Paired and Unpaired Image to Image Translation (Group 32)

Gaurav Kumar (gkumar@ucsd.edu)
Soham Satyadharma (ssatyadh@ucsd.edu)
Harpreet Singh (h1singh@ucsd.edu)

Background

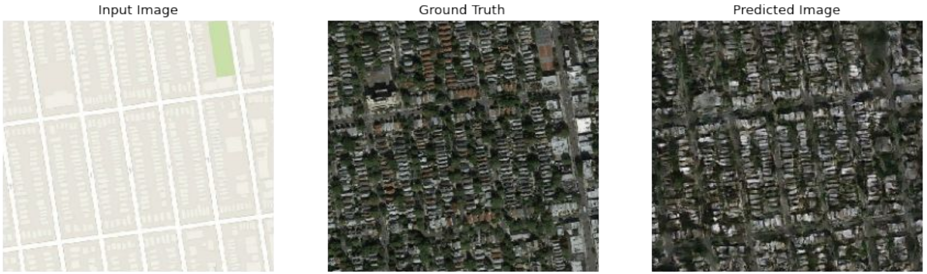
- Image to Image Translation is an active area of research in Computer Vision
- Transforms an image from one domain to another domain
- **Goal** is to learn a mapping function between input and output image that has similar characteristics but different styles
- **Real world Usage:**
  - Helps generate data for low resource domains having less data
  - Colorization of black and white images
  - Restoration of old degraded images
  - Converts satellite images to google map view
- Utilizes recent techniques in Generative Adversarial Networks (GANs) [1] to achieve this task.
- GANs are a zero-sum-game between the generator and a discriminator

Background

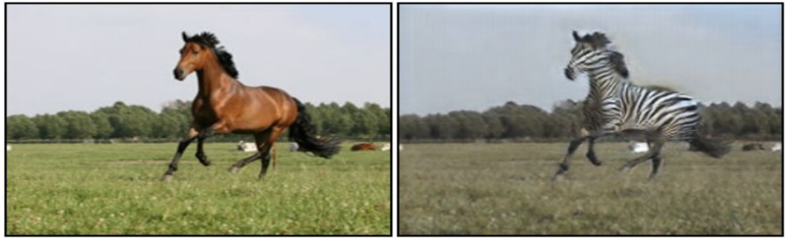
Problem Statement: Utilize different applications of GANs for image to image translation task. Input and output images contains same inherent characteristic features, but differ in domain.

- Paired Image to Image Translation Task
  - Data contains one-to-one mapping of input and output image (paired examples)
  - Image level supervision for the ML problem
- Unpaired Image to Image Translation Task
  - One-to-One mapping of domain image data not available.
  - Contains Domain level supervision with data corpus from both the domains.

Paired image-to-image translation - map dataset



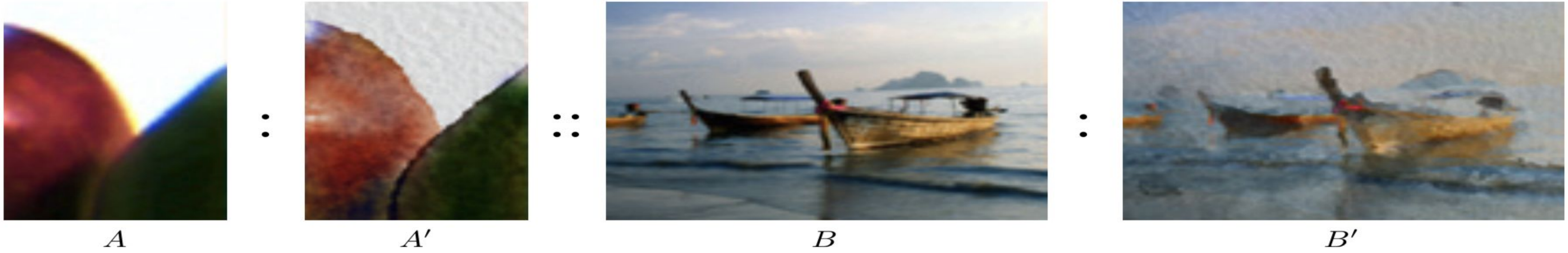
Unpaired Image-to-Image translation - Horse to Zebra



Literature Survey

Image Anomalies [2] - Paired Image-to-Image Translation

- Filters learnt using autoregression algorithm from pairs of unfiltered and filtered images
- Learned features used to generate the analogous image for the target input image
- Similarity metric based on approximation of a Markov random field model using raw pixel values and steerable filter responses used as image features
- Trained model used for image super-resolution, texture transfer, artistic transfer, texture by numbers

