Team #10

Multi-Stage Text to Image Generation

Members: Hamad Alduaij, Muhammad Saleem, Parvathy Ajith, Surya Cherupally

EEE 511: Artificial Neural networks

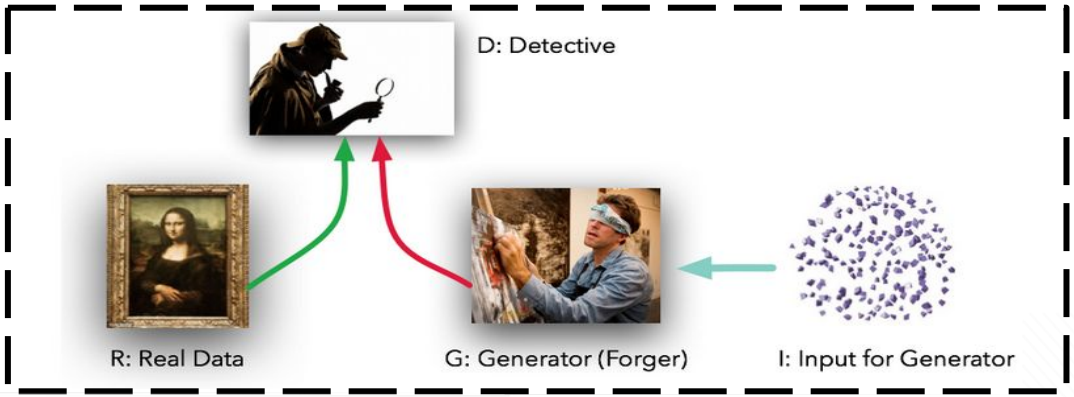
Arizona State University

Text to image conversion

- Among the various applications of Computer Vision, Text to Image construction is one that has gained much popularity in recent times.
- The major challenge is efficiently transferring word meaning to high-dimensional latent space and then mapping it to an image.
- Applications:
  - ❖ Forensic sketching to identify criminals.
  - ❖ Creating avatars in video games from the description given by players
  - ❖ Pictorial representation of story books.
  - ❖ Picture editing without expert artist, or complex software.

How?: Generative Adversarial Neural Networks (GANs)

Concept



Block Diagram

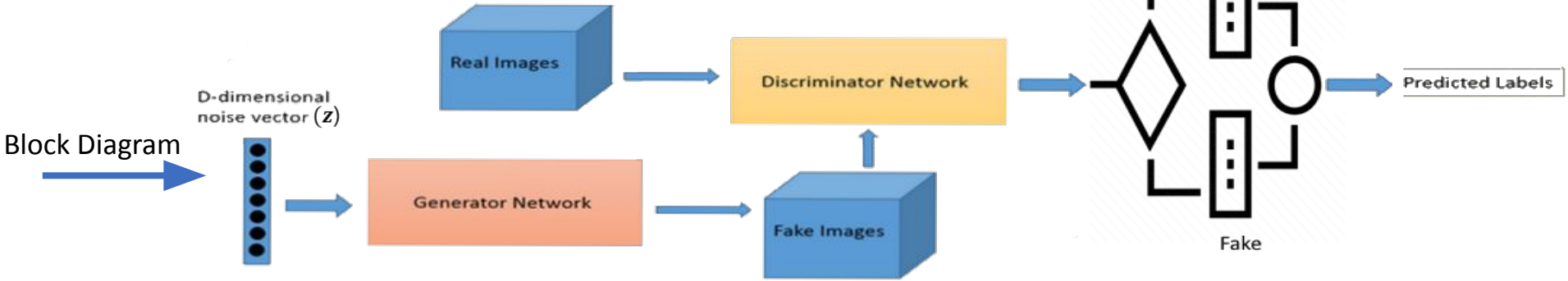


Figure 1: Illustrative representation of GANS with block diagram.
https://towardsdatascience.com/decrypt-generative-artificial-intelligence-and-gans-16646dbb4426

