EE 243 Advanced Computer Vision Project Report

Jiaming Zhou
University of California, Riverside
900 University Ave, Riverside, CA 92521
Jzhou010@ucr.edu

Rongjia Zhang
University of California, Riverside
900 University Ave, Riverside, CA 92521
Rzhan102@ucr.edu

Abstract

In this paper we present a summary report on our computer vision course project. The objective of this project was to implement two research papers: Learning Deep Representation of Fine-Grained Visual Description, and Generative Adversarial Text-to-Image Synthesis. The method in the second paper built upon the result of the first. The end product is a model that generates images that match the input description texts. We have tested the models trained by ourselves based on some self-written sentences that describe details of bird and flower images, and the model performs well.

1. Introduction

Generative Adversarial network has been growing in popularity as more people found its potential in practical applications. Company like Nvidia has demonstrated the power of this network by using a newly developed generator structure to produce ultra-realistic faces [3]. While it will be hard for common people to replicate something similar without delving deep into the field of machine learning, some researchers have proposed models that are more user friendly. Two papers, published by Scott Reed and his research colleagues, allow them to develop a GAN model that generates images using on text-based inputs.

The aforementioned task is accomplished by training a deep convolutional generative adversarial network conditioned on description vectors. The description vectors are obtained through training another model that learns a compatibility function of texts and the corresponding images. These two models are covered separately by the "Generative Adversarial Text-to-Image Synthesis" and "Learning Deep Representation of Fine-Grained Visual Description" papers respectively. The main objective of this project is to replicate the results of these two papers, as well as provide some theoretical analysis.

This report will be presented in the following sequence: First each student will briefly explain the paper he or she is responsible for. Then theoretical analysis of the papers will be explained. Lastly, the results of both our individual and collective implementations will be presented along with analysis.

2. Responsibility

Even though this project is completed by two students, most work is done individually. Each student is responsible for one paper. Below the students' individual and collective responsibility are explained.
(note: Each session in this report is written by the student who is responsible for the corresponding paper. e.g. 2.1 is written by Jiaming and 2.2 is written by Rongjia.)

2.1. Jiaming Zhou's Responsibility

Jiaming was mainly responsible for the "Learning Deep Representation of Fine-Grained Visual Description" paper. His job was to understand the paper's content, provide a theoretical analysis, and replicate its results through running its provided codes and implementing a python version. He had decided not to test it on another data set because that would have required him to acquire a set of high-quality fine-grained description.

2.2. Rongjia Zhang's Responsibility

Rongjia's responsibility was mainly about reading the "Generative Adversarial Text to Image Synthesis" paper and trying to reproduce the results suing the given codes. Because it was her first time to do something with the generative adversarial network, she would spend some time on understanding the basic idea of GAN and explaining it clearly in this report. Also, she would have a look at the synthetic images generated from different descriptions on birds and flowers and make some analysis.

2.3. Collective Responsibility

The output of Jiaming's paper will be used during the training of the DC-GAN by Rongjia. The model will not be trained end to end, which means that we will train the text encoder first and then use this pretrained model to train the GAN. We do this because it will accelerate the training speed and save time. In the end, we will test this model together and analyze the generated images.