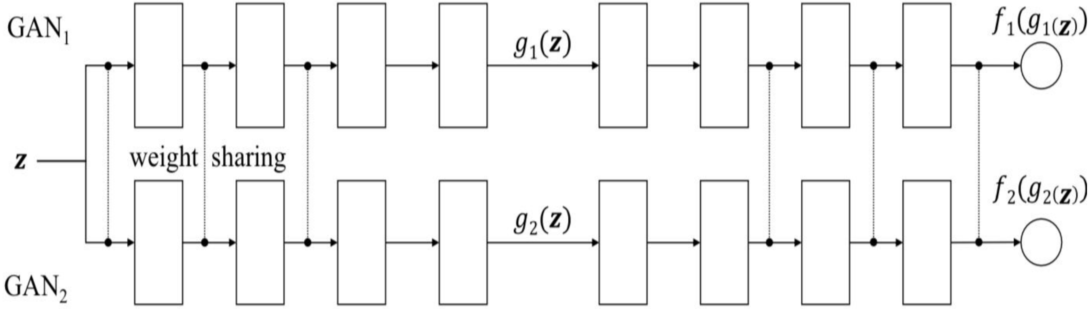


A,A’ are the training pair of images, B is the target image and B’ is the generated image of B analogous to A,A’

Literature Survey

Coupled Generative Adversarial Networks (CoGAN) [3] - unpaired Image-to-Image Translation

- Learn joint distribution of multi-domain images without using labelled training image pairs.
- Set of images are drawn separately from the marginal distributions of the individual domains
- For two image domains -
  - a pair of GANs are used, each generating images from one domain.
  - The generator and discriminator networks of both the GANs have few shared weights and trained in minimax fashion

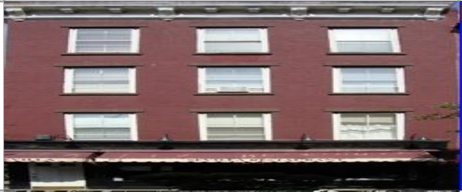
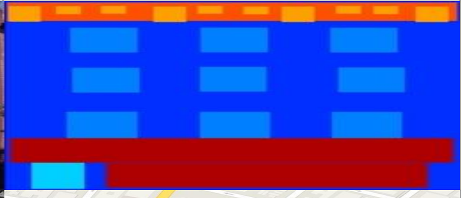
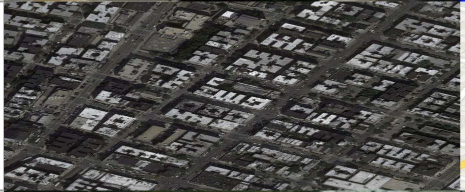
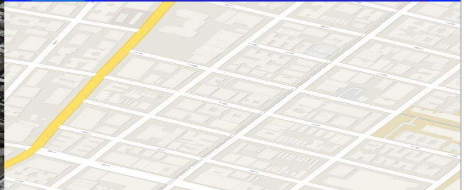
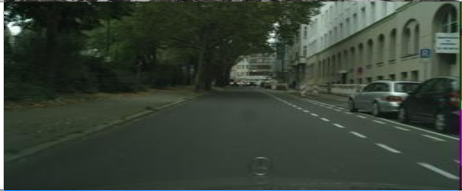
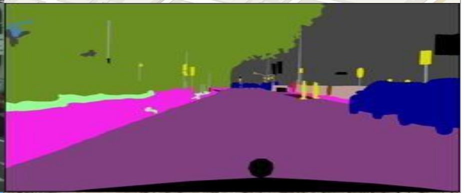


For every pair, left image is input and the right image is the output of CoGAN

Necessity of deep learning

- Complex non linear relationships need to be learnt between the two image domains.
- Difficult to hand-engineer features for complex intricacies in data.
- There is no one size fits all traditional computer vision method that can capture the relationships between so many image domains.
- Deep learning helps learn such relationships
- Different models need to be learnt for different datasets.

Datasets

Dataset Name	Number of images	Resolution	Domain A Image	Domain B Image
Facades [4]	606	256x256		
Maps [5]	2194	600x600		
Cityscapes [6]	3475	256x256		
Horse to Zebra (Unpaired task) [7]	2661	256x256		

