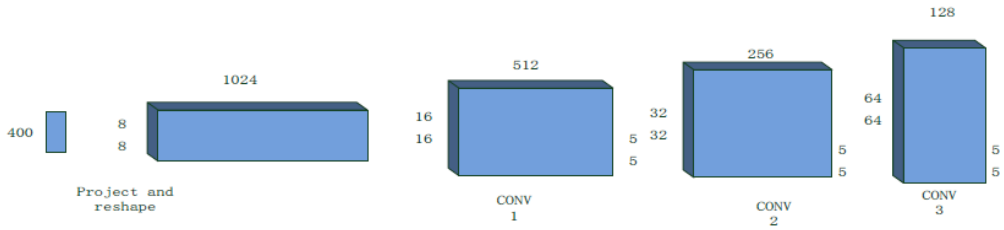


Figure 1: Generative model in DCGAN. A 400 dimensional word2vec Z is projected to a small spatial extent convolutional representation with many features maps. A series of three fractionally-strided convolutions then convert this high level representation into a 64x64 pixel image.

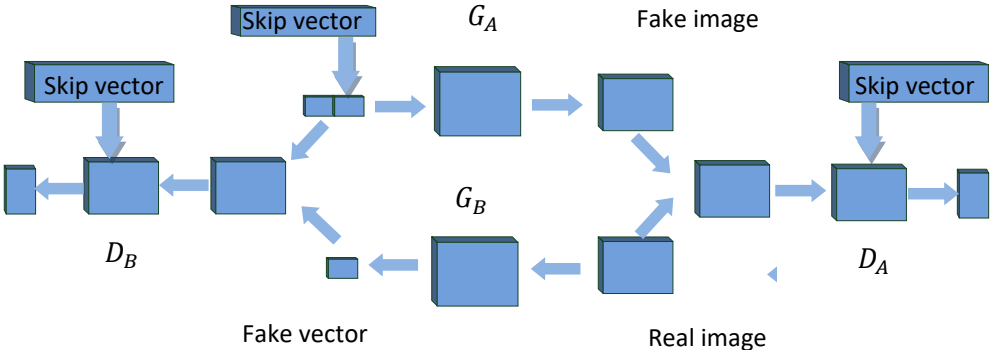


Figure 2: Improved Conditional Cycle GAN structure

There are two GANs in the network, the first part we input 10000 dimentional word2vec features concatenated skip vector into the generator A. The generator would generate 3x64x64 dimensional images via three deconvolutional layers. And then we use the generative images and the real images as the inputs of discriminator A and the skip vector as the condition of discriminator A. The second part we input 3x64x64 real images as input into the generator B. The generator would generate 10000 dimensional word features via three convolutional layers. And then we use the generative word features and word2vec features as the inputs of discriminator B and the skip vector as the condition of discriminator B. As a result, we hope the output of the generator A would be similar to the real image of this description as much as possible.

4.2 Algorithm

Input: minibatch n 10000 dimensional word2vec features, minibatch n 3x64x64 real images, minibatch n 2400 dimensional skip thought vector.

For number of training iterations do

For k steps do

- Sample minibatch of m 10000 dimensional word2vec features from pg(Z)
- Sample minibatch of m real images from data generating distribution pdata(x)

- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(x^{(i)}\right) + \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right) \right]$$

End for

- Sample minibatch of m 400 dimensional word2vec features from pg(z)
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right)$$

End for

The discriminator would label the real images as one, and the generated images as zero. We use binary cross entropy as the loss function. We compute the Euclidean distance between the cycle images and the real images which is corresponding to this description. And then we combine the binary cross entropy of generator with Euclidean distance as the total loss function to update the gradient of generator part. So the stochastic gradient of updating generator formula becomes:

$$\nabla \sum_{\mathrm{i}=1}^{m} (\log\left(1 - D\left(G(z^{(i)})\right)\right) + ||Og - R||)$$

Where the Og represents the output of generator model, and the R represents the real images.

5. Experiment and results

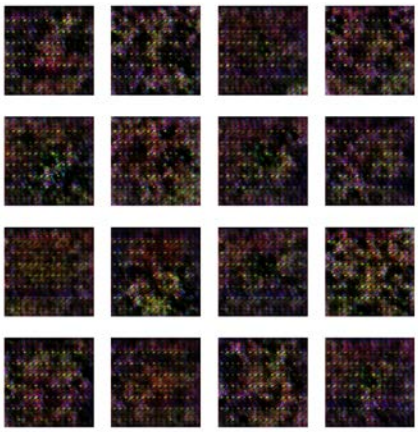
5.1 improved Cycle GAN:

We sum the word2vec features of each word in each sentence as the input of network:

Generative image Real image

As we can see, the results are not good. I still need to improve the performance of this network.

