DRAW-GAN: An Adversarial Procedure For Sequential Image Generation

Harsha Renkila
SCIS
University Of Hyderabad
Hyderabad, India
renkilaharsha1@gmail.com

Avatharam Ganivada
SCIS
University Of Hyderabad
Hyderabad, India
avathar.mtech@gmail.com

Raju S. Bapi
SCIS
University Of Hyderabad
Hyderabad, India
bapics@uohyd.ernet.in

***Abstract*—Varitional autoencoders (VAE), as generative models, generate image by sampling from approximated Guassian distribution. The generated image seems to have noise due to limited number of features captured by VAE during stochastic graident decent learning. DRAW generates image, based on VAE and RNN using attention mechanism. In this article, the concept of adversarial procedure is incorporated into the DRAW network to overcome the problem of noise in image and for better approximation of prior. By replacing the evidence lower bound (ELBO) loss function in DRAW with the adversarial loss function and adding discriminator network into DRAW, *DRAW-GAN* is developed. The performance of DRAW-GAN for generating images is compared with three differnt methods using two data sets, namely OCR Telugu and MNIST. The proposed DRAW-GAN performs better than the remaining methods.**

***Index Terms*—Variational Autoencoder (VAE), generative adversarial networks (GAN), deep recurrent attentive writer (DRAW), data generation.**

I. Introduction

Generative models are new techniques for learning the underlying distribution of any kind of data in an unsupervised fashion. They achieved a great success in the past few years. These models generate new data points that have some variations in it. Even though there are many advancements in generative models, the true distribution of data is not learned by these models. So, approximating the true distribution with the known distribution as close as possible satisfies the purpose of generative models. Advantage of deep neural networks is that they are utilised to acheive approximation of model distribution to the true disribution. Variational Autoencoders (VAE), Generative Adversarial Networks (GAN) etc., are some efficient methods for learning generative models [3].

Most of the generative models learn to generate the whole image in one step like Variational AutoEncoders and Generative Adversarial Networks. In mathematical terms all the pixels in the images are generated by conditioning on the latent distribution. The *Deep Recurrent Attentive Writer* (DRAW) takes a deviation from typical form of generation of images [2].

When a person is asked to draw or paint a picture, he/she will follow some sequential procedure like rough sketch, sharpening the rough sketch by altering or erasing and shading for sketching the entire image. In every step a refinement in the image takes place and finally a well-finished image is delivered as output. DRAW network is designed based on inspiration from the above sequential process for generating images. DRAW uses *attention mechanism* to generate images step-by-step [2].

The DRAW architecture consists of two networks – an *encoder* and a *decoder*. Both encoder and decoder networks are recurrent neural networks (RNNs). The encoder compresses images into a low dimensional space and the decoder reconstructs the image into the original dimension. Both the networks are trained together using an end-to-end stochastic gradient descent procedure, where the loss function is *Evidence lower bound (ELBO)* on the log-likelihood of data distribution. Therefore the underlying concept in DRAW is incorporating an attention mechanism using RNNs in Varitional AutoEncoders (VAE). Thus draw is able to generate images in an iterative fashion. Further, it handles selective read and write operations on images with fully differentiable network and is trained with standard backpropagation learning algorithm.

The ELBO loss function has some drawbacks like "good ElBO values do not imply accurate inference" [6]. Also, several empty pockets are there in the prior distribution without any data point mapped to them [5]. The reconstruction error, that is pixel by pixel reconstruction, is not best similarity metric for images. We know that Generative Adversarial Network's (GAN's) discriminator network has proven as good similarity metric for images by discriminating the real images from non-images [3]. We thus decided to replace pixel-by-pixel recontruction with feature-wise recontruction error and also introduced the adversarial procedure inspired from VAE-GAN [4] in DRAW to design our scheme called *DRAW-GAN*. The proposed scheme shows an improvement in the state-of-art-of generation of images in sequential manner.

II. Generative Models

A. Variational Auto Encoders

1) Latent Variable: The Latent variables are hidden variables and these are inferred from the observable variables. These are used to represent the features of original data in low dimension and to take the feature selection decision before model execution. For example, suppose if we try to generate a digit from 0-9 then, we have to decide which digit needs to

be generated. Because, if we generate one-half letter from 5 and another-half letter from 8, that generation will not make any sense. The variable z is called *latent* because we know the generating number but we do not know the settings of the model that generate the number.