Feature extraction and Preprocessing

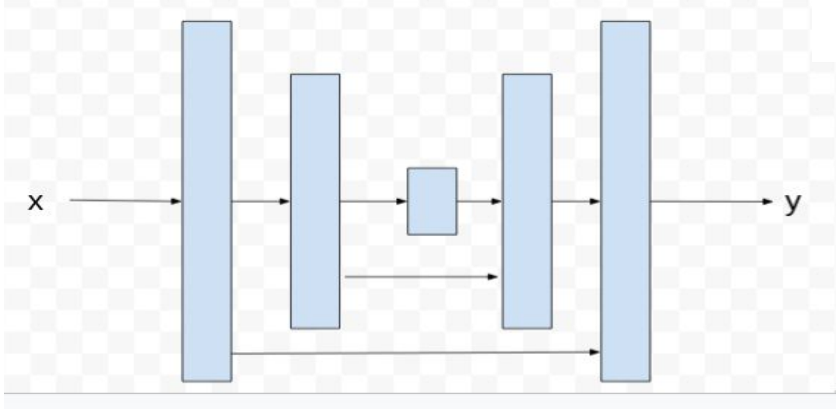
- No explicit hand-engineered features are created.
- Datasets consists of source and target images and the Deep Convolutional Neural Network based models are used which learns descriptive features from preprocessed images.
- The images are preprocessed before passing through the network.
- **Preprocessing steps**
  - random flipping - horizontally flip the image
  - random cropping - randomly cropping the input image for the desired size
  - random jittering - image extrapolation using nearest neighbor technique followed by random cropping
  - Normalization - scaling the image values to [0,1]

Model A - (paired Image-to-Image translation)

Conditional GAN Model [5]

- U-Net based Conditional Generative Adversarial Networks(GAN) is created using skip connections from encoder network activations to the decoder network input.
- The loss objective is the combination of conditional GAN loss and L1 loss.
- The model is trained by alternating between training Generator and Discriminator Network

U-Net architecture based Generator network



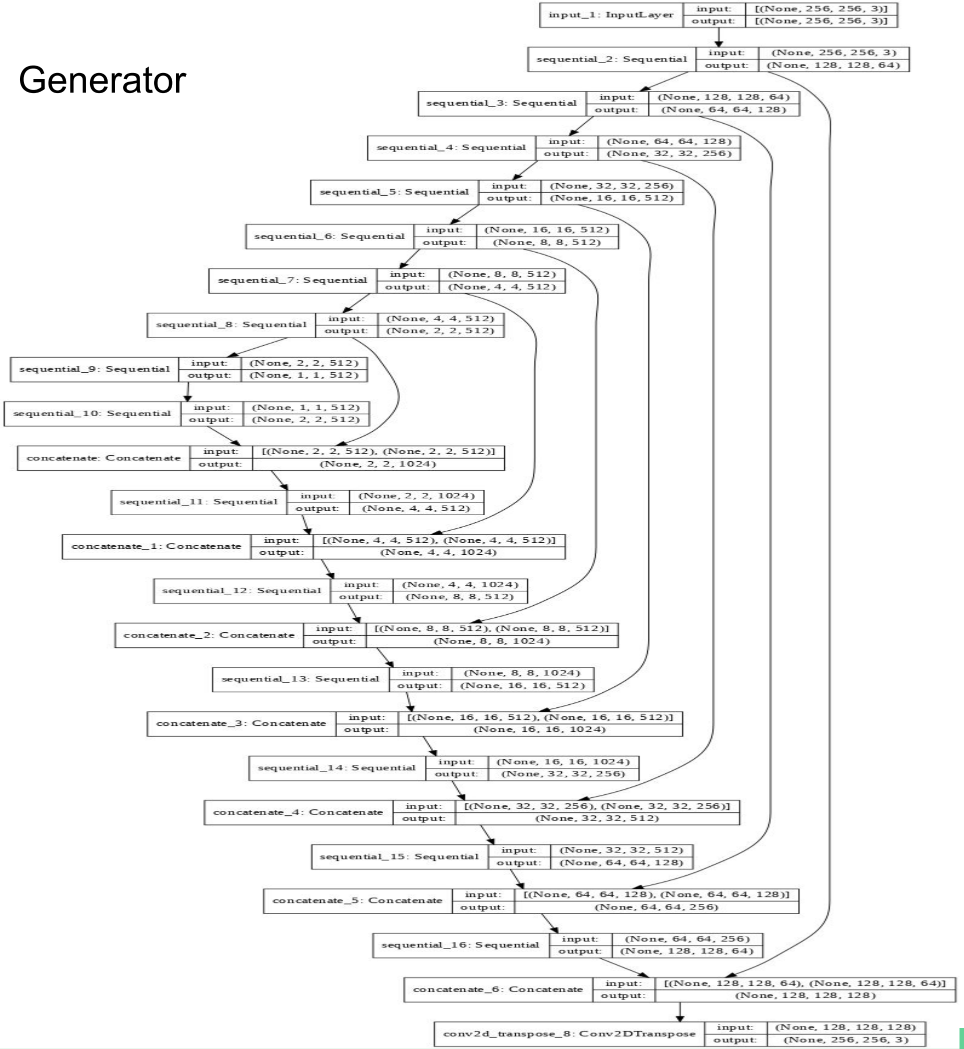
$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z))]$$

