Deep Convolutional and Recurrent Writer

Sadaf Gulshad, Jong-Hwan Kim
School of Electrical Engineering
KAIST
Daejeon, Republic of Korea
Email: sadaf@rit.kaist.ac.kr, johkim@rit.kaist.ac.kr

***Abstract*—This paper proposes a new architecture Deep Convolutional and Recurrent writer (DCRW) for image generation by adapting the deep Recurrent attentive writer (DRAW) architecture which is a sequential variational auto-encoder with a sequential attention mechanism for image generation. The main difference between DRAW and DCRW is that in DCRW we have replaced RNN in encoder with CNN and after replacement attention mechanism have been used for CNN. The reason behind this modification is that CNNs are the state of the art for image processing in deep learning and their basic architecture is inspired from the visual cortex. Further, for the testing of proposed architecture experiments are performed on MNIST handwritten digits data set for generation of images and results are analyzed.**

I. Introduction

In recent years, there have been a lot of progress in supervised learning research but unsupervised learning research has not reached that level yet. As the data are growing massively in the form of images, videos, speech, text and laboratory experiments and most of the available data is unlabeled. So, in order to understand such unlabeled data, probabilistic models are used for learning the underlying structure of data. Probabilistic graphical models can represent the distribution over random variables as the graph and Bayesian networks are one among them. Bayesian networks can perform inference by learning the distribution of random variables parameterized by a set of parameters. These parameters can be learned through various ways such as gradient descent algorithm. Learning of parameters is easy if all the random variables are observable. But that is not the case in complex problems. In complex problems, where hidden variables are in a large number e.g. image pixels it becomes difficult to maximize the observed variables and consequently the posterior distribution of the network becomes intractable. Several solutions have been proposed for this problem, among them one is the mean-field approach, which is taking expectations w.r.t posterior distributions. This technique assumes that the computation of latent variables given observed is easy to compute, but this is not the case in general so as a result, the expectations become intractable. So, in order to solve this problem variational Bayesian models have been proposed, in which instead of maximizing the likelihood of observed variables lower bound of likelihood is maximized [1].

Furthermore, Deep directed generative Models have been introduced by merging the ideas of deep neural networks and Bayesian inference [2]. The deep neural networks being used with Bayesian inference in the proposed architecture are the auto-encoders. Auto-encoders are the neural networks which try to copy the input into the output, but through some restrictions in the hidden layers, i.e. it tries to copy the data which is similar to training data [3][4]. In this way, it learns the nontrivial features of data through reconstruction. The auto-encoder model combined with the Gaussian latent variable at each layer is called a variational auto-encoder (VAE). The variational auto-encoder have an inference or recognition network and a generation network.

Recently, Deep Recurrent attentive writer network (DRAW) has been proposed [5], which tries to copy the natural phenomenon of image reconstruction in sequential manner. It implements the encoder or inferential network and decoder or generation network using Recurrent neural networks with an attention mechanism during reading and writing operations. In this paper, we propose deep Convolutional Recurrent writer architecture, which is the modification of DRAW architecture and analyze the results on MNIST dataset.

The contributions of this paper are:

1) The RNN in encoder has been replaced by CNN as Convolutional Neural Networks are considered to be state of the art in image processing in deep learning.
2) After replacing RNN with CNN the attention mechanism have been introduced into the Convolutional neural network.

Recurrent Neural Network architecture used in this paper is Long Short-Term Memory (LSTM) as they are good at handling long term sequential data by eradicating the problem of vanishing gradients [6] [7].

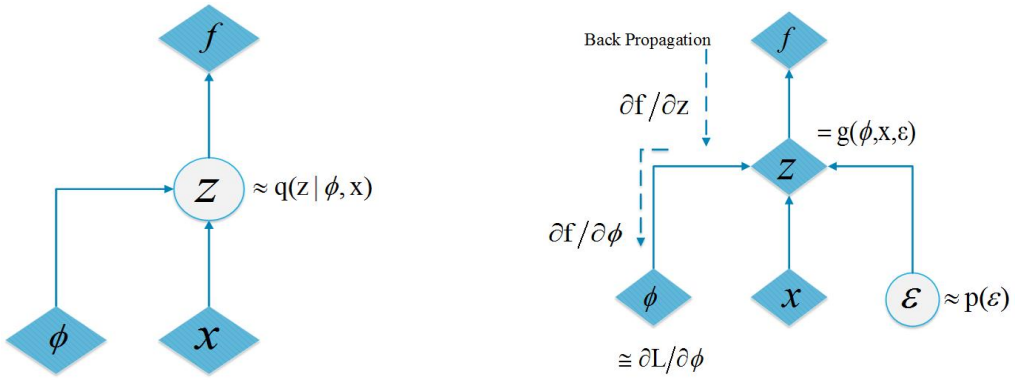
The background of Deep Convolutional and Attentive Writer is presented in section II. It is followed by the explanation of the proposed network's architecture and its equations in section III. Section IV gives the experimental results and finally discussions are given in section V.