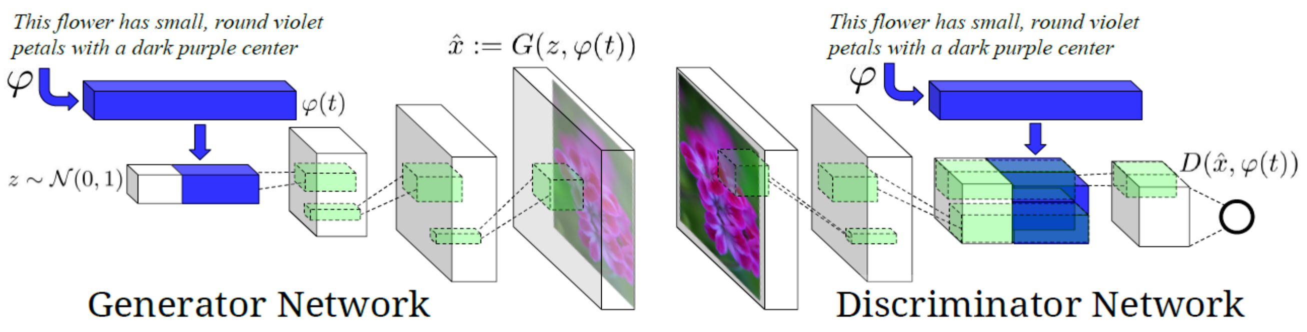

This flower has small, round violet petals with a dark purple center
$\hat{x} := G(z, \varphi(t))$
φ
$\varphi(t)$
$z \sim \mathcal{N}(0,1)$
This flower has small, round violet petals with a dark purple center
φ
$D(\hat{x}, \varphi(t))$
Generator Network
Discriminator Network

Figure 1: Deep Convolution Generative Adversarial Network (source: [2])

3. Background

In this section, we will briefly describe the core content of each paper.

3.1. Learning Deep Representation of Fine-Grained Visual Description

This paper focuses on zero-shot classification and retrieval problems on fine-grained images. Find-grained images are images belonging to multiple different classes that have very subtle differences. Traditional methods in zero-shot learnings utilize additional information, such as hand-crafted attributes and bag of words, to relate images with their labeling information. Promising performances are shown in areas where classes are well separated. However, the author points out that little works have been done in the areas of fine-grain images. This paper proposes to contribute in this area through suggesting a new method to provide side information in additional to fore-mentioned methods, and through proposing a novel model that jointly embed images and their corresponding fine-grained visual descriptions.

The text encoder used for representing fine-grained visual descriptions is a text-based convolution neural network concatenated with a recurrent neural network. Figure 2 displays a graphical representation of the language model.

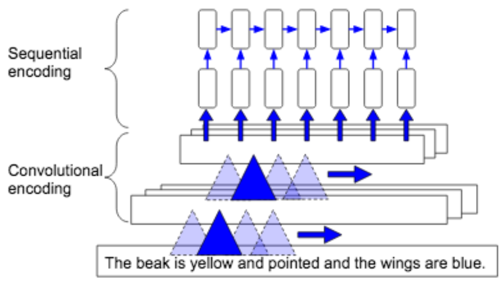


Figure 2: Convolution-Recurrent Net (source: [1])

To train a model that jointly embed the descriptions vector and the corresponding images, the authors propose a model, which they called it the "Deep Structured Joint embedding." This model learns a compatibility function of the images and the texts. The compatibility score is calculated by taking the dot product between the feature vectors of the images and the text. The goal is to minimize the empirical risk which is calculated according formula (1).

$$\frac{1}{N}\sum_{N=1}^{N} \Delta(y_n, f_v(v_n)) + \Delta(y_n, f_t(t_n)) \qquad (1)$$

$v = visual\ information\ (e.g.\ images),$
$t = text\ information\ (e.g.\ description)$
$y = label\ of\ the\ image$
$N = number\ of\ images$

The image and text classifiers are formulated as follow:

$$fv(v) \ = arg\ maxy \in YEt \sim T(y)[F(v,t)]$$

$$ft(t) \ = arg\ maxy \in YEv \sim V(y)[F(v,t)]$$

One unique property of this model is that it is symmetric with respect to the image and text. This provides the benefit of only one of the two functions needs to be trained to optimize the loss function (1) depending on the use of the model. For example, in image classification problems, $f_v(v_n)$ needs to be trained. In contrast, $f_t(t_n)$ needs to be trained when image retrieval is the intended application.

3.2. Generative Adversarial Text-to-Image Synthesis

This paper mainly concentrates on solving two problems. First, how to learn a text feature representation that can describe the dominant visual details in a sentence. Second, how to use these features to synthesize an acceptable result that human cannot distinguish from real images.

For the first problem, they follow the approach by using deep convolutional neural network combined with recurrent neural network. The architecture of this model is explained in detail in the last section. In order to save training time, the pretrained model is directly used to train the generative model. The output of this word embedding is compressed using a fully-connected layer to a small dimension in order to fit the dimension of the input of GAN.