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[||y - G(x, z)||_1]$$

$$G^* = \arg \min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G)$$

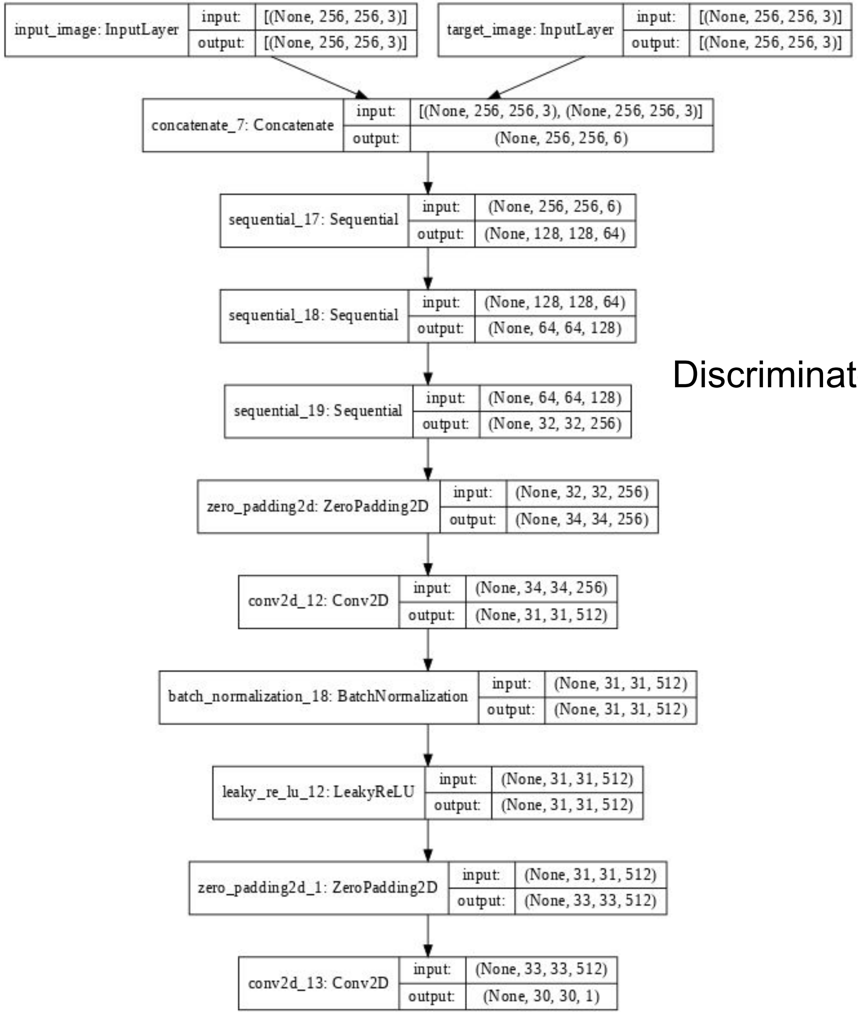
x,y are the input & target images, z is the random noise, G(x,z) is the generated image, D is the output of the discriminator network

Generator



input_1: InputLayer — input: [(None, 256, 256, 3)] — output: [(None, 256, 256, 3)]
sequential_2: Sequential — input: (None, 256, 256, 3) — output: (None, 128, 128, 64)
sequential_3: Sequential — input: (None, 128, 128, 64) — output: (None, 64, 64, 128)
sequential_4: Sequential — input: (None, 64, 64, 128) — output: (None, 32, 32, 256)
sequential_5: Sequential — input: (None, 32, 32, 256) — output: (None, 16, 16, 512)
sequential_6: Sequential — input: (None, 16, 16, 512) — output: (None, 8, 8, 512)
sequential_7: Sequential — input: (None, 8, 8, 512) — output: (None, 4, 4, 512)
sequential_8: Sequential — input: (None, 4, 4, 512) — output: (None, 2, 2, 512)
sequential_9: Sequential — input: (None, 2, 2, 512) — output: (None, 1, 1, 512)
sequential_10: Sequential — input: (None, 1, 1, 512) — output: (None, 2, 2, 512)
concatenate: Concatenate — input: [(None, 2, 2, 512), (None, 2, 2, 512)] — output: (None, 2, 2, 1024)
sequential_11: Sequential — input: (None, 2, 2, 1024) — output: (None, 4, 4, 512)
concatenate_1: Concatenate — input: [(None, 4, 4, 512), (None, 4, 4, 512)] — output: (None, 4, 4, 1024)
sequential_12: Sequential — input: (None, 4, 4, 1024) — output: (None, 8, 8, 512)
concatenate_2: Concatenate — input: [(None, 8, 8, 512), (None, 8, 8, 512)] — output: (None, 8, 8, 1024)
sequential_13: Sequential — input: (None, 8, 8, 1024) — output: (None, 16, 16, 512)
concatenate_3: Concatenate — input: [(None, 16, 16, 512), (None, 16, 16, 512)] — output: (None, 16, 16, 1024)
sequential_14: Sequential — input: (None, 16, 16, 1024) — output: (None, 32, 32, 256)
concatenate_4: Concatenate — input: [(None, 32, 32, 256), (None, 32, 32, 256)] — output: (None, 32, 32, 512)
sequential_15: Sequential — input: (None, 32, 32, 512) — output: (None, 64, 64, 128)
concatenate_5: Concatenate — input: [(None, 64, 64, 128), (None, 64, 64, 128)] — output: (None, 64, 64, 256)
sequential_16: Sequential — input: (None, 64, 64, 256) — output: (None, 128, 128, 64)
concatenate_6: Concatenate — input: [(None, 128, 128, 64), (None, 128, 128, 64)] — output: (None, 128, 128, 128)
conv2d_transpose_8: Conv2DTranspose — input: (None, 128, 128, 128) — output: (None, 256, 256, 3)