II. Background

A. Bayesian Inference

Bayesian networks are a type of directed graphical models. They can be trained to learn the distribution of random variables. They perform inference using Bayes theorem for updating the probability given new information [8]. But while maximizing the likelihood of observed variables the Bayesian learning faces the problem of intractability due to the presence

978-1-5090-6182-2/17/$31.00 ©2017 IEEE



(a) Without Reparametrization Trick

(b) With Reparametrization Trick

Fig. 1: Directed Graphical Model of Variational Bayes

of latent variables.[9] In order to understand the Bayesian inference problem and its proposed solution, let us consider the directed graphical model shown in Fig. 1. Let x be the observed random variable, z be the hidden random variable as shown in Fig. 1a. The prior of latent variable z parametrized by θ is given by $p_\theta(z)$, where θ represents the model parameters and the intractable true posterior is given as $p_\theta(z|x) = \frac{p_\theta(x,z)}{p_\theta(x)}$. Furthermore, let $q_\phi(z|x)$ be the recognition model, which will try to approximate it to the true distribution $p_\theta(z|x)$ [9]. So, instead of maximizing the likelihood of $p_\theta(x|z)$, lower bound of likelihood is maximized.

In this scenario the lower bound of likelihood is given as:

$$log\, p_\theta(x) = D_{KL}(q_\phi(z|x)||p_\theta(z)) - \sum q_\phi(z|x) log\left(\frac{p_\theta(x,z)}{q_\phi(z|x)}\right) \quad (1)$$

$$log\, p(x) = D_{KL}(q_\phi(z|x)||p_\theta(z)) + L(q) \quad (2)$$

There are two terms on the right-hand side of equation (1), the first term is KL-Divergence term which should be minimized i.e. the difference between the approximate posterior $q_\phi(z|x)$ and real $p_\theta(z)$ should be reduced. The second term on the right-hand side is reconstruction cost. Therefore, in order to maximize the likelihood, the first term on the left-hand side of equation (1) should be minimized and the second term should be maximized. In order to do this, the network with generative parameters θ and variational parameters ϕ is trained using backpropagation. But as the approximate posterior $q_\phi(z|x)$ was obtained using sampling, therefore, it is not possible to take the derivative of $q_\phi(z|x)$ which is required for backpropagation. In order to solve this problem, stochastic gradient variational Bayes (SGVB) estimator has been introduced [1]. Which uses reparametrization trick shown in Fig. 1b. In which latent variable z is reparametrized by converting it into a deterministic function of ϕ and noise ϵ:

$$z = g(\phi, \epsilon) \quad (3)$$

Using this technique z becomes deterministic and differentiable as shown in Fig. 1b. Let the sampling of $q_\phi(z|x)$ is done from an isotropic Gaussian distribution, then the equation (3) will become:

$$z = \mu + \sigma\epsilon\ , \epsilon \sim N(0, I) \quad (4)$$

Where μ and σ mean and standard deviation respectively are the outputs of encoder network.

B. Variational Auto-encoder

The main concept behind the variational auto-encoders is to combine the idea of deep neural networks known as an auto-encoder and variational Bayesian theory. The main difference between auto-encoders and variational auto-encoders is that in variational auto-encoders the high dimensional input data represented by x and learned low dimensional data represented by z are random variables and they make it possible for the auto-encoders to sample x from the distribution $p(x|z)$, which is further used for doing the regeneration of the input. Let's consider the variational auto-encoder model shown in Fig. 2. In the figure recognition or inference $q_\phi(z|x)$ is denoted to as *encoder* since given the observed variable x it gives a distribution (e.g. Bernoulli, Gaussian) over the possible values of hidden variable z from which the observed variable x can be generated. In a similar manner $p_\theta(x|z)$ is a probabilistic *decoder* since given hidden variable z it produces a distribution of the possible corresponding values of x and thus, performs the regeneration.

