Text To Image Synthesis Using Generative Adversarial Network

Srikant Nayak

Research Supervisor **Dr. Sumitra S**

Co-Supervisor **Dr. Deepak Mishra**

Department of Mathematics
Indian Institute Of Space Science and Technology
Thiruvananthapuram, Kerala, India 695547

Overview

- Motivation
- Generative Adversarial Networks (GANs)
- Previous work on Text to Image Synthesis
- Current work on Text to Image Synthesis
- Datasets
- Results
- Conclusions
- Future works
- Plagiarism Report
- References

Motivation

- Generating photo-realistic images from text is an important problem and has tremendous applications, including photo-editing, computer-aided design, etc.
- Recently, Generative Adversarial Networks (GAN) have shown promising results in synthesizing real-world images
- Conditioned on given text descriptions, **conditional GANs** are able to generate images that are highly related to the text meanings. However, it is very difficult to train GAN to generate high-resolution images from text descriptions.
- This problem is more severe as the image resolution increases.

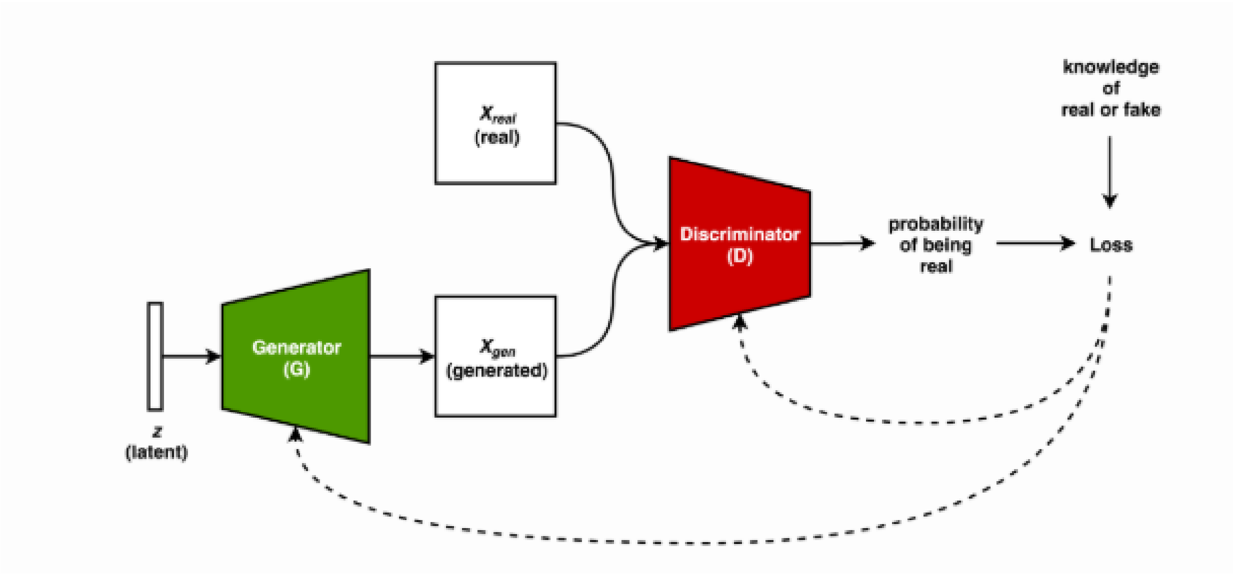
Generative Adversarial Networks (GANs)

GAN consist of 2 model:

Discriminator D estimates the probability of a given sample coming from the real dataset and tries to identify whether the given image is real (from the real data samples) or fake (generated by the generator)

Generator G generates fake image such that the probability distribution of real image is equal to the probability distribution of the fake image, so that it tries to fool the discriminator.

At each training cycle, the generator **G** tries to get better at fooling the discriminator **D** while the discriminator **D** tries to not get fooled.



GANs game:

$$\min_G \max_D V_{GAN}(D, G) = \underbrace{\mathop{\mathbb{E}}_{x \sim p_{data}(x)} [\log D(x)]}_{\text{real samples}} + \underbrace{\mathop{\mathbb{E}}_{z \sim p_z(z)} [\log(1 - D(G(z)))]}_{\text{generated samples}}$$

- **Discriminator** needs to:
  - Correctly classify real data:

$$\boxed{\max_{D} \mathop{\mathbb{E}}_{x \sim p_{data}(x)} [\log D(x)]} \qquad D(x) \to 1$$

  - Correctly classify wrong data:

$$\boxed{\max_{D} \mathop{\mathbb{E}}_{z \sim p_{z}(z)} [\log(1 - D(G(z)))]} \qquad D(G(z)) \to 0$$

- The discriminator is an **adaptive loss function**.

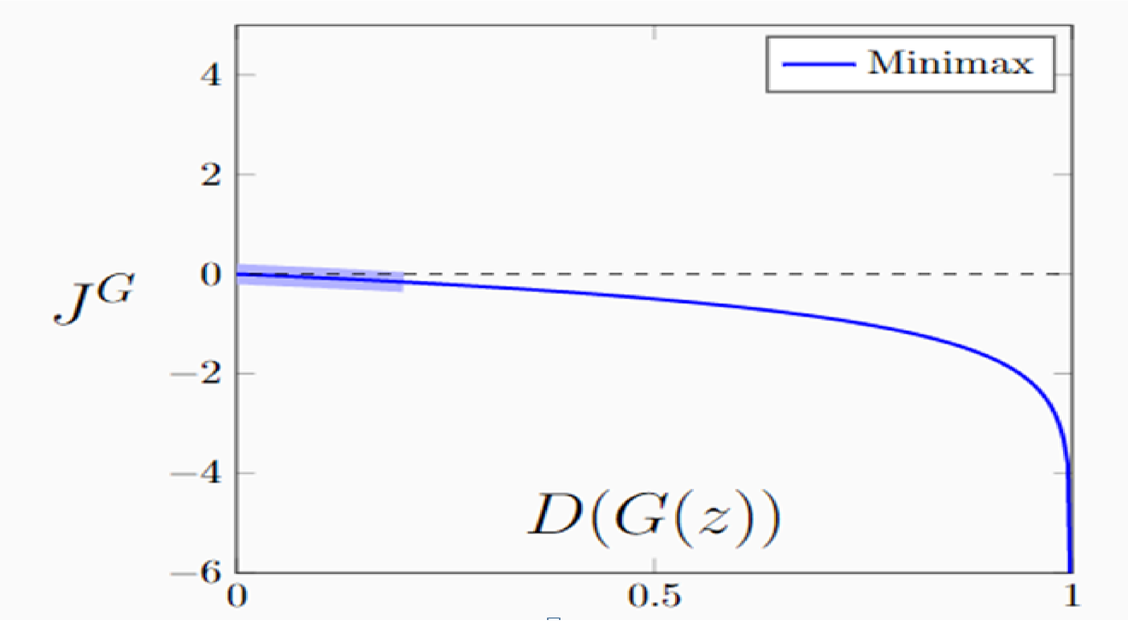
- **Generator** needs to **fool** the discriminator:
    - Generate samples similar to the real ones:

$$\boxed{\min_{G} \mathop{\mathbb{E}}_{z \sim p_z(z)} [\log(1 - D(G(z)))]} \qquad D(G(z)) \rightarrow 1$$

