Image Stitching using GAN

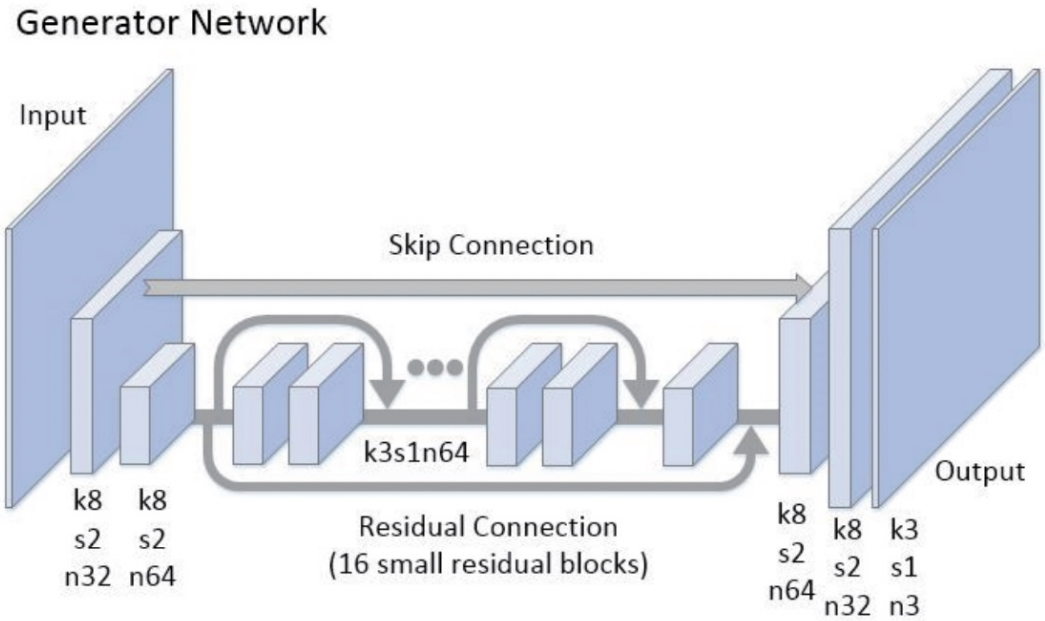
on Images

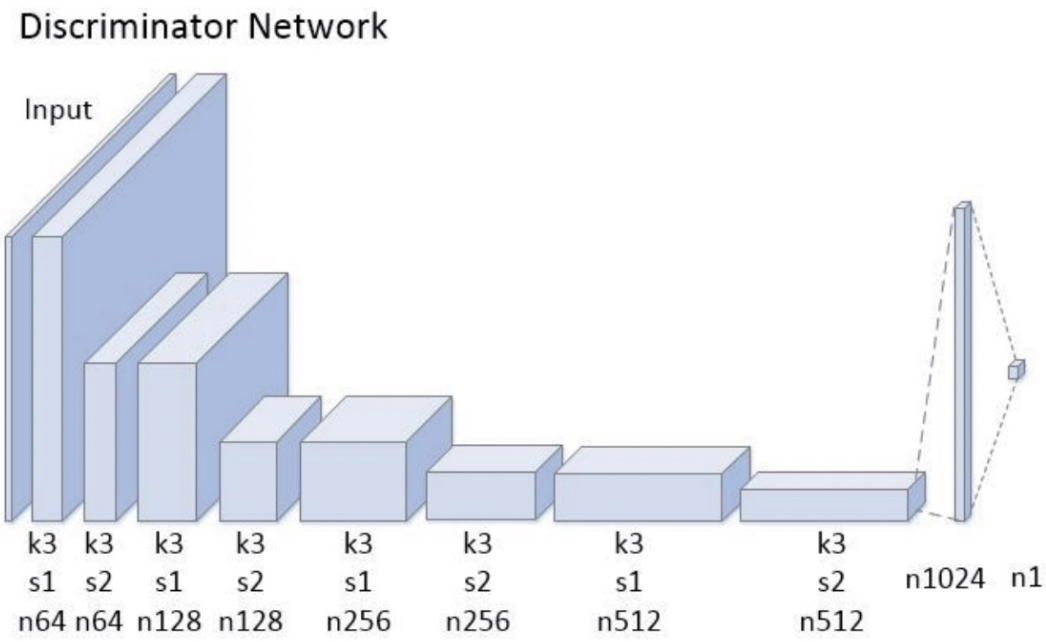
Mithun Parab

Network Architecture

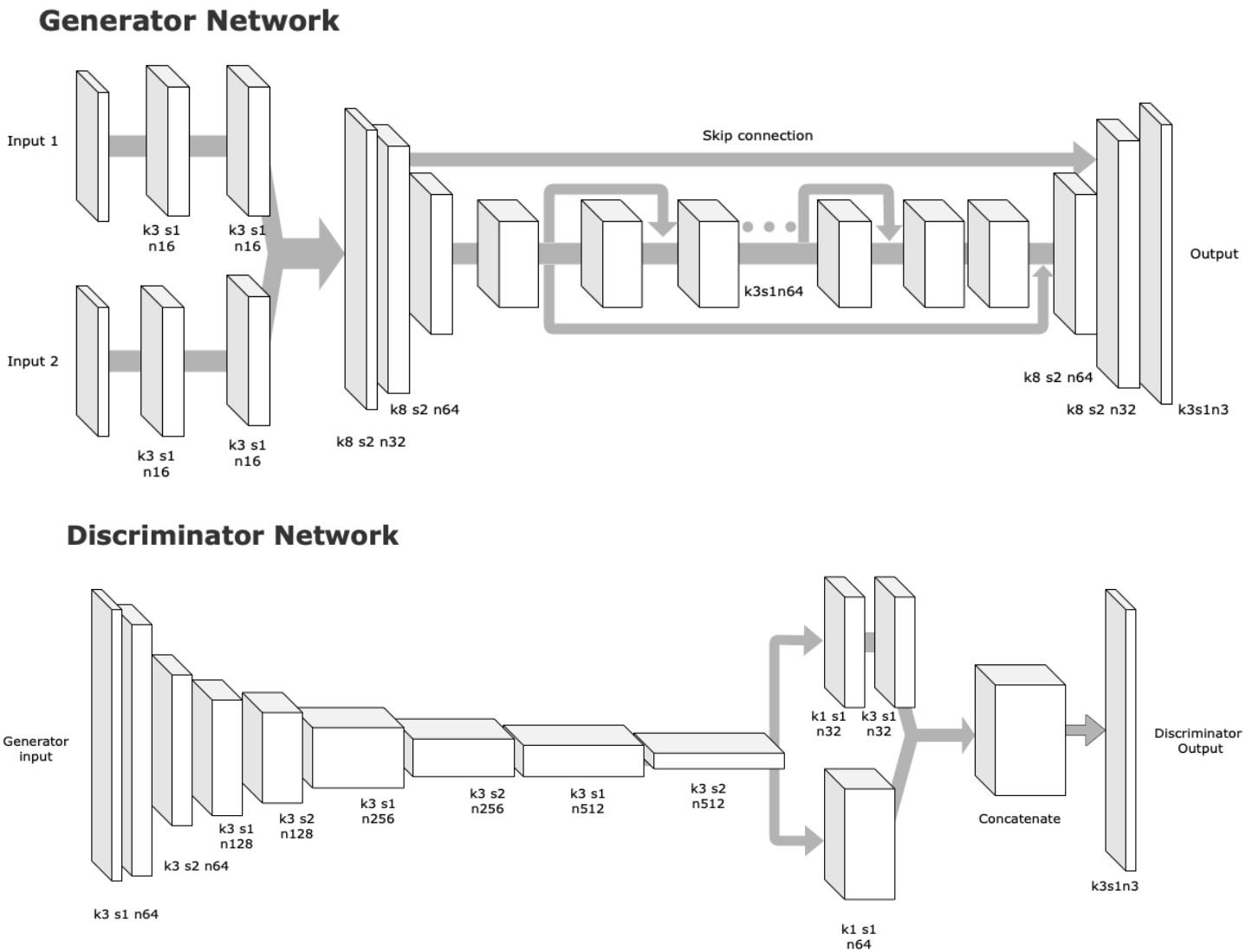
- For the input, We used two input images in RGB format. Later we used convolutional layers with weight normalization and concatenated both of the layers to reconstruct the output image.
- We used Weight Normalization instead of Batch Normalization and AdamW instead of the Adam optimizer with weight decay.
- finally use up-convolution to upscale feature maps and output the blending panorama
- The discriminator is the same as SRGAN. Instead of the fully convolutional layer, we used Network in Network which makes our model compact and efficient

Architecture: Pano-GAN





Architecture: Ours



Inputs and Outputs

- First, resize the images so that their height is 256 and then crop the width to 512. Since the generator network is fully-convolutional, they can be applied to images of any resolution at testing-time.
- The inputs, consisting of two 3-channels (RGB) input images after warping.
- The inputs are scaled to the range of [0, 1] and the ground truths are scaled to the range of [-1, 1].