GANs: Theoretical foundation

Preliminaries

$D(\boldsymbol{x}) = \begin{cases} 1, predict\ Real \\ 0, predict\ Fake \end{cases}$, Recall in classification $if\ y = D(x)$ and y is Boolean, then $E_x[D(x)] = p(y = 1)$.

from a noise vector ***z,*** $G(\boldsymbol{z})$ is the generated image

---

Value Function: $\min_G \max_D V(G,D) = E_{\boldsymbol{x}\sim p_{data}}[log(D(\boldsymbol{x}))] + E_{\boldsymbol{x_z}\sim p_{\boldsymbol{z}}}[log(1 - D(G)(\boldsymbol{z}))]$

$Real$

$Fake$

$Log\ prob.\ of\ predicting\ real\ when\ input\ is\ real$

$Log\ prob.\ of\ predicting\ fake\ when\ input\ is\ fake$

GANs: Loss function

Value Function: $\min_G \max_D V(G,D) = E_{\boldsymbol{x}\sim p_{data}}[log(D(\boldsymbol{x})] + E_{\boldsymbol{x_z}\sim p_z}[log(1-D(G(\boldsymbol{z}))]$

Training: sample m data points from real data and fake data.

Gradient Ascent:

Discriminator $\rightarrow \nabla_d \frac{1}{m}\sum_{i=1}^{m}\left[log\left(D\left(x^{(i)}\right)\right) + log\left(1-D\left(G(z^{(i)})\right)\right)\right]$

Gradient Descent:

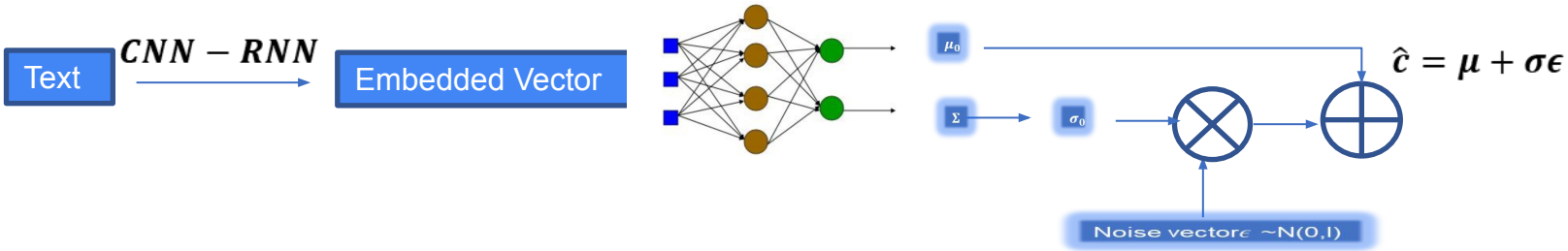
Always zero for good discriminator → vanishing gradient

Generator $\rightarrow \nabla_g \frac{1}{m}\sum_{i=1}^{m}\left[log\left(1-D\left(G(z^{(i)})\right)\right)\right]$

Solution: Generator $\rightarrow \nabla_g \frac{1}{m}\sum_{i=1}^{m}\left[log\left(-D\left(G(z^{(i)})\right)\right)\right]$

GANs for text to image: text embedding

- **$We\ construct\ \hat{c}$ by training tuning μ_0 and Σ through a neural network. Where we find mean and variance that maximize the information captured by c from the text embedding**



$$L_D = E_{(I,t)\sim p_{data}}[\boldsymbol{log}(\boldsymbol{D}(I, \boldsymbol{\phi_t}))] + E_{(z\sim p_z, t\sim p_{data})}[\boldsymbol{log}(\mathbf{1} - \boldsymbol{D}(\boldsymbol{G}(\boldsymbol{z}, \hat{\boldsymbol{c}}), \boldsymbol{\phi_t})]$$

True image Text — Text Embedding — Fake image — Noise — Conditional variable

$$L_G = E_{(z\sim p_z, t\sim p_{data})}[\boldsymbol{log}(-\boldsymbol{D}(\boldsymbol{G}(\boldsymbol{z}, \hat{\boldsymbol{c}}), \boldsymbol{\phi_t})] + \boldsymbol{\lambda} \boldsymbol{D_{KL}}(\boldsymbol{N}(\boldsymbol{\mu_0}(\boldsymbol{\phi_t}), \boldsymbol{\Sigma}(\boldsymbol{\phi_t}))\|\boldsymbol{N}(\mathbf{0}, \boldsymbol{I}))$$