5.2 implement Cycle GAN with mean feature:



Generative image

Real image

5.3 implement Cycle GAN with concatenate feature:

Generative image

Real image

5.4 implement Cycle GAN with 100 penalty on cycle loss with concatenate loss:

Generative image

Real image

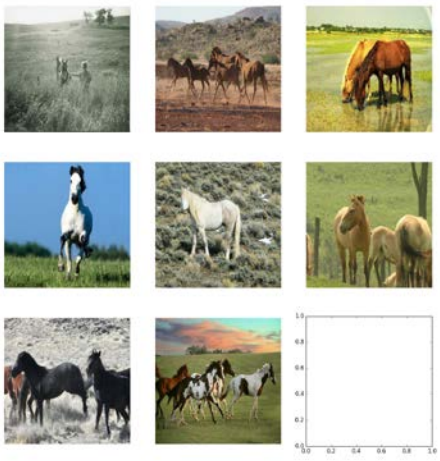
As we can see, the above results are not good, so I need to find out where is the problem coming from. The next step is to reimplement Cycle GAN based on my network.

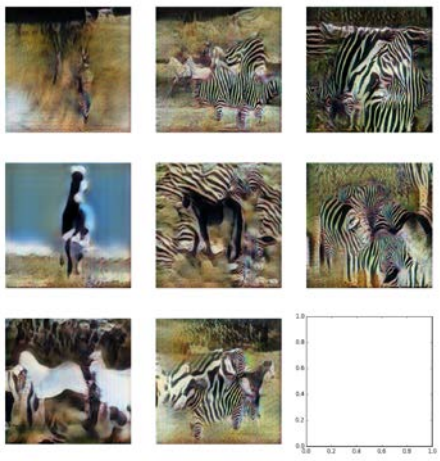
5.5 implement original Cycle GAN using Torch:

Horse

Zebra

5.6 implement original Cycle GAN using Theano:

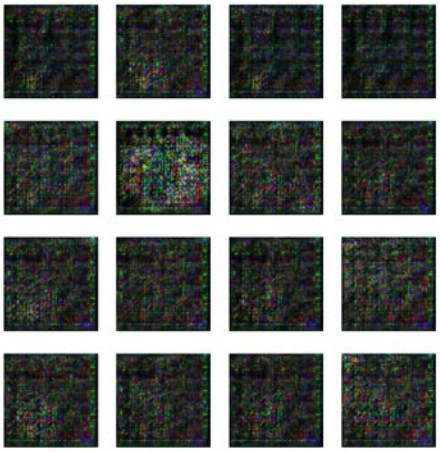




Horse

Zebra

5.7 In order to improve the performance of Cycle GAN, I enlarge the dataset(from 12000 to 72000). The newest results are as the follows:



Generative image

real image

As we can see, the network collapse. I think the reason is that I change the update step. Next, I decide to use some other word features instead of concatenate word2vec features.

5.8 implement the adversarial text to image synthesis using Tensorflow:
the flower shown has yellow anther red pistil and bright red petals:

this flower has petals that are yellow, white and purple and has dark lines:

the petals on this flower are white with a yellow center

this flower has a lot of small round pink petals.

the petals of the flower are pink in color and have a yellow center.

As we can see, this method performance very well.

5.9 improved the performance of cycleGAN using tensorflow.

Horse

Zebra

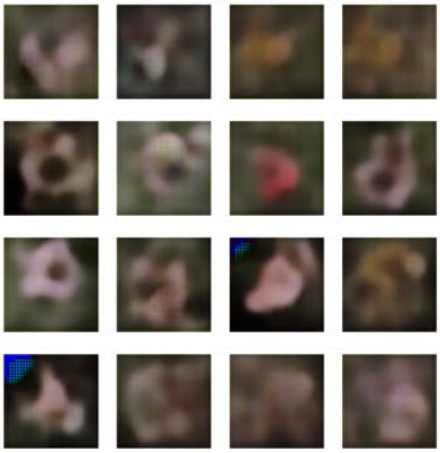
As we can see, the results are better than previous.