2) Mathematical Formulation: As we discussed in subsection II-A1, variational autoencoder uses latent model for generation of images. Let us say $X = (x_1, x_2, x_3, \ldots, x_n)$ are the data points. For every data point in X, there must be a latent variable associated with it.

$$p(X) = \int p(X/z)p(z)d(z) \tag{1}$$

$p(z)$ is the prior distributiuon of latent variables which is assumed as gaussian distribution $\mathcal{N}(0, I)$. So, the main objective is to sample z from $p(z)$ and try to map z to a data point X. As in the equation 1, $p(X)$ is intractable because we do not know all the values of z and computation of the above integration for higher dimensions is not possible.

$$p(z/X) = \frac{p(X/z)p(z)}{p(X)} \tag{2}$$

From Bayes principle we get equation 2 where $p(z)$ and $p(X/z)$ can be calulated for a given z, but $p(X)$ is intractable. So, equation 2 cannot be evaluated. Thus finding the latent representation from input X is not possible. For finding the latent observation we can approximate $p(z/X)$ with a known distribution say $q(z)$ which is tractable.

3) Loss Function: While approximating the prior $p(z)$ with $q(z)$ we have to make sure that $p(z)$ and $q(z)$ are very close. To find the closeness of two distributions, oe can use *KL-Divergence* measure. So, here the objective is to minimize KL-divergence.

$$minimize D_{KL}(q(z)||p(z/x)) \tag{3}$$

$$\log p(x) = D_{KL}(q(z)||p(z/x)) - \sum q(x) \log p(x, z) \tag{4}$$

$$L \leq \log p(x) \tag{5}$$

$$L_{ELBO} = E_{q_\phi(z/x)}[\log p_\theta(x/z)] - D_{KL}(q_\phi(z/x)||p(z)) \tag{6}$$

$$L_{ELBO} = L_{REC} + L_{REG} \tag{7}$$

Expanding the equation 3 using KL-Divergence and by rearraging the terms, we get equation 4. $\log p(X)$ is constant because we have fixed set of inputs. KL-divergence value is always greater than zero. So, the first term in equation 4 is greater than zero so the second term is always less than or equal to $\log p(x)$ as in equation 5. Rearranging the terms in second term of equation 4 produces equation 6 which is the final loss eqaution. The first term corresponds to log likelihood of latent variable which is the reconstruction error and the second term corresponds to minimizing the prior $p(z)$ with $q(z/x)$.

4) Architecture: Due to recent advancements in neural networks, modelling probability of log-likelihood can be calculated directly from neural networks. The VAE architecture consists of two networks Encoder and Decoder as shown in figure 1. The Encoder takes the input images and produces μ and σ of the latent distribution. The decoder takes a sample from the prior and regenerates the image.

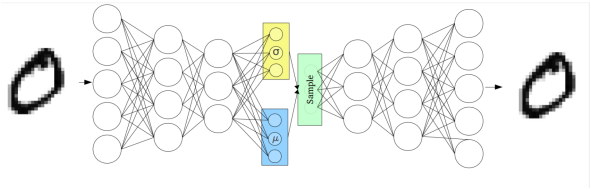

Fig. 1. VAE architecture Reprodced from [4]

5) Reparametrization Trick: Modelling variational inference through neural networks needs training of neural network. But, the encoder outputs the latent distribution $q(z/X)$ not the $p(z)$. So, to train the network we need a connection such that $q(z/X)$ values have to be sampled from $p(z)$. This sampling can be done by using a trick called *Reparametrization Trick*.

In reparametrization trick, we first sample from $\epsilon(0, I)$ and then calculate z as shown in equation 8.

$$z = \mu(X) + \sigma^{1/2}(X) * \epsilon \tag{8}$$

Training of both encoder and decoder networks is done by stochastic gradient-Descent of loss calculated from equation 6. For generation of new images, we sample from $\mathcal{N}(0, I)$ and pass it onto the decoder.

B. DRAW : A Reccurent Neural Network For Image Generation

"The underlying concept in DRAW is VAE, by incorporating Attention Mechanism using RNNs in VAE, it is able to generate images in an iterative fashion." It is proven that generated images from DRAW network are visually good and have more variation of images than other generative models such as Variational AutoEncoders and Generative Adversarial Networks [2].