Regularization term

GANs for text to image: Mutli-Stage(StackGAN)

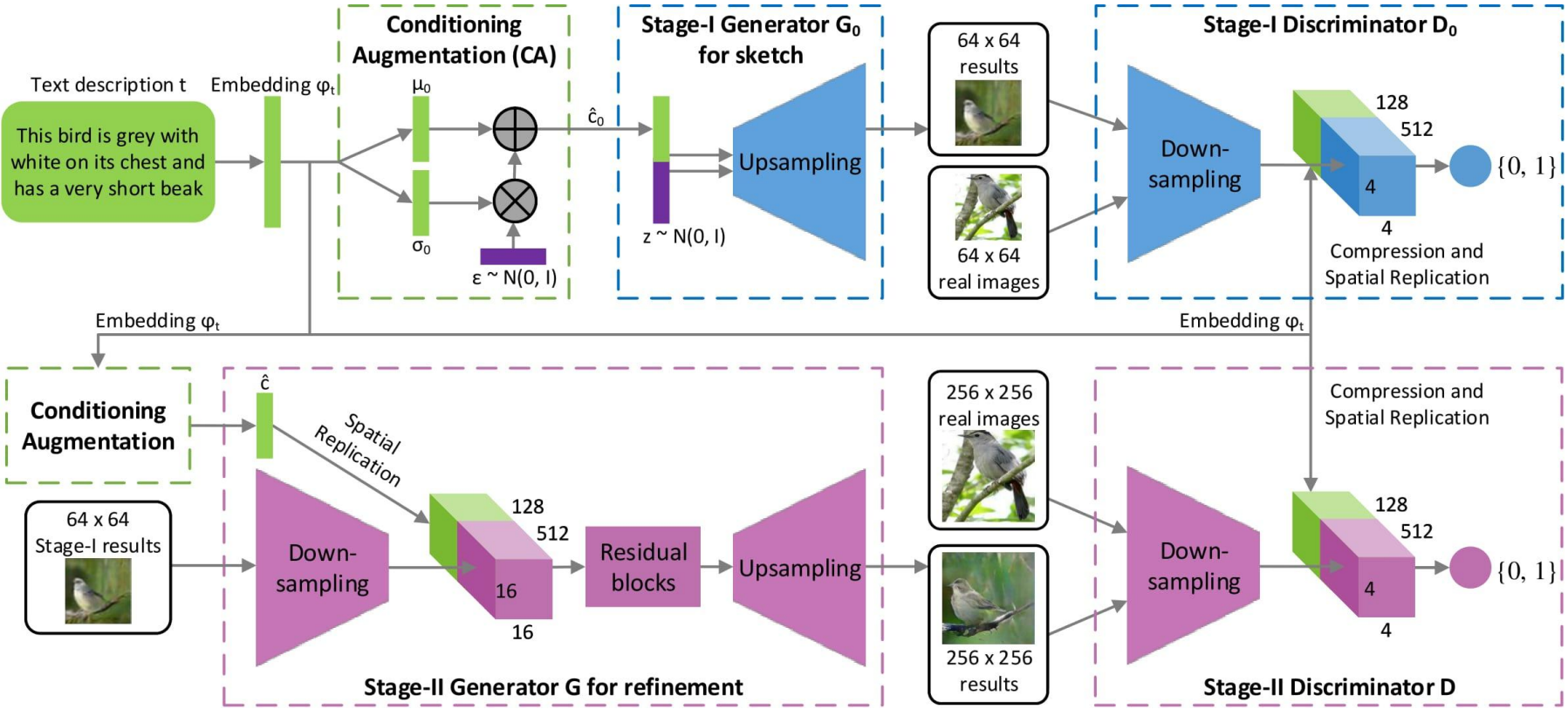


Figure 3: StackGan design Source: https://github.com/hanzhanggit/StackGAN

GANs for text to image: Mutli-Stage(StackGAN)