For the second problem, they choose to use the text-conditional deep convolutional GAN, the architecture of which is shown in Figure 1. The output of text encoder is used by both generator and discriminator by concatenating in depth with noise and image feature maps correspondingly.

The generative adversarial networks consist of a generator G and a discriminator D that will compete in a two-player minmax game, which is shown below.

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

The generator is supposed to synthesize a fake image that looks similar to the real one and the discriminator is responsible for distinguish them. The only difference between GAN and conditional GAN is that the CGAN will conditioned on some priors in the input. In this case, the text encoder output is the prior.

4. Theoretical Analysis

This section includes student's individual theoretical analysis of the paper's content.

4.1. Jiaming's Analysis

Zero-shot learning aims to solve image recognition problems of classes that have limited training samples. This is accomplished by complimenting the visual information with other side information. As been mentioned by Reed, much of today's zero-shot learning uses side information that are either too simple–such as bag of words– or impractical for real life applications. Hence, to improve performance in this area of machine learning, Reed has proposed another way to supplement side information that uses natural language as well as an extension to existing structure joint embedding models. In this section, I will provide an analysis on the CNN-RNN text model and the proposed approach for jointly embedding.

4.1.1 Jiaming's Analysis

Reed argues that the reason he chose description text as a way to supplement side information is that it is both flexible and captures the important visual important in a compacted way. It also provides users a familiar way to annotate images, hence the proposal of the CNN-RNN neural language model.

The use of a neural language network provides several benefits. As opposed to traditional language model, neural model has the ability to generalize well. The second benefit is that it captures the semantic meanings of the text. In the embedded space, texts that share similar meanings are closer together. Lastly, neural language models also scale well with increase in dictionary size. Traditional method has dimensionality problem for each word is being represented by distinct vector. As more text are being used for training, the computational cost becomes impractical [4].

Due to the fact that text input is sequential, stacking an RNN on top of the CNN allows the model to capture the contextual meanings of the input text. A downside to simple feedforward neural network is that it does the output does not condition on previous input. While for image processing, this is less problematic as pixels are spatially related. However, for task like speech recognition, a neural network being able to condition on previous calculation is important for accurate prediction [5]. As a result, by concatenating a CNN with an RNN would like this neural language model to not only capture low-level and high level of the fine-grained description texts.

4.1.2 Deep Structure Joint Embedding

In order for a model to reliably classify classes that have not been seen before, a way must be devised to link them to classes that have training samples. The authors of this paper have adapted multimodal structure embedding model to jointly embed images and the corresponding descriptions. Unlike the modal they have reviewed, an improvement has been made in which case the modal is symmetric with respect to the image and text. This provides the benefits of a classifier could still be trained by conditioning on only images or texts.

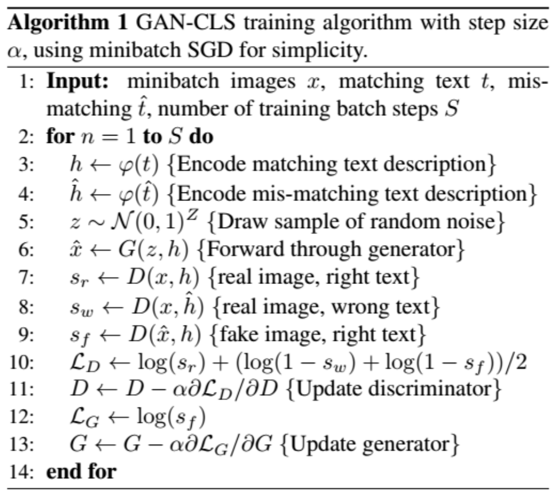
4.2. Rongjia's Analysis

In the text to image paper, they have proposed different methods to train the GAN. Although we do not train our GAN based on all of these methods. I will make some theoretical analysis in this section. These methods include naive GAN, Matching-aware discriminator (GAN-CLS), learning with manifold interpolation (GAN-INT) and inverting the generator for style transfer. The basic concept of naive GAN has already been discussed in the background section, so it will not be mentioned again in this part.