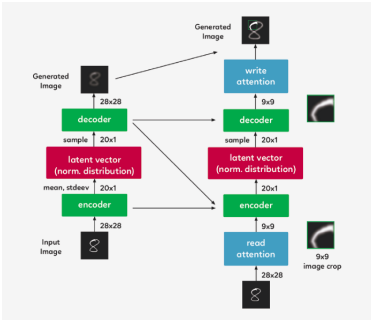

Fig. 2. Draw Architecture one step [10]

1) Architecture: DRAW architecture is as shown in figure 2. The Encoder captures sailent features of the input images over a distribution i,e,. reducing the dimension of image into a distibution which has total information about image. Generally the encoding distribution is Gaussian distribution.

The Decoder takes a sample from the distribution and learns how to reconstruct the original image. In figure 3 RNN^{enc}

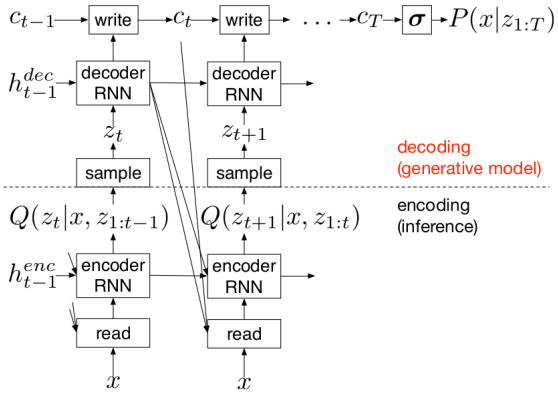

Fig. 3. Draw Architecture Reprodced from [2]

at a particular time step t encoder takes cropped patch from image and the previous-time-step-hidden-vector $h_{t-1}enc$ of encoder as input and outputs hidden vector h_tenc. RNN^{dec} takes a sample from the latent distribution and the previous-time-step-hidden-vector $h_{t-1}enc$ of the decoder.

At each time step t the output of encoder is used to parameterise the parameters of latent distribution $Q_\theta(X/Z)$. The output of the decoder will add to the canvas c_t which is the reconstructed image.

$$\hat{X} \leftarrow X - \sigma(c_{t-1}) \tag{9}$$

$$r_t \leftarrow read(X_t, \hat{X}_t, h_{t-1}^{dec}) \tag{10}$$

$$h_t^{enc} \leftarrow RNN^{enc}(h_{t-1}^{enc}, [r_t, h_{t-1}^{dec}]) \tag{11}$$

$$z_t \sim Q(Z_t|h_t^{enc}) \tag{12}$$

$$h_t^{dec} \leftarrow RNN^{dec}(h_{t-1}^{dec}, z_t) \tag{13}$$

$$c_t \leftarrow c_{t-1} + write(h_t^{dec}) \tag{14}$$

The number of time steps T and patch size N of attention area are specified in advance. At every time step T the encoder tries to encode the data into a gaussian distribution $q(z)$ and samples from the normal distribution decoded through the decoder. For every time step we try to restrict the encoder to output the distribution $q(z)/x)$ to $\mathcal{N}(0, I)$ by imposing the KL-Divergence loss. Since decoder takes a sample from the normal distribution but what we get is $q(z)$ to back propogate error stochastically . To overcome these obstacles, *Reparametrization trick* 8 is introduced to allow us to sample from the normal distribution. After all time steps are completed, reconstruction error($X - X_{recon}$) is computed and back propagated stochastically.

2) Loss Function:

$$L_{ELBO} = L_{REC} + L_{REG} \tag{15}$$

$$L_{ELBO} = E_{q_\phi(z/x)}[\log p_\theta(x/z)] - D_{KL}(q_\phi(z/x)||p(z)) \tag{16}$$

3) Generation: For generation of images we need to sample oly from normal distribution and pass this through the decoder. Thus we do not require the encoder for generation.

for number of time steps T:

$$z_t \sim \mathcal{N}(0, I)) \tag{17}$$

$$h_t^{dec} \leftarrow RNN^{dec}(h_{t-1}^{dec}, z_t) \tag{18}$$

$$c_t \leftarrow c_{t-1} + write(h_t^{dec}) \tag{19}$$

4) Attention Mechanism: The Attention mechanism requires three things as mentioned below:

- Where to Read?
- What to Write?
- Where to Write?