C. DRAW

Deep Recurrent Attentive Writer (DRAW) has been a recently proposed neural network architecture which generates images sequentially. Its main idea constitutes of a Recurrent Neural Network based *encoder*, which encodes the images through compression at each time step and a Recurrent Neural Network based *decoder*, which decodes the compressed data from encoder [5]. It is trained using auto-encoding variational Bayes algorithm, which in turn uses the reparameterization trick for the backpropagation of error through the network. DRAW encodes the image at a time step t while observing the output of decoder from time 0 till time $t-1$. Another important

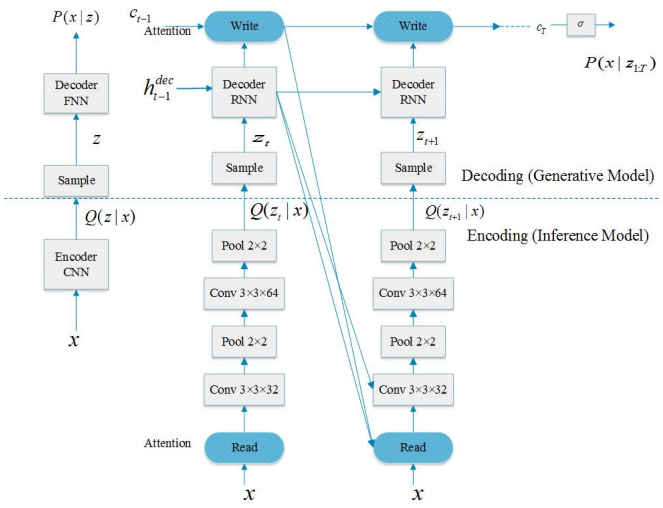


Fig. 2: Left: Convolutional and Feed Forward Writer, Right: Convolutional and Recurrent Writer with attention

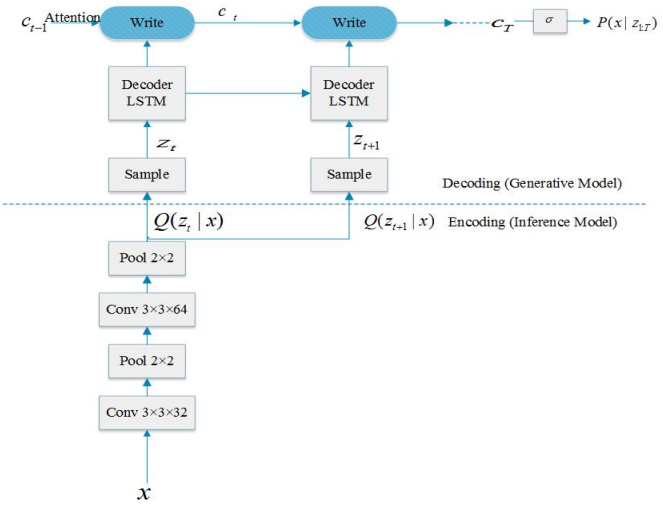


Fig. 3: Convolutional and Recurrent writer without attention in encoding

contribution of this architecture is attention mechanism. It uses a unique sequential attention mechanism, which is applied while reading the image for encoding and writing the image after decoding operation. The attention mechanism introduced in this architecture is differentiable. The architecture is trained end to end for the generation of images.

III. The DCRW Network

The proposed Deep Convolutional and Recurrent Writer (DCRW) architecture is a modification of DRAW architecture, which is explained in the previous section. The two main modifications in the architecture are the replacement of Recurrent Neural Network at the encoder by the Convolutional Neural Network as CNN's are the state of the art for feature extraction in images. The second modification is an attention mechanism applied to CNN architecture at each time step. Usually attention mechanism, which specifies where to look in the image is applied sequentially in Recurrent Neural Networks. But here it is applied at the Convolutional Neural Network at each time step t to restrict the input region being observed by the recognition network. Finally, in the recognition network, the RNN reconstructs the input at each time step. The network architecture is shown in Fig. 2.

A. Architecture

DCRW have been implemented in two different ways that are, with and without attention mechanism during the read operation. At first, the encoder is implemented using Convolutional Neural Network (CNN) and decoder is implemented using Recurrent Neural Network (RNN) with an attention mechanism in encoding (i.e. read) and decoding processes (i.e. write) as shown in Fig. 2 Right side. Next, the encoder is again implemented using CNN and decoder with RNN but without attention mechanism in read as shown in Fig. 3. These architectures will be further explained in the next sections with the theory of variational auto-encoder being used in the architecture.