Minimax: log(1-D(G(z)))

X-axis : Is the output of the discriminator of the fake image i.e. D(G(z)

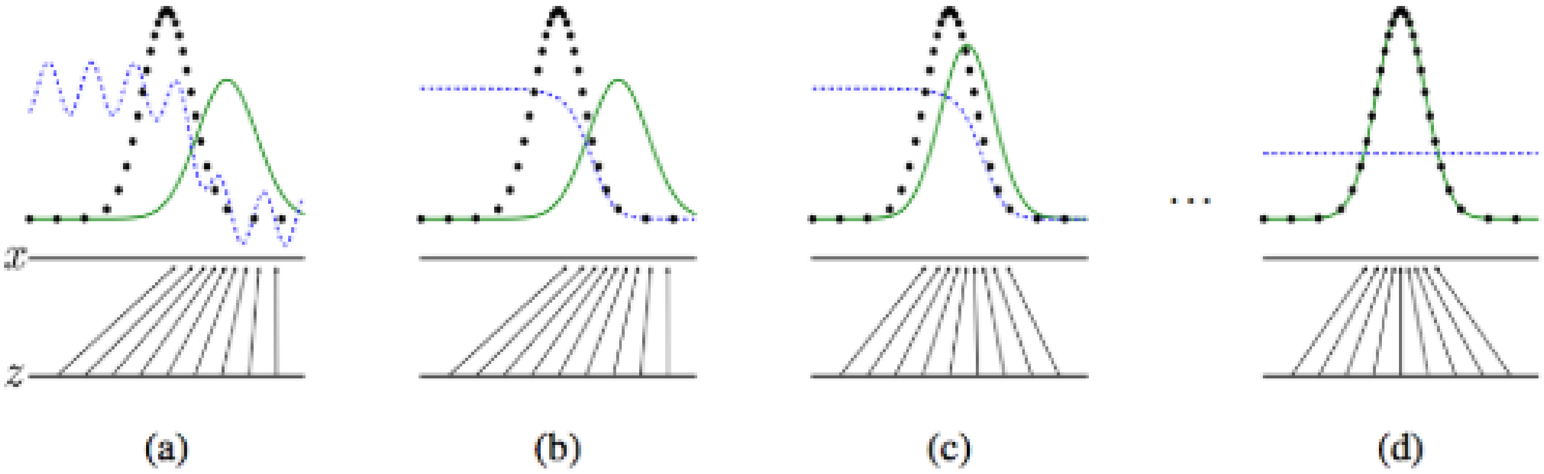
Y-axis : Is the loss function of the generator



GAN Training

- D and G are competing against each other.
- Alternating execution of training steps.
- Use minibatch stochastic gradient descent/ascent
- Optimizer: ADAM , Momentum , RMSProp
- Arbitrary number of steps or epoch
- Training is completed when **D** is **completely fooled** by **G**

Visualization



- Blue dashed line = D(x)
- Green line= probability distribution of the generated sample
- Black dotted line= probability distribution of the real sample

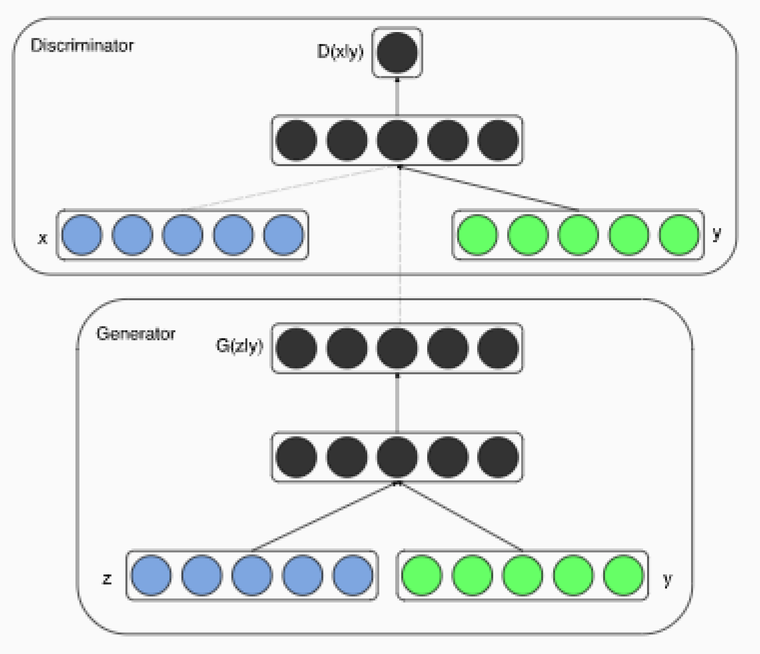
Type of GANs

Two big families:

- Unconditional GANs (just described).
- Conditional GANs (Mirza and Osindero, 2014)

Conditional GAN

- Both G and D are **conditioned** on some extra information y.
- In **practice**: perform conditioning by feeding y into D and G



The GANs game becomes:

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

Notice: the same representation of the condition has to be presented to both network.

