Image-to-Image Translation Task-2

Animesh Sharma
animesh.sharma@rutgers.edu
Rutgers University
Department of Computer Science

Harish Udhaya Kumar
harish.udhayakumar@rutgers.edu
Rutgers University
Department of Computer Science

Jaini Patel
jp1891@scarletmail.rutgers.edu
Rutgers University
Department of Computer Science

Karan Pardasani
karan.pardasani@rutgers.edu
Rutgers University
Department of Computer Science

***Abstract*— In the previous task of this project, we focused on implementing the basic model of Generative Adverserial Network and doing quantitative analysis of the model. In this part of the project, we explored two quantitative measures namely Inception Score (IS) and Frechet Inception Distance (FID) score. In addition to this, we have also implemented the Pix2Pix, a conditional Generative Adverserial Network, to train paired images.**

I. Introduction

In this task, we are going to continue the first task and quantitatively analyse the Generative Adverserial Network we have implemented to generate images from Pizza Dataset. We are going to use the two scores to quantify the performance of the Generative Adverserial Network:

1) Inception Score(IS): This score involves using Inception model - a pre-trained deep learning neural network model that will be used to classify the generated images. The Inception score is the exponentiation of the value of the Expectation of KL divergence value between the $p(y|x)$ and $p(y)$, where $p(y|x)$ is the conditional label distribution of the output image, y is the set of labels and x is the image. Also, $p(y)$ is the marginal distribution. The Inception score can be calculated using the following formula:

$$IS(G) = e^{E_{x \in p_g} D_{KL}(p(y|x)||p(y))} \tag{1}$$

where, $x \in p_g$ says that x belongs to p_g dataset. D_{KL} represents the KL divergence. Also, $p(y)$ is the marginal class distribution which is calculated as:

$$p(y) = \int_x p(y|x) p_g(x) \tag{2}$$

2) Frechet Inception Distance (FID): It is the score that measures the distance between the feature vectors calculated for Generated images and Real images. The FID sco can be calculated as:

$$FID(x, g) = ||\mu_x - \mu_g||_2^2 + Tr(\sum_x + \sum_g - 2(\sum_x \sum_g)^{1/2}) \tag{3}$$

where, μ_x and $\sum_x$ are the mean and covariance of the samples and μ_g and $\sum_g$ is the mean and variance of the data distribution.

Among the two scores discussed above, FID score is better than Inception score due to the following reasons:

1) FID score is more immune to noise as compared to inception score.
2) Inception Score has the problem of intra-class mode dropping i.e. IS will give perfect score even if the model generates one image per class but FID score penalises in this scenario.

In addition tot his, we are also implementing **Conditional GAN for Pix2Pix image translation**. In the previous project, we implemented a Generative Adverserial Network that is fed a random noise vector and it generates images that are extracted from a learned data distribution. It was able to generate images that looks similar to the ones present in the dataset. In this task, we are exploring the paired image to image translation.

In Image to Image Translation, we are translating images from one domain to another by training the model to understand the relation or mapping between the input and output images.

There are two types of Image to Image translation: Paired or Unpaired Image to Image translation. In paired image to image translation, the input image and the ground truth images are related to each other. The network is trained to learn the mapping between the paired images and then we can create the output images from the new input images using the model trained on the training dataset.

There are many tassk that we can perform using Paired Image to Image Translation:

1) Convert a sketch into a drawing
2) Convert a Black and White image to Colored image
3) Convert a street view to Aerial View

In this task, we are going to implement and train a Paired Image to Image Generative Adverserial Network or Pix2Pix network that accomplishes task 3 i.e. given a street view image the model will generate the aerial view of the image and given the aerial view of the image, the model will generate the street

view of the image.
The characteristics of Pix2Pix GAN is the following:

1) The generator receives an image and it outputs a translated version of the input image.
2) The discriminator is a conditional discriminator which receives a real or fake image which is conditioned on the same input image that is received by the generator,
3) The discriminator classifies whether the image is real or not.
4) The goal of the Pix2Pix GAN is to fool the discriminator, such that the generator learns how to translate the images.