4.2.1 Rongjia's Analysis

For naive GAN, the observations are simply image and text pairs, and the model is trained to discriminate fake images from real ones. Discriminator is responsible for judging image and generator learns to produce images which looks real. However, this naive algorithm has its drawback that the discriminator has no aware of whether real training images match the text encoder input.

Intuitively, we are supposed to separate different kinds of error sources as unrealistic images with any text and realistic images with mismatch conditioning information.



Algorithm 1 GAN-CLS training algorithm with step size α, using minibatch SGD for simplicity.

```
1: Input: minibatch images $x$, matching text $t$, mis-
   matching $\hat{t}$, number of training batch steps $S$
2: for $n = 1$ to $S$ do
3:     $h \leftarrow \varphi(t)$ {Encode matching text description}
4:     $\hat{h} \leftarrow \varphi(\hat{t})$ {Encode mis-matching text description}
5:     $z \sim \mathcal{N}(0,1)^Z$ {Draw sample of random noise}
6:     $\hat{x} \leftarrow G(z,h)$ {Forward through generator}
7:     $s_r \leftarrow D(x,h)$ {real image, right text}
8:     $s_w \leftarrow D(x,\hat{h})$ {real image, wrong text}
9:     $s_f \leftarrow D(\hat{x},h)$ {fake image, right text}
10:    $\mathcal{L}_D \leftarrow \log(s_r) + (\log(1-s_w) + \log(1-s_f))/2$
11:    $D \leftarrow D - \alpha\partial\mathcal{L}_D/\partial D$ {Update discriminator}
12:    $\mathcal{L}_G \leftarrow \log(s_f)$
13:    $G \leftarrow G - \alpha\partial\mathcal{L}_G/\partial G$ {Update generator}
14: end for
```

Figure 2: GAN algorithm

The algorithm 1 shown above summarizes the training process. s means the score from discriminator for different kinds of input pairs. The parameter of discriminator and generator are updated using gradient descent with step size α. One thing to mention is that, we do not update the parameters for discriminator and generator simultaneously, instead we update the discriminator first and then update generator with fixed discriminator parameters.

4.2.2 GAN-INT

With limited training data, it is intuitive to think about generating a large amount of text embeddings by interpolating between embeddings of the training captions. The interpolation does not need to have real corresponding image and text pairs. The loss can be expressed as below.

$$\mathbb{E}_{t_1,t_2\sim p_{data}}[\log(1 - D(G(z, \beta t_1 + (1-\beta)t_2)))]$$

t1 and t2 are two real text embeddings and β is a parameter between 0 and 1 which will control where the synthetic embedding will be between the two real data. In practice, t1 and t2 can even from different images or categories and the model still works well.

4.2.3 Inverting the generator for style transfer

An image should contain not only main object but also the background color and pose. The text encoder result will include main object details, so it obvious that we can contain background and pose information in the noise vector. In this case, we can wish to transfer the background of one image to other images.

To achieve this, we are supposed to train a CNN which can invert from sample images back into noise vectors. The loss is shown below.

$$\mathcal{L}_{style} = \mathbb{E}_{t,z\sim\mathcal{N}(0,1)}||z - S(G(z,\varphi(t)))||_2^2$$

S is a style encoder net and we want the output of this encoder is as close as possible to the real input noise.

5. Implementation Results

5.1. Text Encoder

In this section we will describe the results of the first paper's implementation. We were to able run the provided codes on both the CUB-200 bird and Oxford-102 Flower database. However, we were unable to implement it in python due to multiple challenges that are going to be explained in the challenge section.

For each database, we ran the provided codes with the following hyper parameters:

Hyper Parameters	
Initial Learning Rates	0.007
Max epochs	200
Batch size	40
# of caption/ Img	10
Optimizer	RMSprop

Table 1: Hyper Parameters for Training

Due to the symmetry property of the model, we have for each dataset trained two encoders; One with symmetric option enabled, and the other without. When the symmetric option is enabled, the capability function for texts is trained. Tabel 2 summarizes the results of the evaluations of the trained models.