Loss Functions

- Generator loss, a perceptual loss defined as the weighted sum of a VGG loss and an adversarial loss.

$$\min_{D} l_{Gen} = l_{VGG} + \lambda * l_{Adv}$$

- The VGG loss we use is based on the 4-th ReLU activation feature maps after the 4-th max-pooling layer and before the 5-th max-pooling layer of the pre-trained 19-layer VGG network:

$$l_{VGG} = \frac{1}{whc} \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} \sum_{z=0}^{c-1} (\phi(I^{(Gt)})_{x,y,z} - \phi(G(I^{In})_{x,y,z})^2$$

Loss Functions

- Here, ϕ denotes the pre-trained VGG network, w, h, c denote the width, height and number of channels of the feature maps used, respectively. I^{Gt} denotes the ground truths, I^{In} denotes the input images and denotes the generator network.

Loss Functions

- The adversarial loss follows Charbonnier loss:

$$\mathcal{L}\left(\widehat{I}, I^*\right) = \sqrt{\left\|\widehat{I} - I^*\right\|^2 + \varepsilon^2} \tag{1}$$

- The discriminator loss function also follows charbonnier loss, trying to discriminate the real or fake of its inputs:

$$\mathcal{L}\left(G(\widehat{I}), G(I^*)\right) = \sqrt{\left\|D(G(\widehat{I})) - D(G(I^*))\right\|^2 + \varepsilon^2} \tag{2}$$

Difference

	Pano-GAN	Our
Optimizer	Adam	AdamW
GAN loss	MSE	Charbonnier Loss
Generator loss	VGG loss + Adversarial loss	VGG loss + Charbonnier Loss
Discriminator loss	LSGAN	Charbonnier Loss
Normalisation	BatchNormalization	WeightNormalization
Total inputs	1	2
Binary mask as a input	✓	×
Weight decay	×	✓
Input shape	(None, None, 7)	(None, None, 3), (None, None, 3)

Table: Difference between our and panoGan model

Testing

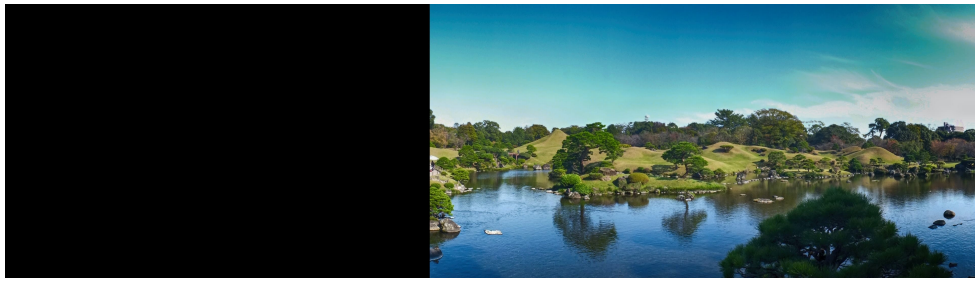
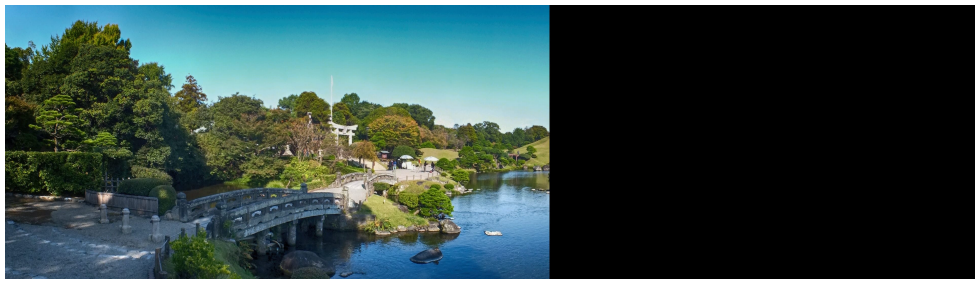
Input 1	
Input 2	
Output	
Ground Truth	

Table: Inputs and Output vs Ground Truth

Thank you for listening!

Mithun Parab

mithun.sp@somaiya.edu