Discriminator



input_image: InputLayer — input: [(None, 256, 256, 3)] — output: [(None, 256, 256, 3)]
target_image: InputLayer — input: [(None, 256, 256, 3)] — output: [(None, 256, 256, 3)]
concatenate_7: Concatenate — input: [(None, 256, 256, 3), (None, 256, 256, 3)] — output: (None, 256, 256, 6)
sequential_17: Sequential — input: (None, 256, 256, 6) — output: (None, 128, 128, 64)
sequential_18: Sequential — input: (None, 128, 128, 64) — output: (None, 64, 64, 128)
sequential_19: Sequential — input: (None, 64, 64, 128) — output: (None, 32, 32, 256)
zero_padding2d: ZeroPadding2D — input: (None, 32, 32, 256) — output: (None, 34, 34, 256)
conv2d_12: Conv2D — input: (None, 34, 34, 256) — output: (None, 31, 31, 512)
batch_normalization_18: BatchNormalization — input: (None, 31, 31, 512) — output: (None, 31, 31, 512)
leaky_re_lu_12: LeakyReLU — input: (None, 31, 31, 512) — output: (None, 31, 31, 512)
zero_padding2d_1: ZeroPadding2D — input: (None, 31, 31, 512) — output: (None, 33, 33, 512)
conv2d_13: Conv2D — input: (None, 33, 33, 512) — output: (None, 30, 30, 1)

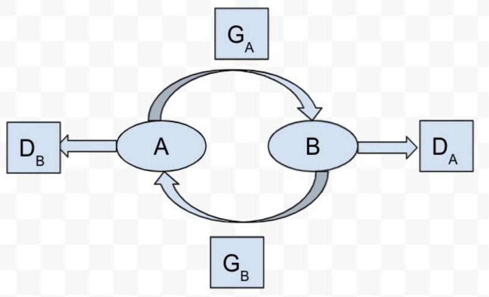
Model B (Unpaired Image Translation)

CycleGAN Architecture [7]

- Used for unpaired image translation task from domain A to domain B
- Trains 2 generators (G_A , G_B) and 2 discriminators (D_A , D_B) for the task
    - Pass image from domain A to G_A to generate image (O_B) from domain B.
    - Pass output of G_A as input to G_B to generate image (O_A) from domain A.
    - D_A outputs the likelihood that generated image is from domain B
    - D_B outputs the likelihood that generated image is from domain A

Both the generators and discriminators are trained using standard adversarial loss

Model flow



- Additional Loss:
    - Forward Cycle consistency loss: Loss enforcing output of generator B to be same as input A
    - Backward Cycle consistency loss: Loss enforcing output of generator A to be same as input B

Experiments performed

Multiple experiments were conducted to understand the effectiveness of GANs.

1. **Paired image to image translation**
   - *L1 loss* - Absolute loss + GAN loss as the generator loss
   - *L2 loss* - Mean squared loss + GAN loss as the generator loss
   - *0.5*L1 + 0.5*L2* - A convex combination of L1 and L2 loss in GAN loss for generator
   - *Patch GAN 16* - Used a receptive field of 16x16 for the discriminator
   - *Patch GAN 286* - Used a receptive field of 286x286 for the discriminator
   - *Skip* - Removed skip connections in the U-Net generator
2. **Unpaired image to image translation**
   - *L1 cyclic loss* - Absolute loss for having cyclic consistency
   - *L2 cyclic loss* - Mean squared error based loss for enforcing cyclic consistency
   - *0.5*L1 + 0.5*L2* - A convex combination of L1 and L2 cyclic oss

Training Loss curves

Training loss curves generated from the L1 loss experiment on the maps dataset for 150 epochs with batch size 16