B. Convolutional and Recurrent Writer with attention

Convolutional and Recurrent architecture with attention as shown in Fig. 2 right side is a deep variational auto-encoder with read and write operations. Let $X = \{x_1, x_2, ..., x_n\}$ be the input data being fed into the architecture. The encoder architecture performs $Conv(3\times3\times32)$, $Pool(2\times2)$, $Conv(3\times3\times64)$ and $Pool(2\times2)$ operations on the incoming input images. On the other hand decoder architecture is implemented using Recurrent Neural Network (RNN^{dec}).

The Convolutional neural network receives image input x and h_{t-1}^{dec} through read operation, where h_{t-1}^{dec} is the output of the previous decoder at each time step t. After extracting the features through convolution operation outputs of h_t^{enc} are used for sampling of latent distribution $z_t \sim q(z_t|h_t^{enc})$. The latent distribution used in the architecture is Gaussian distribution $N(Z_t|\mu_t, \sigma_t)$ and the mean and variance of the distribution are the outputs of encoder network given by following equations.

$$\mu_t = W(h_t^{enc}) \tag{5}$$

$$\sigma_t = \exp(W(h_t^{enc})) \tag{6}$$

The RNN network takes latent distribution as input and outputs h_t^{dec} at each time step through the write operation. This output is stored in cumulative *canvas* matrix and after T time steps $P(x|z_{1:T})$ is calculated from it. For each image x the network performs the operations defined in following equations:

$$\hat{x_t} = x - \sigma(c_{t-1}) \tag{7}$$

$$r_t = read(x_t, \hat{x}_t, h_{t-1}^{dec}) \tag{8}$$

$$h_t^{enc} = CNN^{enc}([r_t, h_{t-1}^{dec}]) \tag{9}$$

$$z_t \sim q(z_t|h_t^{enc}) \tag{10}$$

$$h_t^{dec} = RNN^{dec}(h_{t-1}^{dec}, z_t) \tag{11}$$

$$c_t = c_{t-1} + write(h_t^{dec}) \tag{12}$$

where σ is the exponential $\sigma(x) = \frac{1}{1+\exp(-x)}$ of output from decoder network.

1) Read and Write Operations with attention: In the DRAW architecture image read and write operations have been introduced with selective attention mechanism. In our architecture DCRW, we also used the same attention mechanism on the Convolutional Neural Network. The attention mechanism introduced is fully differentiable making it possible for the network to be trained end to end. The two-dimensional attention mechanism is a 2D Gaussian array applied to the image as filter yielding a patch of the image. The center of the filter is specified as g_x, g_y and stride as δ which controls the zoom of the filter. It's mean location is given by:

$$\mu_X^i = g_X + (\, i - \frac{N}{2} - 0.5)\delta \tag{13}$$

$$\mu_Y^j = g_Y + (\, j - \frac{N}{2} - 0.5)\delta \tag{14}$$

As Gaussian is being used as a filter so the variance γ of the filter is also required, which will be calculated dynamically from the output of the decoder:

$$(\hat{g_X}, \hat{g_Y}, \log\hat{\delta}, \log\gamma) = W(h^{dec}) \tag{15}$$

$$g_X = \frac{A+1}{2}(\hat{g_X} + 1) \tag{16}$$

$$g_Y = \frac{B+1}{2}(\hat{g_Y} + 1) \tag{17}$$

$$\delta = \frac{\max(A,B)-1}{N-1}\hat{\delta} \tag{18}$$

After calculating the parameters for the filter the horizontal and vertical filter matrices are calculated as:

$$F_X[i,a] = \frac{1}{Z_X}\exp(-\frac{(a-\mu_X^i)^2}{2\sigma^2} \tag{19}$$

$$F_Y[j,b] = \frac{1}{Z_Y}\exp(-\frac{(a-\mu_Y^j)^2}{2\sigma^2} \tag{20}$$

where i, j denotes a point in filter patch and a, b a point in the whole image. Hence read and write operations with attention are given by:

$$read(x, \hat{x}_t, h_{t-1}^{dec}) = \gamma[F_Y x F_X^T, F_Y \hat{x} F_X^T] \tag{21}$$

$$w_t = W(h_t^{dec}) \tag{22}$$

$$write(h_t^{dec}) = \frac{1}{\hat{\gamma}}[F_Y x F_X^T, F_Y \hat{x} F_X^T] \tag{23}$$