Why do you need attention in images?.

The image has many features, out of them there are only some useful features to play with. Attention mechanism allows us to choose the appropriate features rather than useless features in an image.

In every step of DRAW algorithm, image is cropped into patches. Each patch has centre $(g(x), g(y))$ and scale(δ). The patch has gaussain filters placed in equal distance. Each gaussian filter has mean. Gaussian filters are used to crop the image.

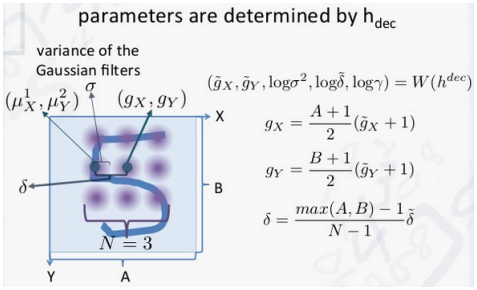

Fig. 4. DRAW-Attention Reproduceed from [2]

As shown in figure 4 the next reading patch centre is predicted by h_{dec} of decoder. In writing the image decoded to size of entire image and added to the canvas matrix.

5) Drawbacks of ELBO [6]:

- Good Elbo values do not imply accurate inference.
- $ElBO = L(Inference) + L(Reconstruction)$.
  - If we are maximising $L(Reconstruction)$, i.e., maximising log likelihood, need not simultaneously minimise the Inference.
  - No data points map to several places in the prior distribution.

- Even though the draw approximates inference very well, reconstructed images are not sharp (they are blurry).

III. Adversarial Training Procedure In DRAW Network

The DRAW network has proven the better approximation of prior over other generative models and the GAN's uses the adversarial training procedure to generate image and the generated images are sharp. So, by combaining the DRAW network and GAN network will take the advantages of both DRAW network and adversarial procedure can enhance the state of art of generation of images.

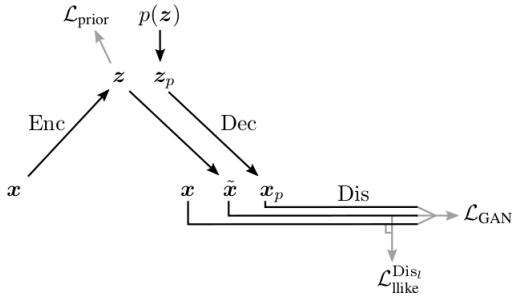

Fig. 5. DRAW-GAN Reproduced from [4]???

1) DRAW-GAN Architecture: Consider the architecture in figure 5 where, the DRAW network II-B acts as generator in the adversarial network. The Discriminator networ is added at the end of the DRAW network. The Discriminator network takes images as input and outputs features of images and classification score of image. As described in II KL-Divergence is used for approximating prior. Now in addition to that the reconstructed images are sent through discriminator to distinguish original and reconstructed images. By imposing the additional generator loss to the DRAW network, the DRAW network tries to fool discrminator and simultaneously approximate the posterior.

By adding the Discriminator to the network, it helps generator(DRAW network) to reconstruct the images very near to original images. Since we are sampling from the normal distribution to generate images. While training we sample from normal distribution and send through the decoder to get a image. This reconstructing from normal distribution helps the discriminator network to distinguish from origial images and loss function helps DRAW network to learn to fool the discriminator effectively. The generated image from normal distribution is sent through the discrimantor along with original image, the loss of discriminator is used for better approximation of prior having no gaps in mapping the data to latent normal distribution.

As shown in the figure 6, Mean Squared Error (MSE) is computed for all images. By observing the MSE of images, we came to know that even for images that are noisy and blurry, the error is the same and this is undesirable. Backpropagating this MSE would lead to poor reconstruction of images. So, in

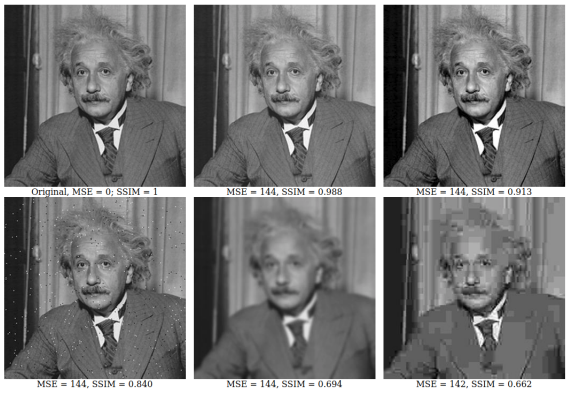

Fig. 6. MSE loss calculations Reproduced from [11]

the proposed model (DRAW-GAN) we considered the feature-wise loss L_{llike}^{Disc}, instead of MSE loss for reconstruction. Feature-wise loss is calculated from the l^{th}layer of the discriminator network.