II. Related Work

The I2I conversion are generally computed by pixel-2-pixel comparisons. The computations assume the output space to be unconstrained and hence the loss that is obtained is "unstructured". However, in CGAN (Conditional GAN), the output and target are strictly differentiated.
Two-domain I2I: We have qualitatively and quantitatively varying models i.e. pix2pix , BicycleGAN , CycleGAN , U-GATIT , GDWCT , CUT and MUNIT in single-modal and multi-modal setting respectively.
Multi-domain I2I We qualitatively and quantitatively compare StarGAN, AttGAN, STGAN, DosGAN and StarGANv2 in single-modal and multimodal setting respectively.
On the basis of observations on common practice, supervised approaches usually produce better translated results than unsupervised approaches on similar network structure. However, in some special cases, supervised methods do not always perform better than unsupervised methods. It is difficult to declare that one algorithm has an absolute superiority over the others. Besides model design itself, there are still many factors influencing the performance, such as training time, batch size and iteration times, FLOPs and number of parameters, etc.

III. Method

In this task, we are following the model of the Generator and Discriminator as described by the author in [1]. We have implemented the model in PyTorch and we have followed the article [2]. The architecture of Pix2Pix GAN is inspired from the Conditional GAN, which works as follows:

- The architecture consists of two networks: Generator and Discriminator. Both the networks are trained such that the Generator tries to mimic the real data distribution and the Discriminator network classifies the real images from the fake images.
- After the training of the generator is complete, the model is able to output similar images when we input a random noise as the input.
- The difference between normal and conditional GAN is that the generator was fed random input conditional on the label of the input image and then for different types of random input, it will generate different images from that class label.

The model follows the following training process to learn the translation of the input image - given an input image x:

1) The generator is trained to translate x to the generated photo $G(x)$ which should be similar to the real photo y.
2) The discriminator D learns to classify
   a) the generated photo $G(x)$ conditioned on the input as fake
   b) the real photo y conditioned on input x is real.

Also, the author mentions that to bring variety in the output of the generator, instead of using Random noise as input, they add Dropout in the Generator Network.

A. Generator

The Generator model used in Pix2Pix GAN is called UNet-Generator model. The model of UnetGenerator is similar to the encode-decoder architecture with skip connections between mirrored layers. The characteristics of UnetGenerator is as follows:

1) The output y is the translated and conditioned version of x.
2) The skip connections are used to retain the information that was lost during down-sampling. Also, the skip connections helps void the vanishing gradient issue in back propagation.

1) Loss Function: The Loss Function that is used for Generator is as follows:
L-1 Loss : $\sum_{i=1}^{n}$ ‖Generated Output - Target Output‖
The total generator loss would be:
Generator Loss = BCE Loss with Real Labels + λ·L-1 Loss

B. Discriminator

The discriminator for Pix2Pix GAN is called PatchGAN, which produces an output of 30×30 matrix. Each cell of the matrix represents the probability of a 70×70 patch being real or fake. The model for the PatchGAN architecture is as follows:

- The discriminator model uses the standard Convolution-BatchNormalization-ReLU blocks.
- The network will give the output of the shape 30×30 which represents whether each 70×70 patch of the input image is real or fake.
- The value of 0 represents that the image is fake and the value of 1 represents that the image is real.

1) Loss Function: We use the Batch Cross-Entropy Loss Function. We need to slow down the rate at which the discriminator learns as compared to the generator, the authors has advised to divide the loss by 2. So, the total discriminator loss is the (BCE Loss for Real Images + BCE Loss for Generated Images)/2.

IV. Datasets and Pre-processing

We used the Dayton dataset from which we select 76,202 images and create a train/test/val split of 55,000/21,048/144