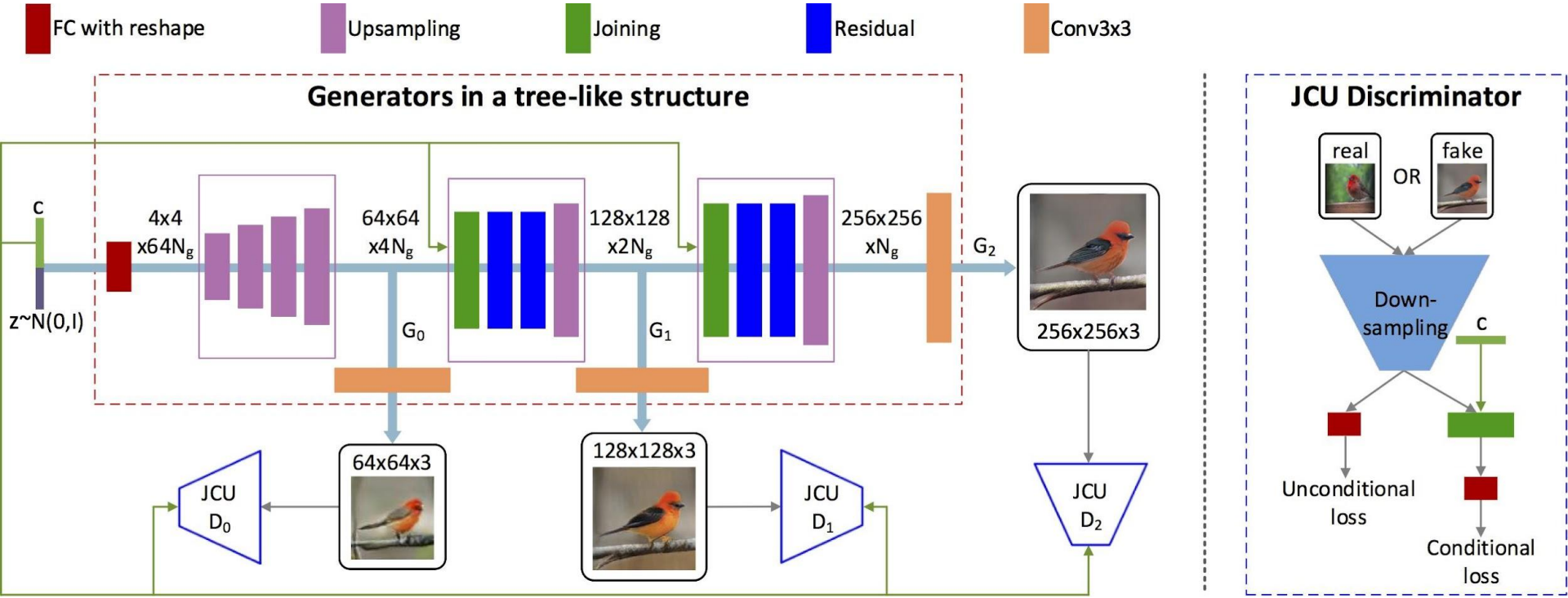


Figure 4: StackGan tree design Source: https://github.com/hanzhanggit/StackGAN

Additional enhancement

Loss function modification:

$$L_{G_i} = \underbrace{-E_{s_i \sim p_{G_i}}[\log D_i(s_i)]}_{unconditional\ loss} + \underbrace{-E_{s_i \sim p_{G_i}}[\log D_i(s_i, c)]}_{+conditional\ loss} + \lambda D_{KL}(N(\mu_0(\phi_t), \Sigma(\phi_t)) \| N(0, I))$$

$$L_{D_i} = \underbrace{-E_{x_i \sim p_{data_i}}[log\, D_i(x_i)] - E_{s_i \sim p_{G_i}}[log(1 - D_i(s_i)]}_{unconditional\ loss} + \underbrace{-E_{x_i \sim p_{data_i}}[log\, D_i(x_i, c)] - E_{s_i \sim p_{G_i}}[log(1 - D_i(s_i, c)]}_{conditional\ loss}$$