Kullback–Leibler and Jensen–Shannon Divergence

KL-Divergence: measures how one probability distribution p(x) diverges from a second expected probability distribution q(x)

$$D_{KL}(p||q) = \int_x p(x) \log \frac{p(x)}{q(x)} dx$$

Jensen-Shannon divergence(JSD) : It also measure the similarity between the probability distribution . It is also known as Information radius or total divergence to the average. Let P(x) and Q(x) be the two different probability distribution then

$$D_{JS}(p||q) = \frac{1}{2} D_{KL}(p||\frac{p+q}{2}) + \frac{1}{2} D_{KL}(q||\frac{p+q}{2})$$

Wasserstein Distance

- It is also called Earth Mover Distance. This Earth-Mover distance is the stable distance metric for PDF comparison and measure the similarity between the two probability distribution by calculating the distance between them **horizontally not vertically** as we are doing in KL and JS divergence.

probability distribution of x & y
- before and after the move

a.k.a. min

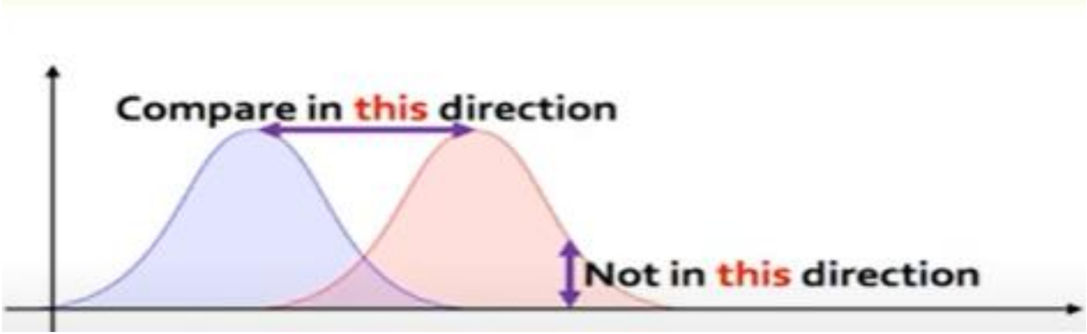
expected moving cost of the plan γ

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}\left[\, \|x - y\| \,\right]$$

find one that has the lowest cost

all possible moving plans

distance between x & y



Why GAN is unstable?

- Supports of $p(x)$ and $p_\theta(x)$ are disjoint[1] a.s.
- Then

$$JSD(p \| p_\theta) = \log 2$$

$$KL(p \| p_\theta) = KL(p_\theta \| p) = +\infty$$

- The loss *does not* provide a valuable information

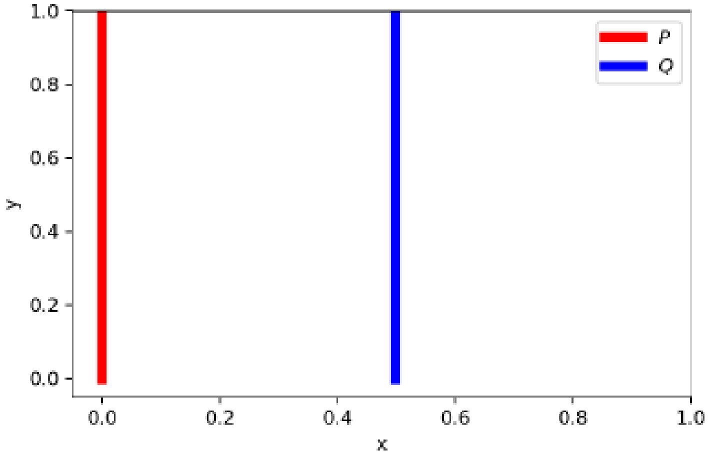
Solution

1. Add noise to overlap supports
2. Use *better* divergence

Why Wasserstein is better than JS or KL divergence?

Toy example

- Let $z \sim U[0,1]$ and $x = (0, z) \sim p(x)$
- Let $G_\theta(z) = (\theta, z)$, hence $p_\theta(x) = p(x)$ for $\theta = 0$



When $\theta \neq 0$:

$$
\begin{aligned}
D_{KL}(P\|Q) &= \sum_{x=0, y\sim U(0,1)} 1 \cdot \log\frac{1}{0} = +\infty \\
D_{KL}(Q\|P) &= \sum_{x=\theta, y\sim U(0,1)} 1 \cdot \log\frac{1}{0} = +\infty \\
D_{JS}(P,Q) &= \frac{1}{2}\left(\sum_{x=0, y\sim U(0,1)} 1 \cdot \log\frac{1}{1/2} + \sum_{x=0, y\sim U(0,1)} 1 \cdot \log\frac{1}{1/2}\right) = \log 2 \\
W(P,Q) &= |\theta|
\end{aligned}
$$

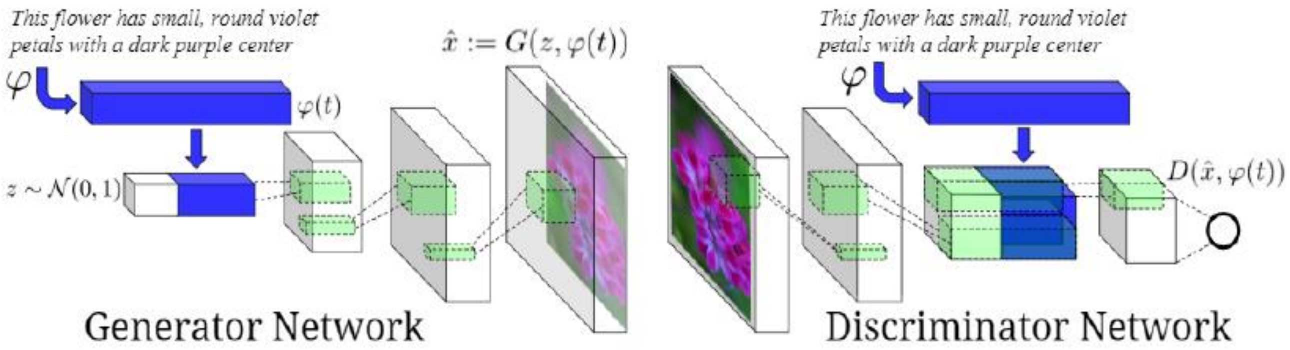
But when $\theta = 0$, two distributions are fully overlapped:

$$
\begin{aligned}
D_{KL}(P\|Q) = D_{KL}(Q\|P) = D_{JS}(P,Q) = 0 \\
W(P,Q) = 0 = |\theta|
\end{aligned}
$$

Previous Work on
Text to Image synthesis

State of art Model:

- In 2016, text to image have introduced .They used this application by using Deep convolution GAN(DCGAN) architecture.
- The architecture of the text to image is as follows:



Limitation

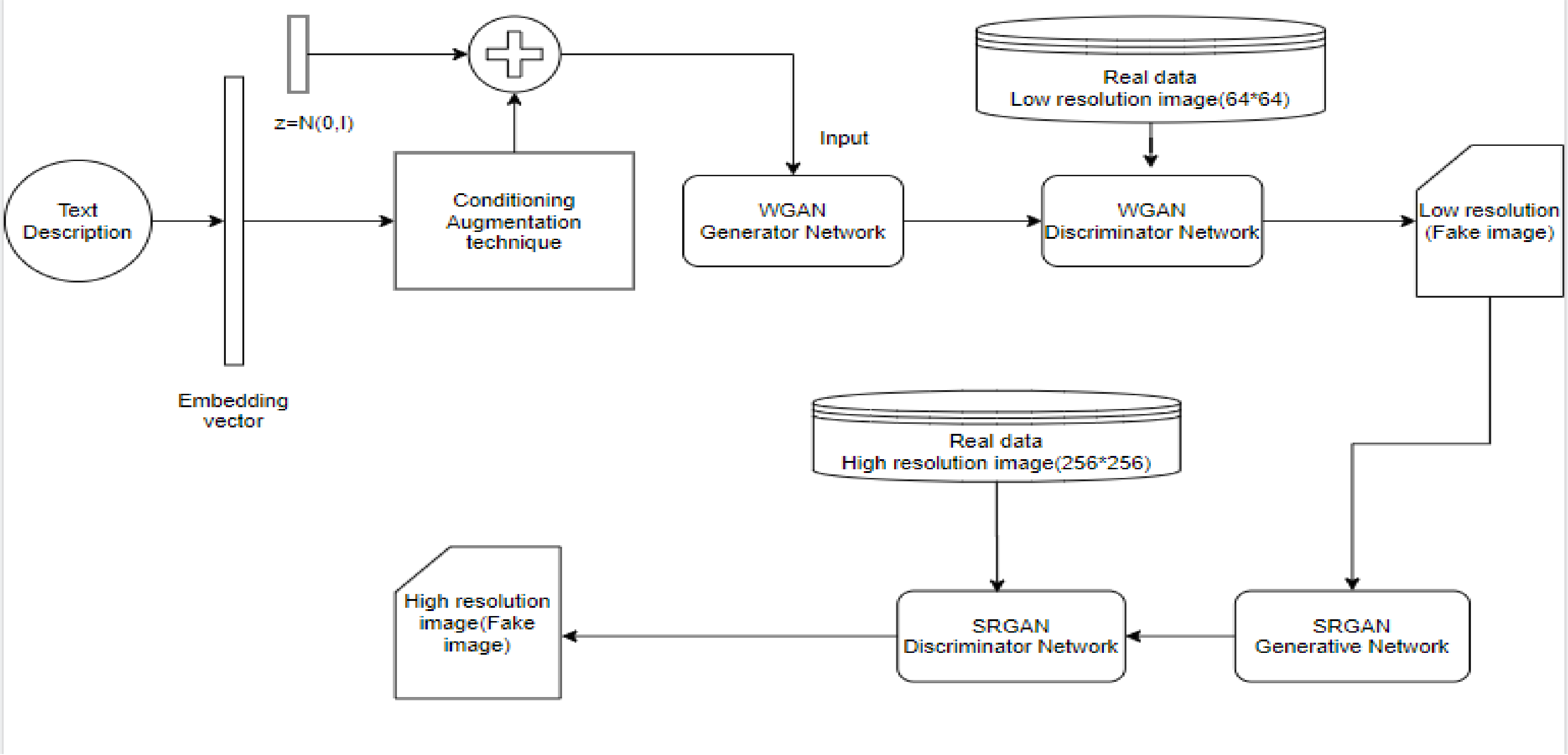
- Instability during training
- Mode collapse occurs
- Failed to contains useful information according to the text description
- Unable to generate high resolution Image

Contributions

- To overcome the limitation we apply conditional Wasserstein GAN to generate low resolution image
- To enhance the resolution , we apply super resolution GAN

Current work on
Text to Image Synthesis

Architecture of Text to Image Synthesis



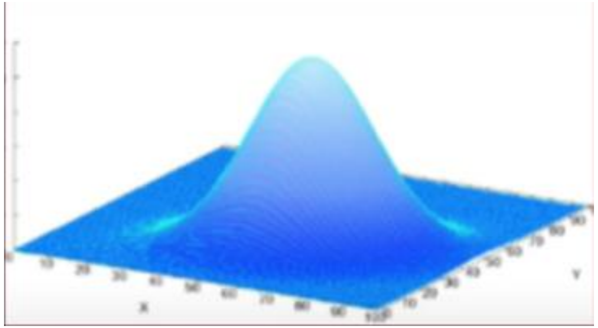
Text to Image

This architecture is divided into three sub stage:

1. Conditional Augmentation Technique
2. Stage-I, Low resolution text to image
3. Stage-II, High resolution text to image

Conditional Augmentation Technique

- Technique used is Data augmentation technique conditioned on text embedding
- Gives more training pairs given a small number of image-text pairs
- It thus encourages robustness to small perturbations



Stage-I GAN (Low resolution text to image)