5.10 implement improved cycle GAN with 10 times of penalty on cycle loss:

Generative image

real image

5.11 implement improved cycle GAN with 100 times of penalty on cycle loss:



Generative image

real image

5.12 implement improved cycle GAN with 10 times of penalty on cycle loss and without word feature cycle loss:

Generative image

real image

5.13 improved cycle gan:

Generative image

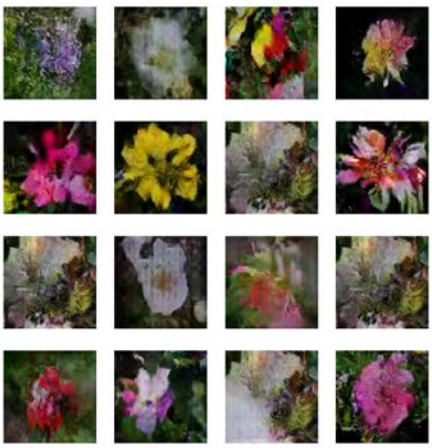
real image

5.14 conditional improved cyclegan(condition in word to image):

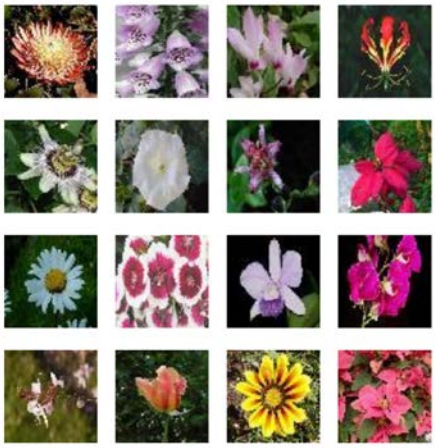
Generative image

real image

5.15 conditional improved cyclegan(condition in both generator):

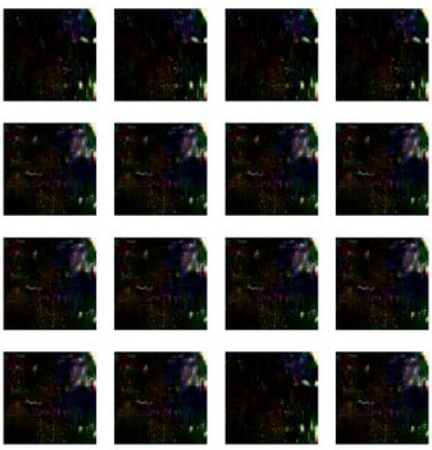


Generative image



real image

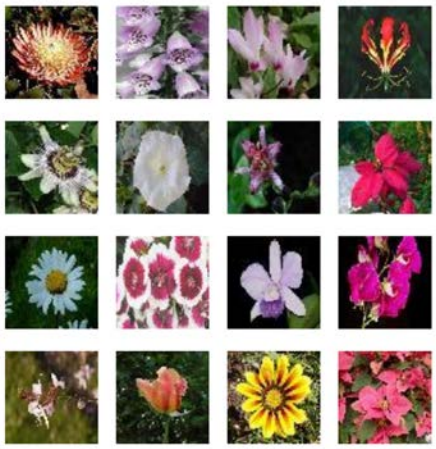
5.16 conditional gan with skip vector:



Generative image

real image

5.17 conditional gan with word2vec:



Generative image real image

As we can see, the results look like so bad. Next, I will improve the performance of network.

5.18 improved conditional gan with skip vector:

Generative image real image

5.19 improved conditional gan with word2vec:

Generative image real image

As we can see, it works now. The results using word2vec are even better than skip vector.

6. Conclusion and future work

In conclusion, we proved that the GNN (generative adversarial neural network), conditional GAN, Cycle GAN works on flower dataset well. And we have combined these three neural network together to see the output of network.

In the next step, we will optimize the performance of final neural network.

7. Reference

[1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B.,Warde-Farley, D., Ozair, S., Courville, A., and Bengio,Y. Generative adversarial nets. In NIPS, 2014.

[2] Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. 2016.

[3] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, Honglak Lee. Generative Adversarial Text to Image Synthesis, 2016

[4] Dosovitskiy, A., Tobias Springenberg, J., and Brox, T. Learning to generate chairs with convolutional neural networks. In CVPR, 2015.

[5] Mansimov, E., Parisotto, E., Ba, J. L., and Salakhutdinov,R. Generating images from captions with attention. ICLR, 2016.

[6] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen. Improved Techniques for Training GANs, 2016

[7] Sergey Ioffe, Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, 2015

[8] Xingjian S H I, Chen Z, Wang H, et al. Convolutional LSTM network: A machine learning approach for precipitation nowcasting[C]. Advances in Neural Information Processing Systems. 2015: 802-810.

[9] Gemici M, Hung C C, Santoro A, et al. Generative Temporal Models with Memory[J]. arXiv preprint arXiv:1702.04649, 2017.

[10] Arjovsky M, Bottou L. Towards principled methods for training generative adversarial networks[C]. NIPS 2016 Workshop on Adversarial Training. In review for ICLR. 2017, 2016.

[11] Ioffe, Sergey and Szegedy, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167, 2015

[12] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv:1703.10593, 2017.

[13] Y. Lu, Y.-W. Tai, and C.-K. Tang. Conditional cyclegan for attribute guided face image generation. arXiv preprint arXiv:1705.09966, 2017.

[14] Mirza, M. and Osindero, S. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784, 2014.