Color Consistency regularization:

$$L_{C_i} = \frac{1}{n}\sum_{j=1}^{n} \lambda_1 \left|\left|\mu_{s_i^j} - \mu_{s_{i-1}^j}\right|\right|_2^2 + \lambda_2 \left|\left|\Sigma_{s_i^j} - \Sigma_{s_{i-1}^j}\right|\right|_F^2$$

$$L_{G_i}^* = L_{G_i} + \alpha L_{C_i}$$

Implementation details

Platform:

Python 3.6+

Pytorch 1.1.0+

Hyper-parameters

Activation function: Relu for all except output layer with sigmoid.

Training Epochs=600

Learning rate=0.002

Learning rate decay=0.5/100 epochs

Optimizer: Adam with beta=0.5

- Bird data set
- ~11000 images of different species.
- Each image has 10 captions
- 8000 training
- 2900 testing

Code: https://github.com/akhilvasvani/StackGAN-v2

Training results: Generated images

the bird is short and stubby with yellow on its body

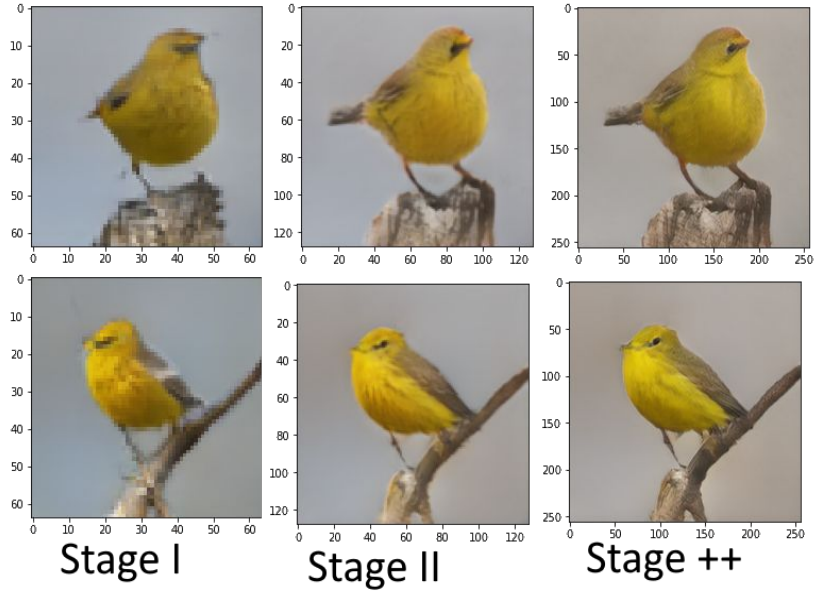


Figure 5: Results 1

this bird is red and brown in color. with a stubby beak.