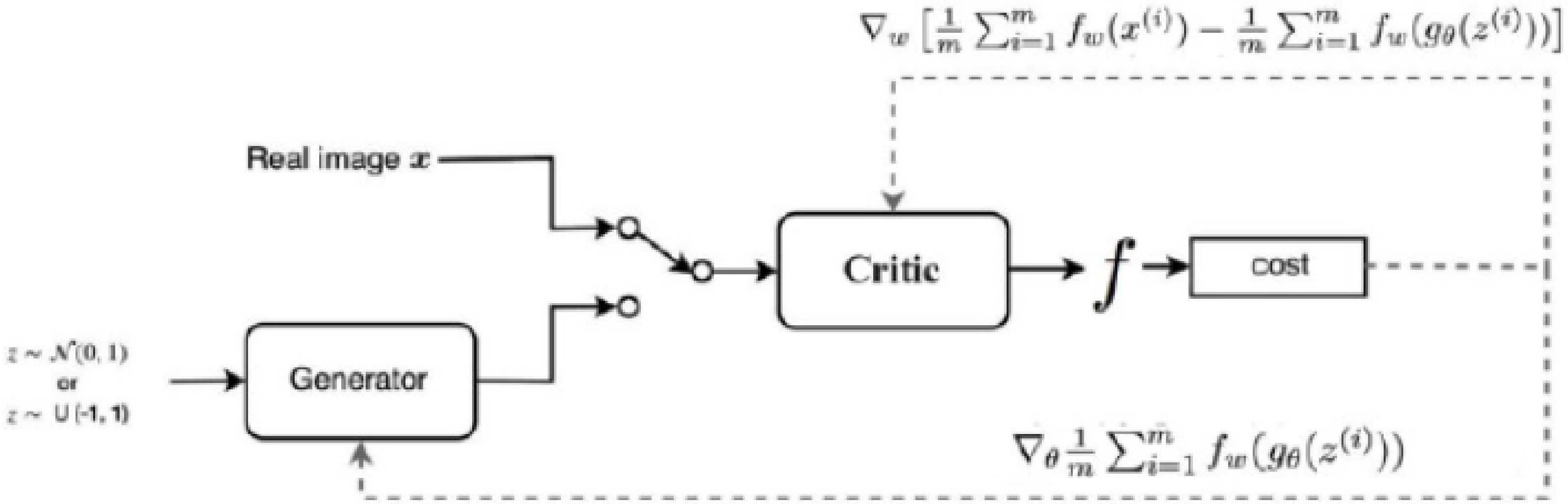


Figure 2.5: THE WGAN Architecture

Wasserstein GAN

- **Deer** act as a **discriminator** try to **maximize** the distance
- **Leopard** act as a **generator** try to **minimize** the distance

Loss function

The loss function makes the discriminator approximate $W(\mathbb{P}(x,e)_{r-mat}, \mathbb{P}(\dot{x},e)_{r-mis})$ he joint distributions of matching and mismatching text-image pairs

Loss function of discriminator:

$$L_D = \mathbb{E}_{(X,E)\sim\mathbb{P}_{ge}}[D(x,e)] + \alpha\mathbb{E}_{(X,E)\sim\mathbb{P}_{r-mis}}[D(x,e)] - (1+\alpha)\mathbb{E}_{(X,E)\sim\mathbb{P}_{r-mat}}[D(x,e)] + \lambda L_{LP}$$

α is the parameter that controls the level of text-image matching.

$$L_{LP} = \mathbb{E}_{(\hat{X},E)\sim\mathbb{P}_\eta}[max(0, \|\nabla_{\hat{x}} D(\bar{x},e)\| - 1)^2 + max(0, \|\nabla_e D(\hat{x},e\| - 1)^2]$$

It is another regularization term that enforces the Lipschitz constraint.

Loss function of generator:

$$L_G = -\mathbb{E}_{(X,E)\sim\mathbb{P}_g}[D(x,e)] + \mathbb{E}_{T\sim\mathbb{P}_r}[\rho KL(\mathcal{N}(\mathbf{0},\mathbf{I}) \parallel \mathcal{N}(\mu(\phi(t)), \Sigma(\phi(t)))]$$

Algorithm

Input: minibatch images **x**, matching text **t**, mismatching $\hat{t}$,number of training batch step **S**

for n=1 to S:

$h \leftarrow \phi(t)$ Encode matching text description

$\hat{h} \leftarrow \phi(\hat{t})$ Encode mis-matching text description

$z \sim \mathcal{N}(0, I)$ Draw sample of random noise

$\hat{x} \leftarrow \boldsymbol{G}(z, h)$Forward through generator

$s_r \leftarrow \boldsymbol{D}(x, h)$ real image, right text

$s_w \leftarrow \boldsymbol{D}(x, \hat{h})$ real image, wrong text

$s_f \leftarrow \boldsymbol{D}(\hat{x}, h)$ fake image, right text

$\hat{x} \leftarrow \varepsilon x + (1 - \varepsilon)\hat{x}$

$\boldsymbol{D_w}(\hat{x})_r - s_f$ wasserstein distance between $s_r ands_f$

$\boldsymbol{D_w}(x)_r - s_w$ wasserstein distance between s_r and s_w

$\mathcal{L}_D{}^{(i)} \leftarrow \boldsymbol{D_w}(x) - \boldsymbol{D_w}(\hat{x}) + (\lambda \left\|\nabla_{\hat{x}} D_w(\hat{x})\right\|_2 - 1)^2$

$\boldsymbol{D} \leftarrow \boldsymbol{D} - \alpha \frac{dL_D}{dD}$ Update discriminator

$\mathcal{L}_g^{(i)} \leftarrow -\boldsymbol{D_w}(G_\theta(z))$

$\boldsymbol{G} \leftarrow \boldsymbol{G} - \alpha \frac{dL_g}{dG}$ Update generator

end for

Summary

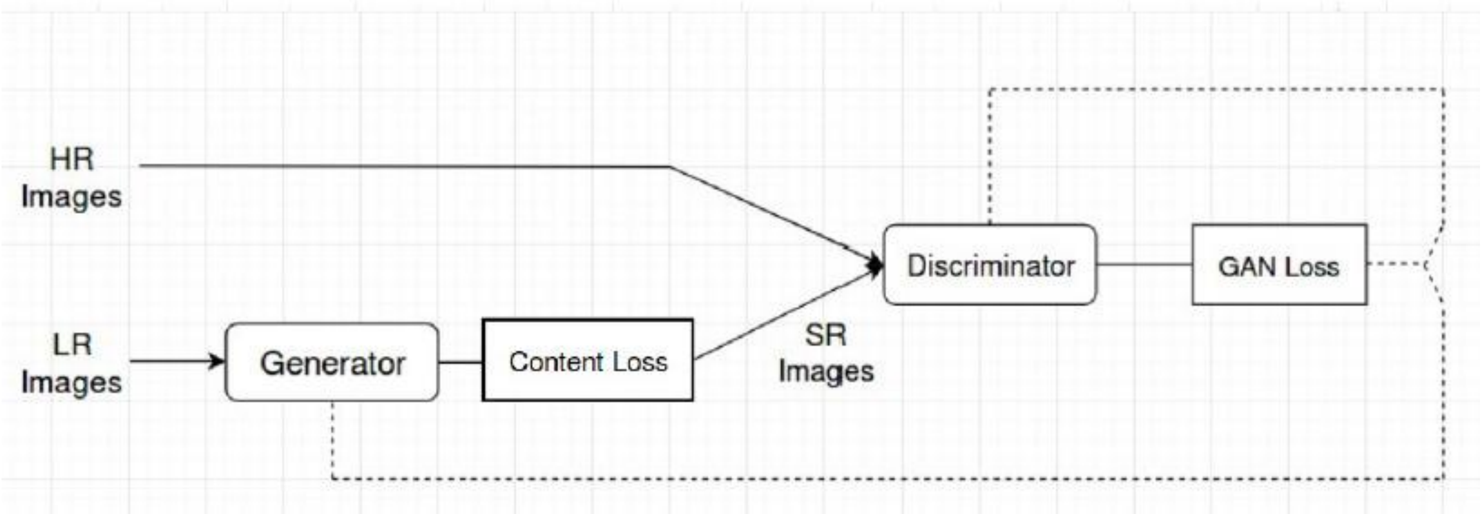
Benefits

- Stable training
- Loss correlates with desired result
- No mode collapse

Problems

- Does not work well with momentum-based optimizer s e.g. Adam
- Slower to converge than KL loss
- Requires hyper-parameter tuning