Fig. 1. Overhead Images present in the dataset

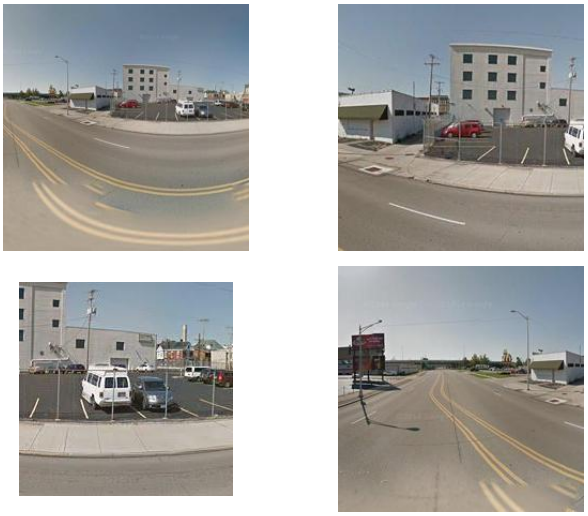

Fig. 2. Street View Images present in the dataset

pairs. We convert the image pixels to Numpy array and append two images (aerial and it's respective streetview) from dataset into one image. The resultant image has both the input and ground-truth images concatenated along the width dimension. The image is then taken as input, we do slicing and indexing to create the input image. The (on horizontal axis) left half of the image is assigned to source and right half to target(ground truth). We resize both the images from 256×256 to 286 ×286, using the opencv resize method, with nearest-neighbor interpolation. The resized images are horizontally stacked with each other and finally cropped back to 256 × 256 pixels. We swap the channel measurements first channel and last channel, in PyTorch model and convert the NumPy array to a PyTorch tensor. We pass the training directory and preprocessing transform function to the ImageFolder, which is passed to the DataLoader function.

V. Evaluation

We have calculated the FID score and Inceptions core using the open source libraries defined in [3] and [4].

A. Quantitative Evaluation of GAN trained in Pizza Dataset

The first part of this task is to generate the FID score and Inception score. We trained the graph for last 40 epochs for real and synthetic pizza dataset. We trained it for lower epochs due to time constraint.

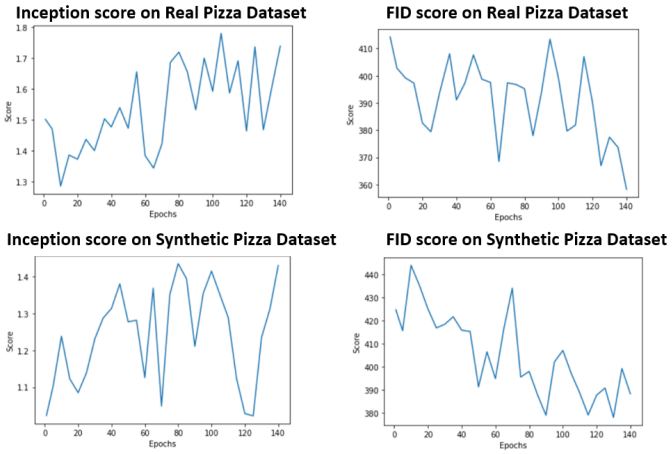

Fig. 3. Quantitative evaluation metrics for Pizza dataset. For

From the graphs, we can note that the FID and Inception scores are oscillating a little but we can say that the Inception score is increasing and the FID score is decreasing. Both these trends indicate that the model is improving it's performance as the epochs increases.

Since, the scores are oscillating from low to high and high to low, we can say that in the performance of the Generator is oscillating between good and bad in consecutive epochs. This can be interpreted as the Generator is winning in some of the rounds and is losing for some of the rounds. This shows that the performance of the Generator is in the equilibrium and out model is not collapsing.

Also, from the graphs we can also see that the FID scores of Real Pizza is higher than FID scores of synthetic pizza. Also, the Inception Score of the Synthetic Pizza is higher than the Inception Score of Real Pizza. This shows that the model is generating closer to original images from synthetic pizza datasets than real pizza dataset.

B. Qualitative Analysis of Pix2Pix GAN with Dayton Dataset

In this task, we are training two models here one model that maps the overhead view to street view and the other model trains to map from street view to overhead view. We trained both the models for 150 epochs using batch size of 128.

1) GAN that maps street view to overhead view: From the images we can analyse that the generated images are very similar to the real images and it is very hard to distinguish the real images and the fake images. We can see from the images that the GAN has learned to create roads, houses, trees and forest but it is really difficult for the GAN to generate smaller objects like cars. Also from the images, we can see that it is difficult to generate curves in the road. The images are slightly distorted towards the curve.

Fig. 4. GAN that generates Overhead Images

2) GAN that maps overhead view to street view: Also, from the images that are generated from GAN has learned to generate cars, similar colours and the shape of tree and leaves. Also, the intersection of the roads are more well defined and the curves are more smooth.

Fig. 5. GAN that generates Street View Images

C. Qualitative Evaluation of Pix2Pix GAN on small batch size

As discussed with the TA and to mimick the parameters used by the author, based on the available resources we have also trained the Pix2Pix model in batch size of 6 images. Due to low availability of resources we were able to train the GAN model for low number of epochs.