GAN loss

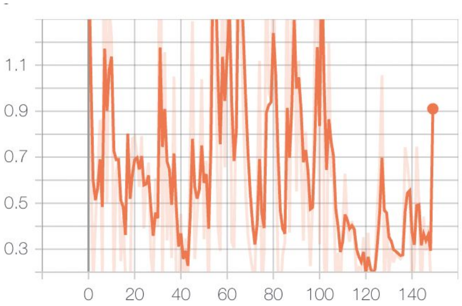
$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z))]$$

L1 loss

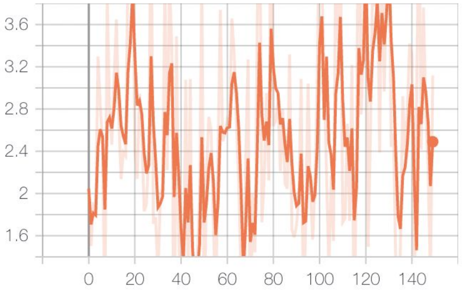
$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1]$$

Total loss = GAN loss + λ * L1 loss

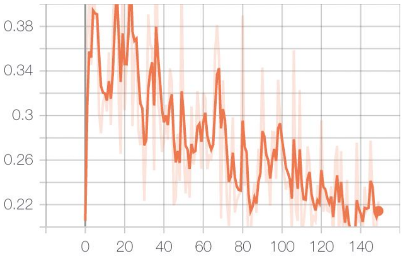
x,y are the input & target images, z is the random noise, G(x, z) is the generated image, D is the output of the discriminator network.



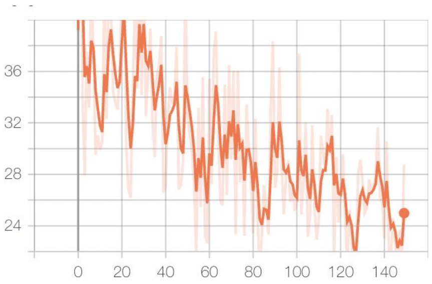
Discriminator loss



Generator - GAN loss



Generator - L1 loss



Generator - total loss

Metrics

- **Precision and recall [8]**
  - Precision: Proportion of generated images that are realistic.
  - Recall: Proportion of realm of realistic images covered.
- **Inception FID score [9]**
  - Calculates the distance between features vectors for real and generated images.
  - Lower the better

Both the metrics are calculated using the feature maps generated from the fake and real images after passing them through the Inception V3 network.

Results: Paired image to image translation

Dataset	**Experiment**	**Precision**	**Recall**	**Inception Score**
Cityscapes	L1 loss	0.52	0.32	100.524
	L2 loss	0.59	0.3	115.312
	0.5*L1 + 0.5*L2	0.61	0.19	108.7
	Patch GAN 16	0.19	0.23	147.508
	Patch GAN 286	0.48	0.19	154.226
	Skip	0.03	0.01	340.1
Maps	L1 Loss	0.43	0.32	152.138
	L2 Loss	0.35	0.15	195.112
	0.5*L1 + 0.5*L2	0.42	0.05	205.145
	Patch GAN 16	0.33	0.06	211.698
	Patch GAN 286	0.31	0.15	198.186
	Skip	0.23	0.01	275.910

Results: Paired image to image translation

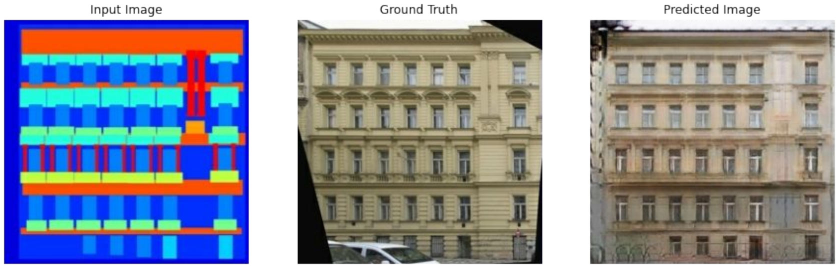
Dataset	Experiment	Precision	Recall	Inception Score
Facades	L1 Loss	0.76	0.39	110.68
	L2 loss	0.64	0.26	118.845
	0.5*L1 + 0.5*L2	0.71	0.39	116.895
	Patch GAN 16	0.76	0.29	116.5
	Patch GAN 286	0.73	0.21	124.9
	Skip	0.13	0.01	275.91