Stage-II GAN (High resolution Text to Image synthesis)



Real

Latent code(Z)

$$\min_{G}\max_{D} \mathbf{E}_{I^{HR} \sim p_{traion}}\left[\log D_{\theta_D}(I^{HR})\right] + \mathbf{E}_{I^{LR} \sim p_g}\left[\log\left(1 - D_{\theta_D}(G_{\theta_G}(I^{LR}))\right)\right]$$

Architecture

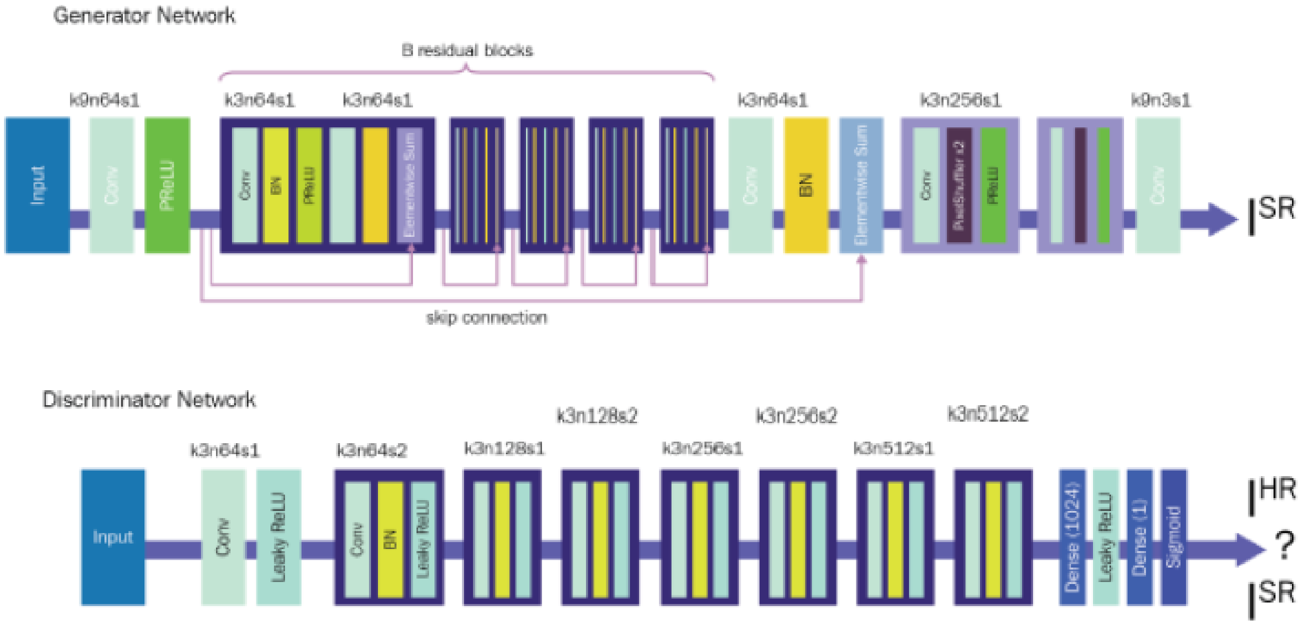


Figure 5.2: Architecture OF SRGAN

Loss function

In SRGAN, there are two kinds of loss are present as:

1. Content Loss
2. GAN Loss(Adversarial Loss)

Content Loss:

There are two types of content loss:

1.Pixel wise MSE(mean squared error) loss

2. VGG19 Loss

Loss Function

Pixel wise MSE loss: Calculated between each pixel value of the generated image and each pixel value of the real image. It shows how different the generated image from the real image.

$$l_{MSE}^{SR} = \frac{1}{r^2WH} \sum_{x=1}^{rW} \sum_{y=1}^{rH} (I_{x,y}^{HR} - G_{\theta_G}(I^{LR})x, y)^2$$

VGG19 Loss: Calculated as the feature map of the real image and the   generated image .

$$l_{VGG/i,j}^{SR} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\Phi_{i,j}(I^{HR})x, y - \Phi_{i,j}(G_{\theta_G}(I^{LR}))x, y)^2$$

Loss Function

GAN Loss : This loss is calculated on the probabilities returned by the discriminated network. In the adversarial model the input of the discriminator model is the output of the generator model.

$$l_{Gen}^{SR} = \sum_{n=1}^{N} -log D_{\theta_D}(G_{\theta_G}(I^{LR}))$$

Perceptual loss is the weighted sum of the adversarial loss and the content loss which is represented by,

$$l^{SR} = .001 * l_{Gen}^{SR} + 1 * l_X^{SR}$$

Objective: Minimize the perceptual loss

Algorithm

1. ***For** epoch **do***
2. *Obtain higher dimensional I^{SR} form I^{LR}: $I^{SR} = G(I^{LR})$*
3. *Export the high-dimensional real (I^{HR}) and fake (I^{SR}) image as input to D.*
4. *Train D (update weights):*

$$\nabla_{\theta_d} = \frac{1}{m}\sum_{i=1}^{m}[logD(I^{HR}) + log(1 - D(I^{SR}))]$$

5. *Train G (update weights):*

$$\nabla_{\theta_g} = 10^{-3} * \nabla_{\theta_{adversarial}} + \nabla_{\theta_{content}}$$

$$\nabla_{\theta_{adversarial}} = \frac{1}{m}\sum_{i=1}^{m} -log(D(I^{SR}))$$

$$\nabla_{\theta_{content}} = \|VGG19(I^{HR}) - VGG19(I^{SR})\|_2$$

6. ***endFor***

Datasets

Datasets

Caltech CUB-200 birds dataset: contains 11,788 birds images of 200 different type of birds

Small, mostly yellow bird, with brown, white and black stripes on its wings and tail.