Update equations for Discriminator and Generator(DRAW):

$$L_{GAN} = E_{x\sim p_{data}}[\log D(x)] + E_{z\sim p(z)}[\log(1 - D(z)] + E_{z\sim p(z)}[\log(1 - D(G(z)))] \tag{20}$$

$$L_{Draw} = L_{prior} + \gamma L_{llike}^{Disc} - L_{GAN} \tag{21}$$

As shown in the equation 21, we added γ, a weight parameter to weigh between the reconstruction and to fool the discriminator.

The Flow of DRAW-VAE architecture as follows:

IV. Evaluation Metrics

The main idea of generative models is to generate new data such that both original and generated data are very close. In terms of distributions the generated data is as sampled from original data distribution i.e., both the distributions need to be similar. We can say if the degree of dissimilarity of two distributions is more, then the method used for generation is not performing well. In an another way if the classifer is not able to distinguish the generated and original images, then we can say that the generator is performing well.

A. Probability Distribution Based Methods

The generated data distribution and the original data distribution have to be similar.

- Let $O(x)$ and $G(x)$ be the two distibutions of original and generated data, respectively.
- $O(x)$ and $G(x)$ are identical or the distance between the two distributions is zero (in the ideal case).

Methods For Comparing Distributions:

- Maximum Mean Discrepancy(MMD)
  - $L_{MMD} = ||\frac{1}{n}\sum_{i=1}^{n}\phi(x_i) - \frac{1}{n}\sum_{i=1}^{n}\phi(y_i)||^2$
- Frchet Inception Distance (FID)
  - $FID(P_o, P_g) = ||\mu_o - \mu_g|| + Tr(C_o + C_g - 2(C_oC_g)^{\frac{1}{2}})$

Algorithm 1 DRAW-GAN

$\theta_{Draw}, \theta_{Dis} \leftarrow$ intialise network parameters
repeat
$\mathbf{X} \leftarrow$ Random mini-batch from dataset

for t in T **do**
$\hat{X} \leftarrow X$- $\sigma(c_{t-1})$
$r_t \leftarrow read(X_t, \hat{X}_t, h_{t-1}^{dec})$
$h_t^{enc} \leftarrow RNN^{enc}(h_{t-1}^{enc}, [r_t, h_{t-1}^{dec}])$
$z_t \sim Q(Z_t|h_t^{enc})$
$L_{prior} \leftarrow D_{KL}(q_\phi(z/x)||p(z))$
$h_t^{dec} \leftarrow RNN^{dec}(h_{t-1}^{dec}, z_t)$
$c_t \leftarrow c_{t-1} + write(h_t^{dec})$
end for
$X_{Rec} \leftarrow \sigma(c_t)$

for t in T **do**
$z_{P_t} \sim N(0, I)$
$h_t^{dec} \leftarrow RNN^{dec}(h_{t-1}^{dec}, z_{P_t})$
$c_{p_t} \leftarrow c_{p_{t-1}} + write(h_t^{dec})$
end for
$X_{P_{Rec}} \leftarrow \sigma(c_{p_t})$
$L_{GAN} \leftarrow \log D(X) + \log(1 - D(X_{P_{Rec}}) + \log(1 - D(X_{Rec})$
.
// Update parameters according to gradients
$\theta_{Draw} \leftarrow$ - $\delta_{\theta_{Draw}}(L_{prior} + \gamma L_{rec} - L_{GAN})$
$\theta_{Dis} \leftarrow -L_{GAN}$
until convergence

B. Classification Based Methods

Classifier is not able to classify original and generated data.

- Using the Data Augmentation technique to combine original data and generated data in 50-50 (ratio).
- Train a classifer on augmented data.
- Test the classifier with 50-50 (ratio) of original and generated data.
- Estimate the confusion matrix and calculate $\frac{True-Negative}{Totalnoofpositives}$, $\frac{False-Positive}{Totalnoofnegatives}$ and find the maximum of both.
- Above calculation gives the misclassification rate. The model which gives greater misclassfication rate is considered the better generative model.