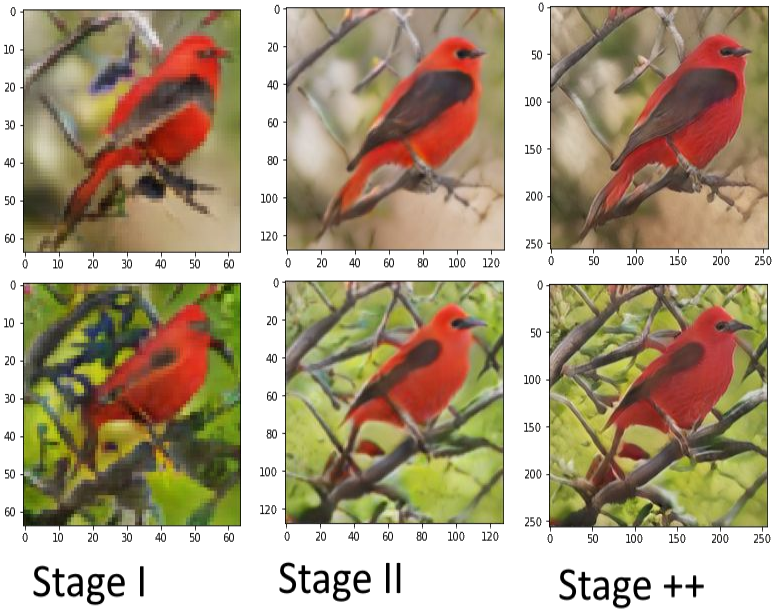


Figure 6: Results 2

Results: Sentence interpolation and collapsed modes

Sentence interpolation

$$\phi_t = \frac{1}{2}(\phi_{t_1} + \phi_{t_2})$$

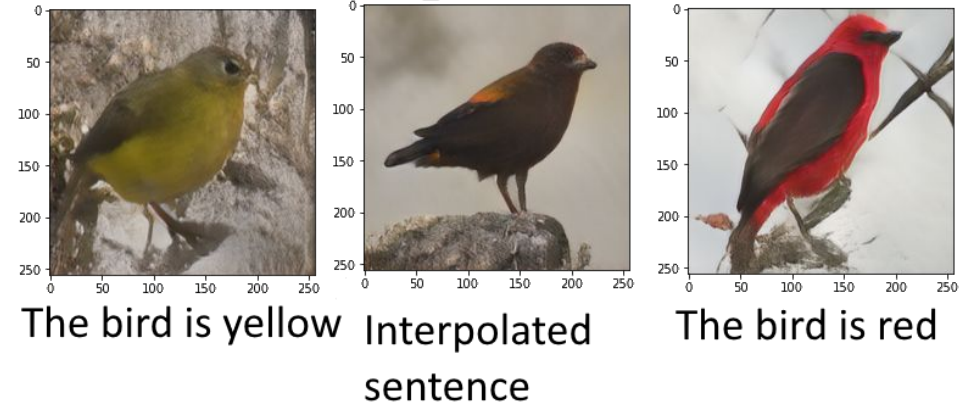


Figure 7: Sentence interpolation results

Collapsed mode (Failure case)

the medium sized bird has a dark grey color, a black downward curved beak, and long wings.