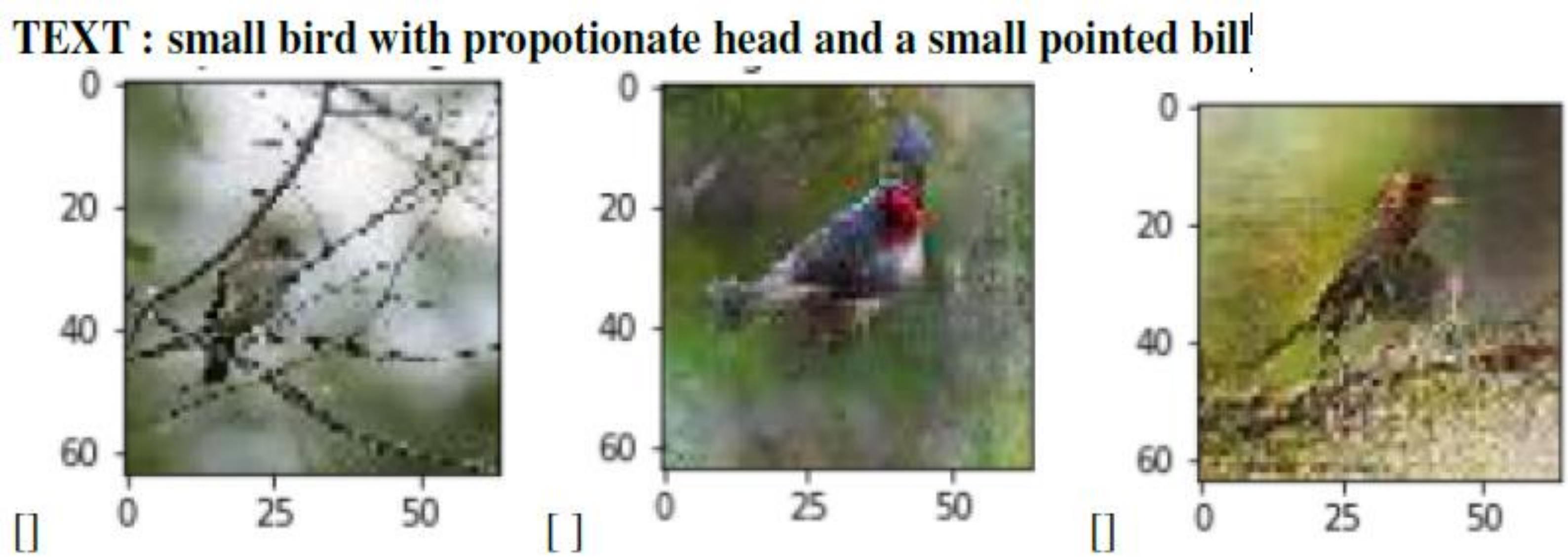
A bird with a small triangular bill, black cheek patch and blue plumage across its body.

Datasets

OXFORD-102 flower dataset contains 8192 flower images from 102 categories of the flower which have large scale and light variations.

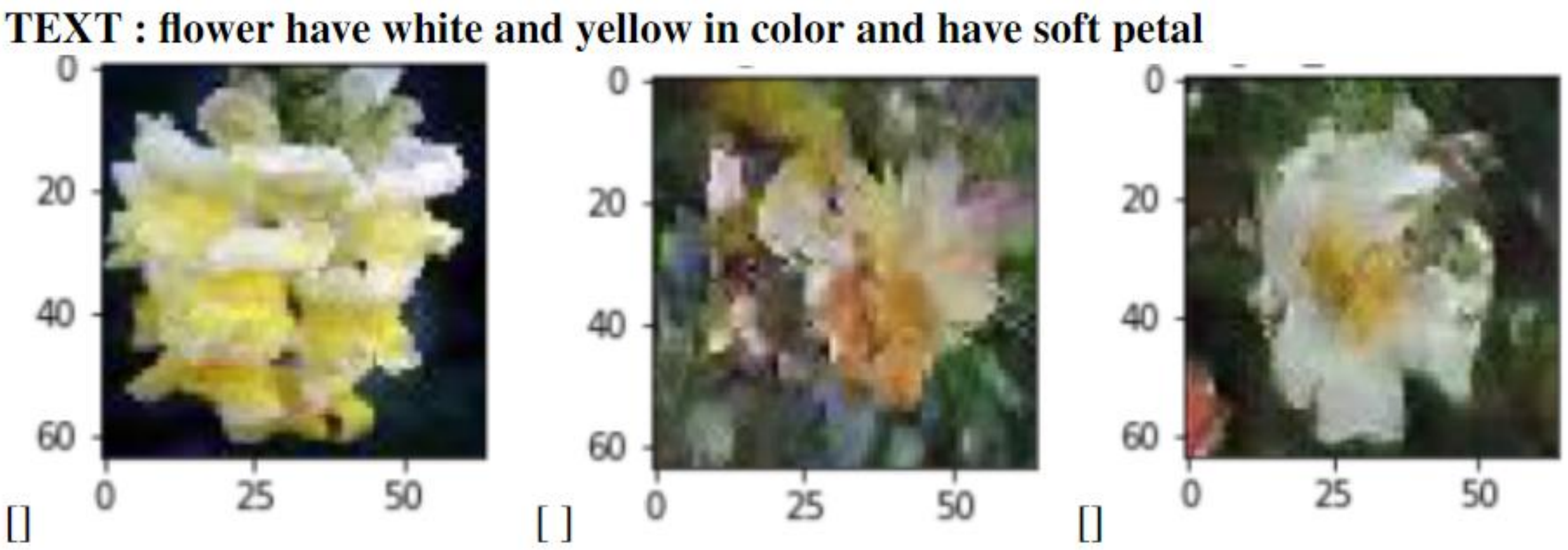
Results

Results of Stage-I GAN



Comparison between (a) EXPECTED IMAGE (b) GAN-CLS (c) CONDITIONAL WGAN.