D. GAN that maps Street View to Overhead

Due to low availability of resources we were only able to train the GAN model upto 7th epoch. The GAN was able to learn the shape of the trees and the roads. The GAN has not yet learned to generate the shape of the roof of the houses and cars. The model has learned to fill similar colors and build the outline of the terrain. The images in figure 6 are of 7th epoch of the Pix2pix GAN:

Fig. 6. Street view to Overhead view Images generated by Pix2Pix GAN when trained in small batches

E. GAN that maps Overhead to Street View

We were able to train this model for 25 epochs The GAN recognizes trees, roads and structures of houses in the image. The generated image also recognizes the structure of streetlight. The images in figure 7 are of 25th epoch of Pix2pix GAN.

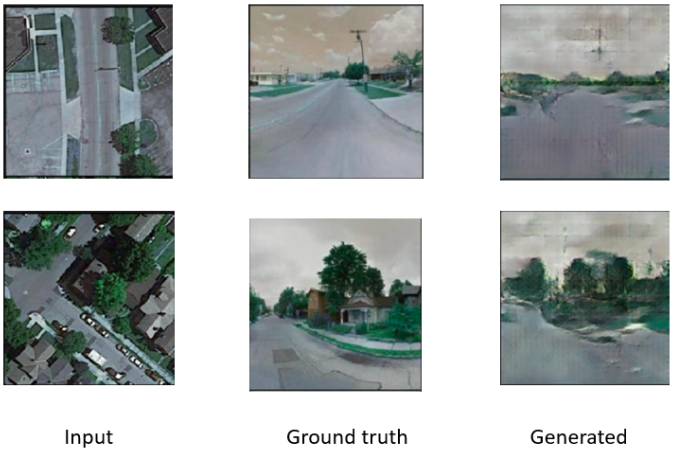

Fig. 7. Overhead view to street view for Dayton dataset when trained in small batches

VI. A. Quanitative Analysis of Pix2Pix GAN on Dayton Dataset

As mentioned in the report, we have calculated the FID and Inception score for Pix2Pix GAN on Dayton dataset. Below is the plot for the Inception Score and FID score for training on the GAN that maps from Aerial (Overhead View) to Street View. We have calculated the FID score for every epoch and we have trained the model for 140 epochs. From the trends in FID score, we can see that the FID score gradually decreases as epochs increases which in turn indicates that GAN is learning from the dataset. The images that the Generator is producing are closer to the ground truth images in training images.

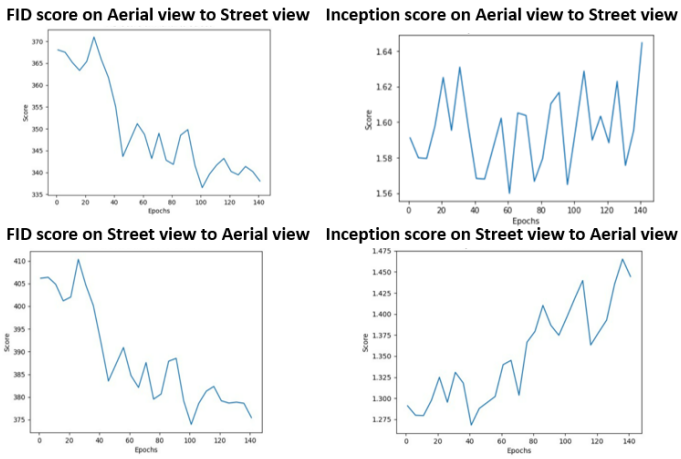

Fig. 8. Quantitative evaluation metrics for Dayton dataset from aerial view to street view

Also, we trained the same GAN model to map the images from Street View to Aerial (Overhead View). We got the similar trend of FID score and Inception score from this model. From the above graphs we can infer that the FID score is decreasing and the Inceptions core is increasing which means that the GAN is learning. Also, due to similar reasons described in the report we can conclude that the GAN model will not collapse.

Also, we can see that the FID score of **Street View to Aerial View** is lower than the FID score of **Aerial View to Street View**. Also better inception score is for **Street View to Aerial View**. So we can say that the model performs better for the **Street View to Aerial View**.

VII. References

1) P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1125â 1134, 2017.
2) https://learnopencv.com/paired-image-to-image-translation-pix2pix/
3) https://github.com/mseitzer/pytorch-fid
4) https://github.com/sbarratt/inception-score-pytorch
5) N. N. Vo and J. Hays, âLocalizing and orienting street views using overhead imagery,â in European conference on computer vision, pp. 494â509, Springer, 2016.
6) K. Regmi and A. Borji, âCross-view image synthesis using conditional gans,â in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3501â3510, 2018