2) Training of Network: In order to calculate the loss for the training of network we will rewrite Variational lower bound from Equation (1) as:

$$\log p_\theta(x) = E_{q_\phi}[\log p_\theta(x|z)] - D_{KL}[\, q_\phi(z|x)||p_\theta(z)] \tag{24}$$

The first term on the right-hand side of Equation (24) is *reconstruction loss* denoted by L^x and the second term is *regularization loss* denoted by L^z. In DRAW architecture the final output is stored in canvass matrix c_T after T time steps and is used to determine the $D(X|c_T)$. Where D is the Gaussian distribution considered here. So the losses for this architecture are given by:

$$L^x = -\log D(x|c_T) \tag{25}$$

$$L^z = \sum_{t=1}^{T} D_{KL}(q(z_t|h_t^{enc})||p(z_t)) \tag{26}$$

It can be clearly observed from Equation (26) that the regularization loss depends upon the samples drawn from $q(z_t|h_t^{enc})$ which in turn depends on the input. If the latent distribution is chosen to be Gaussian, then it can be calculated analytically and is given by the formula:

$$L^z = \frac{1}{2}\left(\sum_{t=1}^{T}\mu_t^2 + \sigma_t^2 - \log\sigma_t^2\right) - \frac{T}{2} \tag{27}$$

The total loss for the network is given by:

$$L = L^z + L^x \tag{28}$$

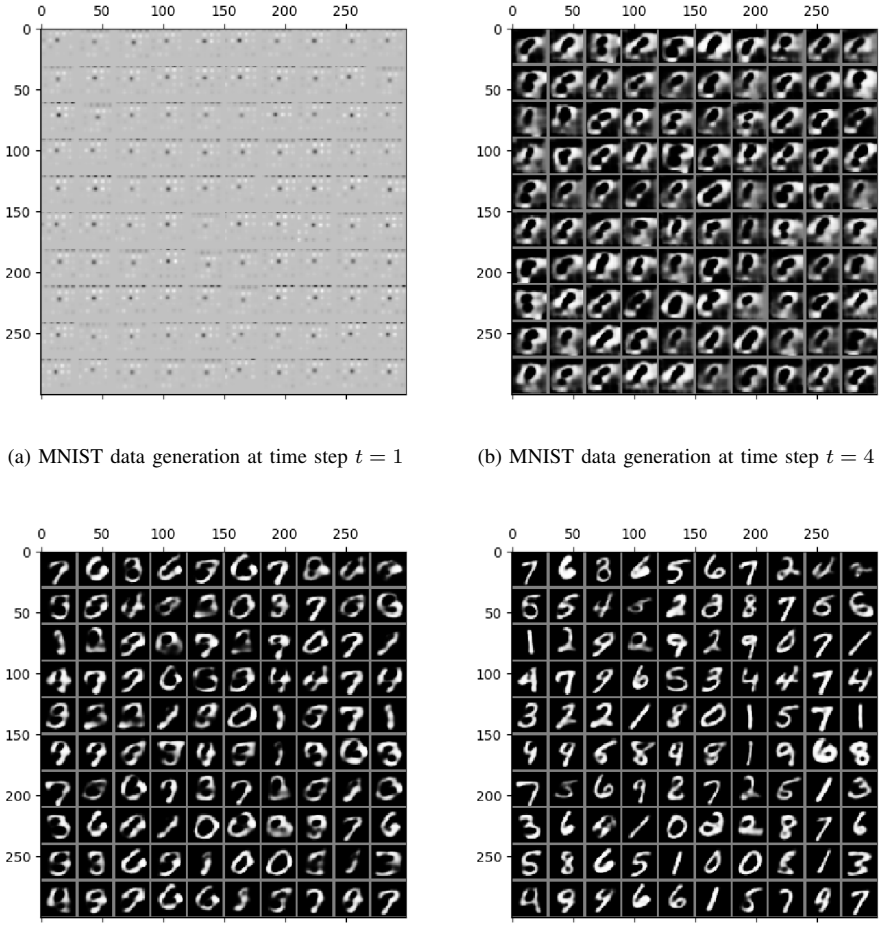
Hence, In order to maximize the likelihood of data KL Divergence should be minimized.