Figure 8: Failure case

Training results: Loss functions

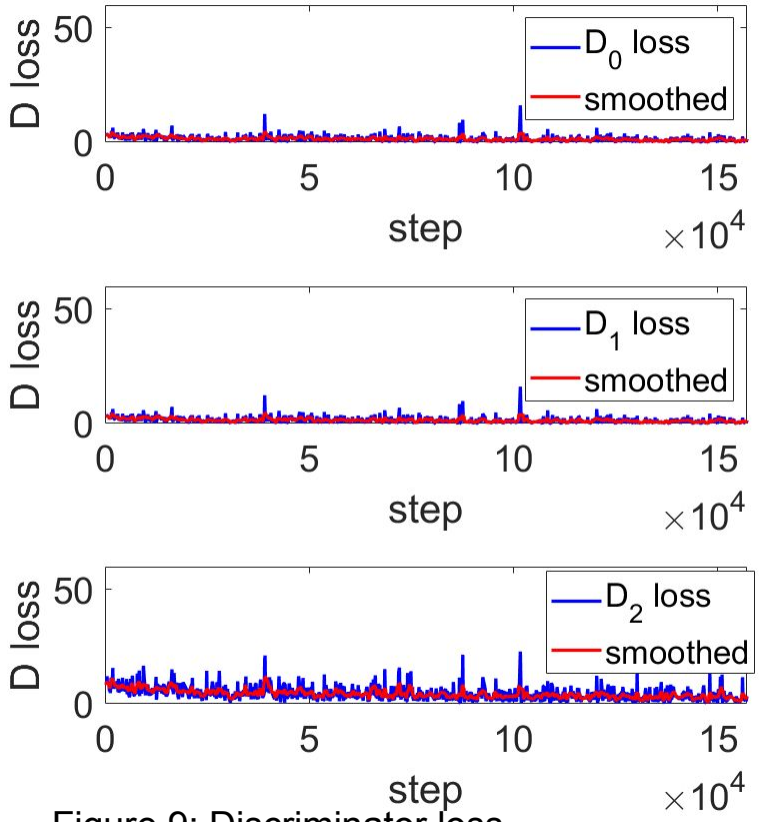


Figure 9: Discriminator loss

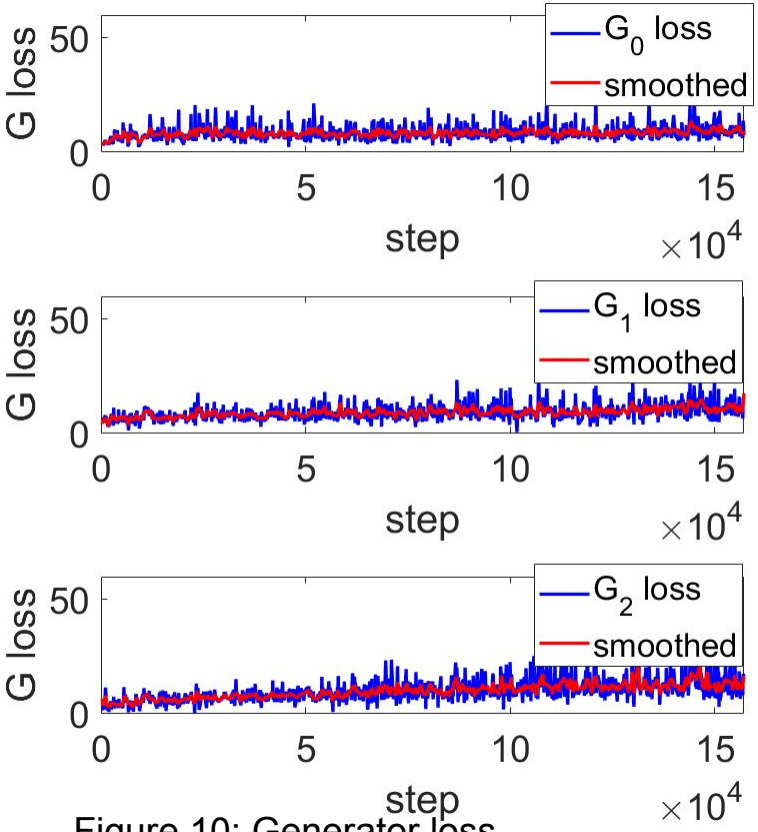


Figure 10: Generator loss

Training results: Inception score

$$IS(G) = \exp\left(E_{x\sim p_g} D_{KL}\left(p(y|\boldsymbol{x})||p(y)\right)\right)$$

IS is a measure of the quality and variety of generated images.

It measures the relative entropy between the conditional label distribution and marginal distribution.

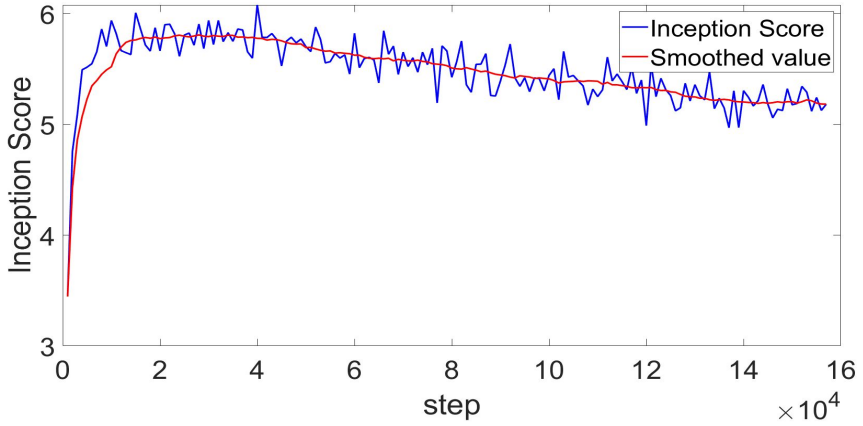


Figure 11: Inception score

Conclusion

- GANs can be used various computer generated tasks using two competing networks.
- GANs require fine tuning and regularization for optimized output
- StackGAN is a multi-stage conditional GAN framework to generate text from images.
- Promising results with large variety, but failure cases still exist.