Results of Stage-I GAN



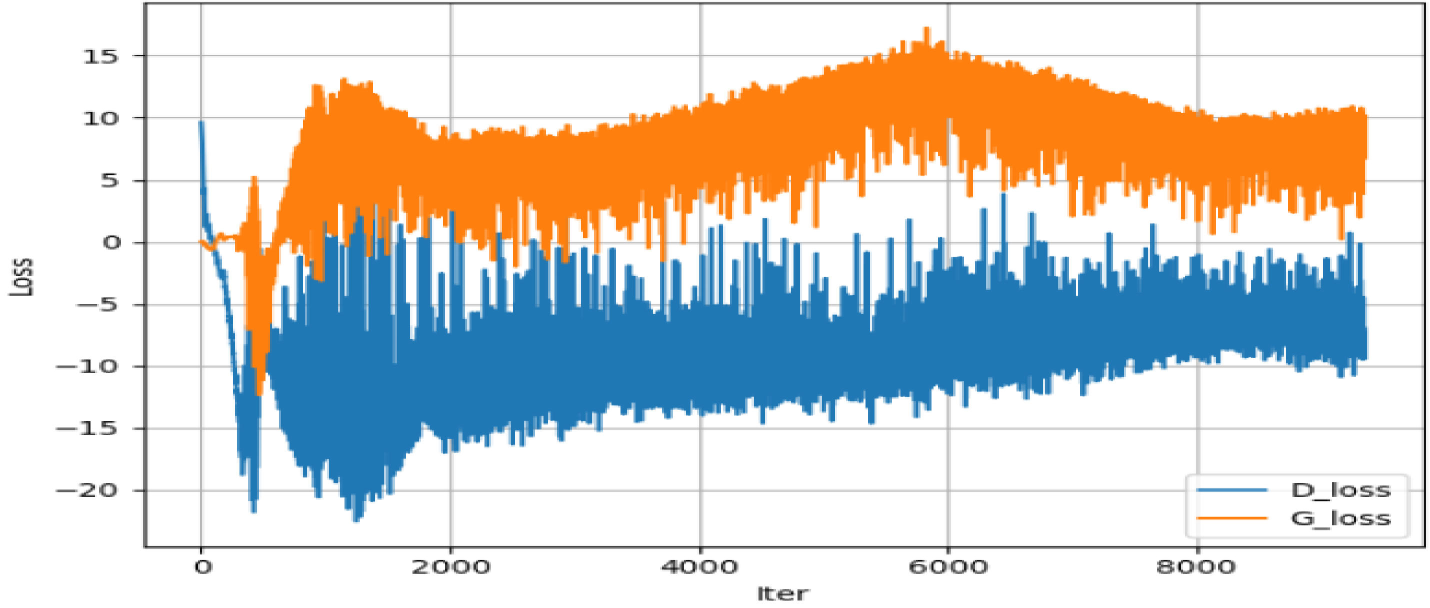
Comparison between (a) EXPECTED IMAGE (b) GAN-CLS (c) CONDITIONAL WGAN

Results of Stage-I GAN

Analysis of Loss Function

X-axis: Number of iteration

Y-axis: Generator Loss and Discriminator Loss



Results of Stage-II GAN

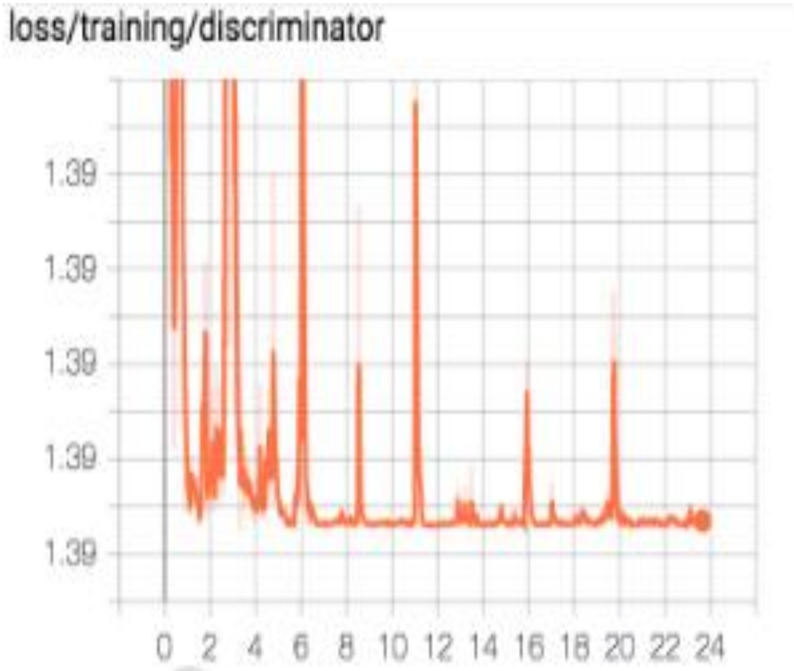
comparison between **Low resolution** image and **super resolution** image

Results of Stage-II GAN

comparison between **Low resolution** image and **super resolution** image

Results of Stage-II GAN

Analysis of Loss Function:



Conclusions

And

Future works

Conclusions

- We discuss various architecture of GAN. We came to know that WGAN is better then GAN for generative model
- We applied Conditional Wasserstein GAN for low resolution Text to Image synthesis and SRGAN for high resolution Text to Image synthesis
- We compare the result to the current state of art model

Future Works

- Find how the conditional Wasserstein loss function can be used in a more advanced model.
- Work to gain more information on generated image condition on text description
- We also try a model named progressive growing conditional Wasserstein GAN for a better image.
- Enhance resolution to (512*512) and (1024*1024)

Plagiarism Report

GENERATIVE ADVERSARIAL TEXT TO IMAGE SYNTHESIS

ORIGINALITY REPORT

16%	8%	12%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.arxiv-vanity.com Internet Source	1%

References

[1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[2] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona, "Caltech-UCSD Birds 200," California Institute of Technology, Tech. Rep. CNS-TR-2010-001, 2010.

[3] Y. Bengio, "The consciousness prior," *arXiv e-prints*, vol. 1709.08568, Sep. 2017. [Online]. Available: https://arxiv.org/abs/1709.08568

[4] L. M. Alec Radford and S. Chintala., "Unsupervised representation learning with deep convolutional generative adversarial networks," *CoRR, abs/1511.06434*,, 2015.

[5] A. F. Henning Petzka and D. Lukovnikov., "On the regularization of wasserstein gans." *In International Conference on Learning Representations*, 2018.

[6] B. S. Scott Reed, Zeynep Akata and H. Lee., "Learning deep representations of fine-grained visual descriptions." *In IEEE Computer Vision and PatternRecognition*, 2016.