Some Methods for Classification:-

- KNN (K-nearest neighbour)
- Bayes Classifier

V. Experimental Results

A. MNIST Dataset

As shown in figure 7 the vae is able to generate MNIST dataset but it failed to generate OCR-Telugu Dataset.

Figure 8 shows the generation of MNIST images. Although visually there is not much difference in both the images, however as shown in table I there we find DRAW-GAN is better approximating generated distribution than different generative models.

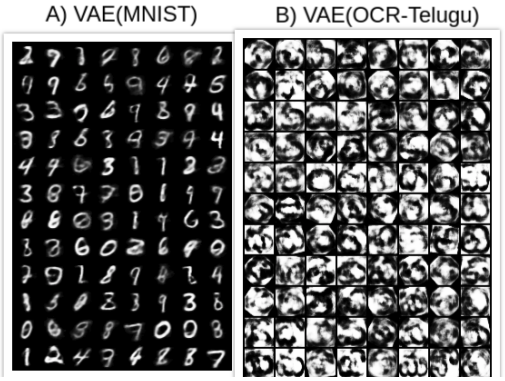

Fig. 7. VAE Images

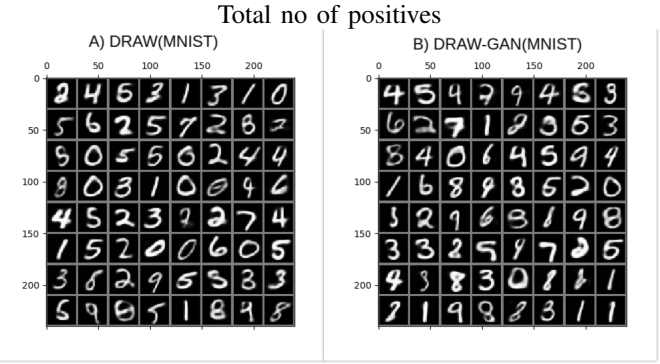

Fig. 8. DRAW and DRAW-GAN Generated images of MNIST

TABLE I
Comparing Methods with Performance Measures on MNIST

Method	MMD	FID	K-NN	Bayes
VAE	0.1464	799.6821	0.2176	0.4617
Info VAE	0.4113	1798.1730	0.1145	0.3776
DRAW	0.1057	715.1757	0.4356	0.7436
DRAW-GAN	0.0963	701.546	0.6434	0.8124

B. OCR-Telugu DataSet

OCR-Telugu is a printed charater dataset, consists of 325 classes and it is imbalanced, the no of samples of classes is different. Some classes are very less in number less than 10 and some are more than 2000 [9].

Figure 9 shows the generated images of DRAW and DRAW-GAN. We observe that there are more number of images that are unidentified (blurry) in DRAW generated images than with DRAW-GAN. Because, the OCR-Teulgu dataset is imbalanced i,e.. less no of samples for certain classes. So, the feature capturing of those classes will not bee done quiet effectively. When sampling from those classes will result in blurry images. The table II also shows the better approximation capability of generated distribution of DRAW-GAN than many other generative models.

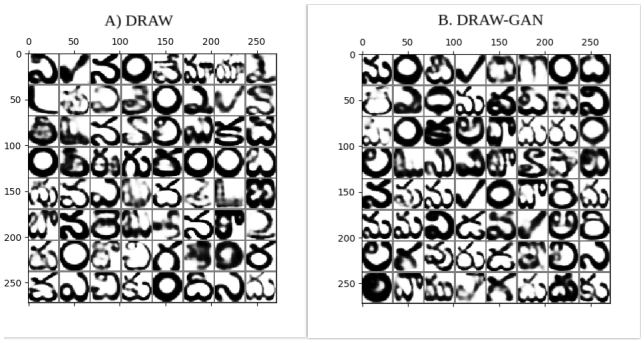

Fig. 9. DRAW and DRAW-GAN Generated images

TABLE II
COMPARING METHODS WITH PERFORMANCE MEASURES ON OCR-TELUGU

Method	MMD	FID	K-NN	Bayes
VAE	0.1916	2399.6821	0.1576	0.3268
Info VAE	0.6113	2798.1730	0.0678	0.1776
DRAW	0.1146	1889.2534	0.5545	0.6541
DRAW-GAN	0.1035	1843.6734	0.6435	0.8745