Results: Unpaired Image to image translation

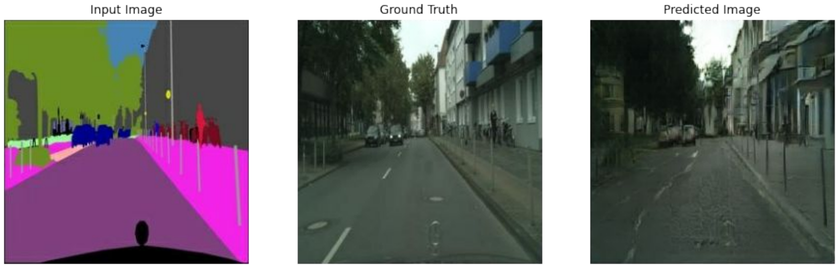
- Only the Inception score is calculated for unpaired task
- Precision and recall require ground truth images, which are not available in the unpaired task.
- L1 loss increases contrast and prevents blurring of edges

Experiment	Maps	Facades	Horse to zebra
L1 cyclic loss	235.67	166.12	204.09
L2 cyclic loss	245.76	205.95	245.89
0.5 * L1 + 0.5 * L2	298.45	187.34	211.78

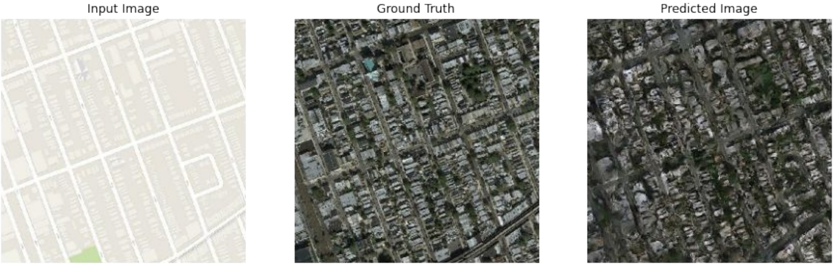
Examples of generated images - paired task