Accuracy (%)	Image Classification		Image Retrieval	
	Sym	Asym	Sym	Asym
CUB	54.69	52.57	45.52	4.72
Flower	60.09	59.91	55.10	10.10

Table2: Implementation Results

Accuracy (%)	Image Classification		Image Retrieval	
	Sym	Asym	Sym	Asym
CUB	56.8	54.3	48.7	4.8
Flower	65.6	60.9	59.6	7.6

Table 3: Paper's Results

While the results do not exactly match with the results reported by the authors, the overall pattern is similar. The differences could be caused by the choices of the hyper parameters. The low accuracy of the asymmetric model for image retrieval makes sense as the capability function of the text encoder is not trained when symmetric option is turned off during training.

5.2. DC-GAN

In this experiment, we will present some results from the model trained by ourselves with descriptions for birds, because we just used the text encoder trained on CUB dataset to generate inputs for GAN. The input image size is set to 64×64×3 and the output of the text encoder is projected from 1024 to 128 dimensions for both generator and discriminator. No quantitative results will be given in this part, as it hard to score how close the relation is between the images and corresponding sentences.

Only three groups of generative results from self-written sentences are shown below. The three captions describe three kinds of bird in details like the color of the belly and shape of the beak.

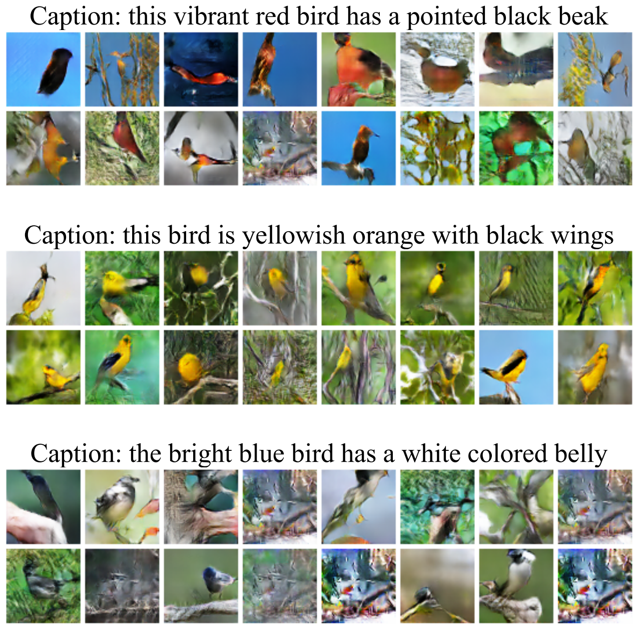

Caption: this vibrant red bird has a pointed black beak
Caption: this bird is yellowish orange with black wings
Caption: the bright blue bird has a white colored belly

Figure 3: Generated Bird Images

If we take a close look at these images, it is not difficult to distinguish them from real ones, and some are just like random combination of color blocks. However, it is hard to tell the difference from just a glance. We can safely say that the text encoder captures the important features well, like the color of feather and beaks. Even the backgrounds are generated closely to real sky or branches patterns by the GAN.

5.3. Challenges

As been mentioned before, we were unable to complete a python implementation of the papers. The biggest challenge when attempting to implement the papers by ourselves was converting theories into codes. With limited coding knowledge, we were only able to get some parts working, such as the data loader.

Even with the given codes, getting the environment set up was also very difficult. Some packages were not compatible. In multiple occasions, we had to revert some packages back to their older version, which were used at the time when the papers were published. Additionally, we have not been exposed to GAN and text encoder before, it takes us a lot of time to understand the theory part and figure out what is going on during the training and testing process.

References

[1] Reed, S., Akata, Z., Lee, H., and Schiele, B. Learning deep representations for fine-grained visual descriptions. InCVPR, 2016.

[2] Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., Lee, H. Generative Adversarial Text to Image Synthesis. In ICML, 2016.

[3] Karras, T., Laine, S., Aila, T., A Style-Based Generator Architecture for Generative Adversarial Networks, In CVPR, 2019.

[4] Kim, Y., Jernite, Y., Sontag, D., M. Rush, A., Character-Aware Neural Language Models, In AAAI, 2016.

[5] Roell, J., & Roell, J., Understanding Recurrent Neural Networks: The Preferred Neural Network for Time-Series Data. 2017

[6] 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.