C. Trade-off Between Computational Power and Accuracy

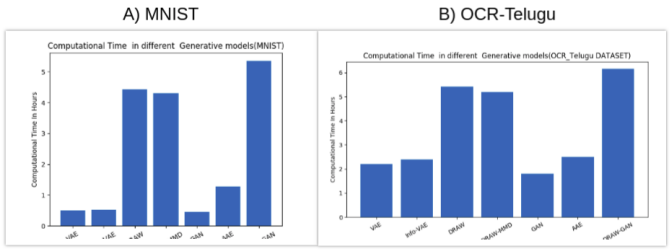

Fig. 10. compuational time

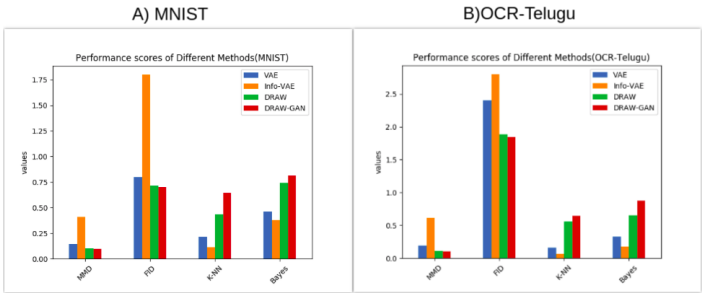

Fig. 11. Performance Scores

As shown in Figures 10 and 11, even though DRAW-GAN model achieves better accuracy for generation of images, it requires more computational power and time. So, there is a trade-off between accuray and computational time. However in most cases, generation of images is done off-line and not in real time. Hence depending on the need of the problem, one can choose an appropriate model.

VI. CONCLUSION

This paper introduced a new adversarial procedure in sequential image generation as discussed in section III, called *DRAW-GAN*. The proposed scheme shows an improvement in image generation although at a computational cost. As future work, adversarial procedure can be used for approximation of prior instead of KL-Divergence. In DRAW-GAN, we used feature-wise loss for image similarity. Instead, there are different image similarity metrics like structural similarity index measure (SSIM) that are better similarity metrics for images. Finding a way to embed SSIM as reconstruction error can enhance the qulaity of image reconstruction.

REFERENCES

[1] Diederik P. Kingma, and Max Welling, "Auto-Encoding Variational Bayes", 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings.
[2] Karol Gregor and Ivo Danihelka and Alex Graves and Danilo Rezende and Daan Wierstra, "DRAW: A Recurrent Neural Network For Image Generation", Proceedings of the 32nd International Conference on Machine Learning, pp. 1462–1471, 2015.
[3] Ian J. Goodfellow, "Generative Adversarial Networks, CoRR,2016,abs/1701.00160
[4] Anders Boesen Lindbo Larsen and Søren Kaae Sønderby and Ole Winther, "Autoencoding beyond pixels using a learned similarity metric", CoRR, abs/1512.09300, 2015, http://arxiv.org/abs/1512.09300.
[5] Alireza Makhzani and Jonathon Shlens and Navdeep Jaitly and Ian Goodfellow, "Adversarial Autoencoders",International Conference on Learning Representations 2016. http://arxiv.org/abs/1511.05644.
[6] Shengjia Zhao,Jiaming Song,Stefano Ermon. "InfoVAE: Balancing Learning and Inference in Variational Autoencoders", arXiv:1706.02262v3, [cs.LG] 30 May 2018.
[7] Yujia Li ,Kevin Swersky ,Richard Zemel, "Generative Moment Matching Networks", arXiv:1502.02761v1, [cs.LG] 10 Feb 2015.
[8] "An Emperical Study on Evaluation Metrics Of Generative Adversarial Networks", Conference Paper at ICLR May 2018.
[9] Rakesh Kummari and Bhagavati Chakravarthi ,"UHTelPCC: A Dataset For Telugu Printed Character Recognition", International conference on recent trends in Image processing and pattern recognition", 2019, unpublished.
[10] https://towardsdatascience.com/generative-adversarial-networks-explained-34472718707a.
[11] https://ece.uwaterloo.ca/ z70wang/research/ssim.