Facades Dataset

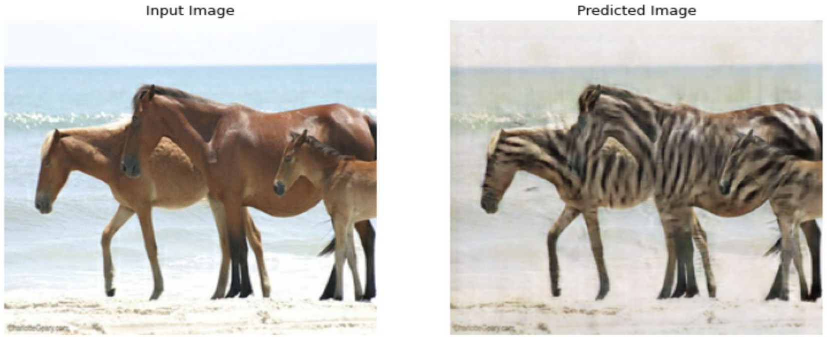


Cityscapes dataset

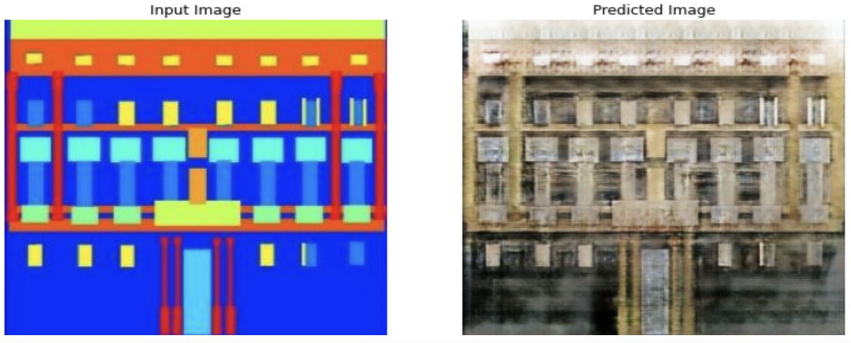


Maps Dataset

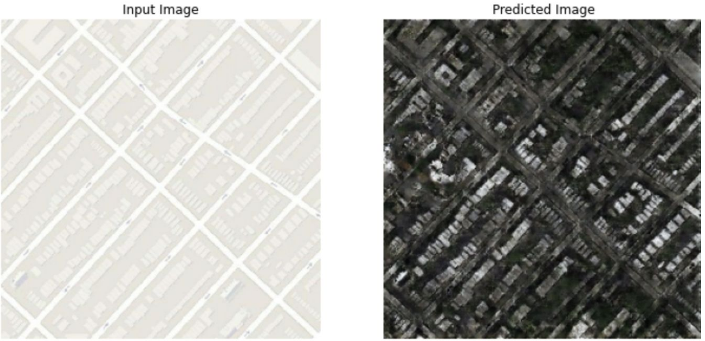
Generated Model images - Unpaired task



Horse to Zebra

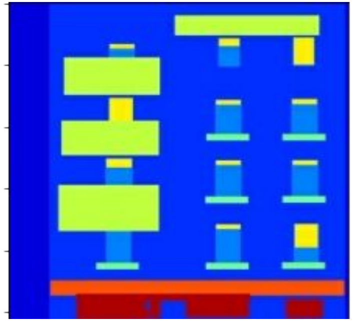


Facades

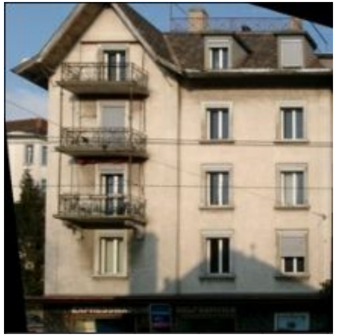


Maps

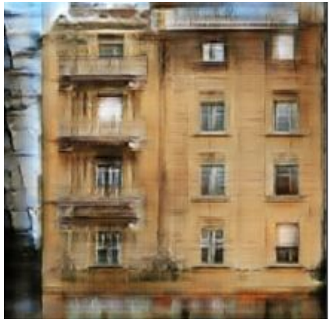
Qualitative Results on Facade Dataset



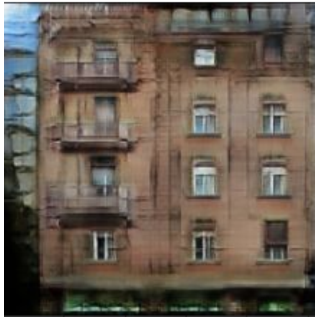
Input



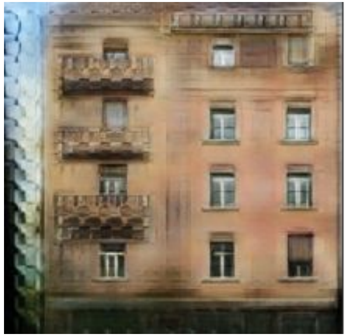
Ground truth



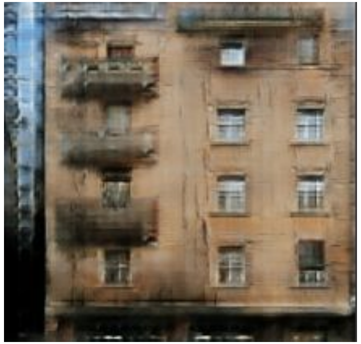
L2 Loss - paired



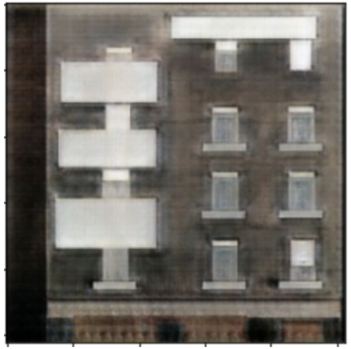
L1 Loss - paired



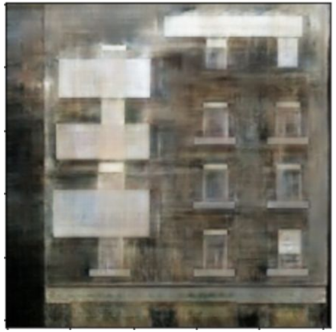
Patch 16 - paired



Patch 286 - paired



L1 cyclic loss - unpaired



L2 cyclic loss - unpaired

Conclusion and Further work

Conclusion

- For paired task, L1 loss generates the best images.
- For unpaired task, L1 cyclic loss generates the best images.
- The images generated by the unpaired task are worse than the images generated by paired task.

Further work

- Experiment with patch GAN variations in unpaired image to image translation task.
- Modularize the code further and make it more readable.
- Qualitative analysis of generated images on cycleGAN dataset.

References

[1] Goodfellow, Ian J., Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial networks." arXiv preprint arXiv:1406.2661 (2014).

[2] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies.

[3]M.-Y. Liu and O. Tuzel. Coupled generative adversarial networks.

[4] R. S. Radim Tylecek. Spatial pattern templates for recognition of objects with regular structure. In Proc. GCPR, Saarbrucken, Germany, 2013.

[5] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In CVPR, 2017.

[6] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In CVPR, 2016.

[7] Jun-Yan Zhu*, Taesung Park*, Phillip Isola, and Alexei A. Efros. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, in IEEE International Conference on Computer Vision (ICCV), 2017.

[8] Kynkäänniemi, T., Karras, T., Laine, S., Lehtinen, J. and Aila, T., 2019. Improved precision and recall metric for assessing generative models. arXiv preprint arXiv:1904.06991.

[9] Heusel, Martin, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. "Gans trained by a two time-scale update rule converge to a local nash equilibrium." arXiv preprint arXiv:1706.08500 (2017).