C. Convolutional and Recurrent Writer without Attention in Read

Convolutional and Recurrent auto-encoder architecture without attention at read is shown in Fig. 3. It differs from the architecture with attention in following aspects: Firstly, attention mechanism is a sequential process through which it reads the image but as the attention mechanism is removed from encoder so the sequential read operation is also removed along with it. Secondly Convolutional network is not being used for T time steps instead CNN extracts features from input in a single time step. But the sampling of the output of CNN is being performed at each time step t for T time steps. So in this case T at encoder network is one. After sampling z_t from the output of encoder h^{enc} it is given as input to decoder Long Short Term Memory Network (LSTM) for decoding at each time step t. The decoder will give its output h_t^{dec} to write module as input, which will keep writing at each time step t in canvass matrix. Attention mechanism is still present in writing but has been removed while reading the image in this architecture.

The modified equations for the Deep Convolutional Recurrent Writer without attention while reading is given by:

$$h^{enc} = CNN^{enc}(x) \tag{29}$$



(a) MNIST data generation at time step $t = 1$

(b) MNIST data generation at time step $t = 4$

(c) MNIST data generation at time step $t = 6$

(d) MNIST data generation at time step $t = 10$

Fig. 4: MNIST data generation

$$z_t \sim q(z_t|h^{enc}) \tag{30}$$

$$h_t^{dec} = RNN^{dec}(h_{t-1}^{dec}, z_t) \tag{31}$$

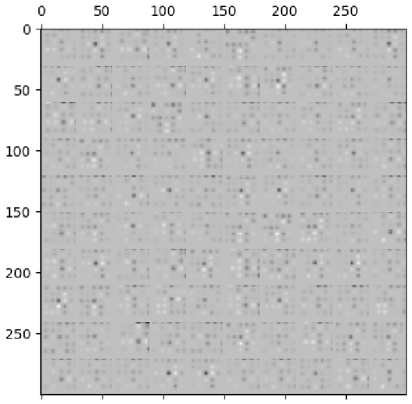
$$c_t = c_{t-1} + write(h_t^{dec}) \tag{32}$$

It can be clearly observed from the above modified equations that now input image x is directly given to the Convolutional Neural Network, and after the computation of h^{enc} sampling is done at each time step from the CNN output. Then, using these samples decoding process is performed by RNN at each time step.

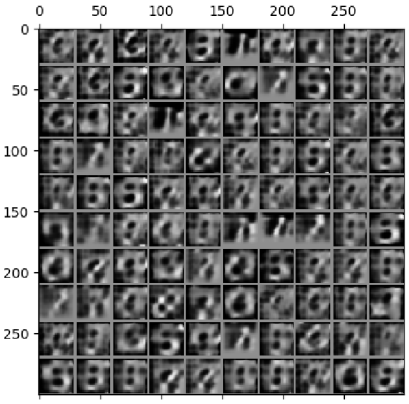
1) Read without Attention and Write with Attention: In this architecture, the encoding operation is implemented without using attention mechanism. So the equations for attention mechanism introduced in DRAW are being used only for write operation, i.e. Gaussian filter matrices $F_X[i, a]$ and $F_Y[j, b]$ are applied only while writing as given in Equation (19), (20) and (23). The equation for encoding without attention mechanism is given below:

$$read(x) = CNN^{enc}(x) \tag{33}$$

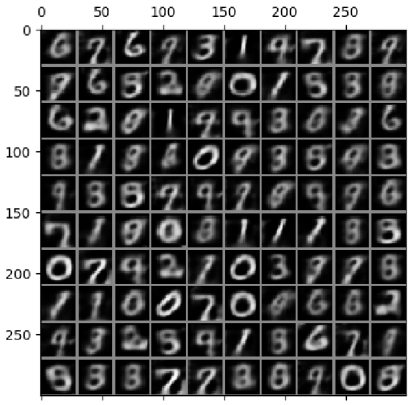
2) Training of Network: The loss function for the training of this architecture without the attention at the input will be same as the architecture with attention at the input. Because, still the architecture is Variational auto-encoder as it carries the basic idea of auto-encoder neural networks and Variational Bayes. Furthermore, although we are using CNN only at one time step, but the sampling from its output is being done for T time steps or T times therefore, the loss equations will still be same and the total loss will consist of *regularization loss* calculated analytically as we are using Gaussian, and *reconstruction loss* calculated from sampling. Hence the loss equations will be same for architecture with attention and without attention mechanism. The loss equations will be optimized using a stochastic gradient descent algorithm.



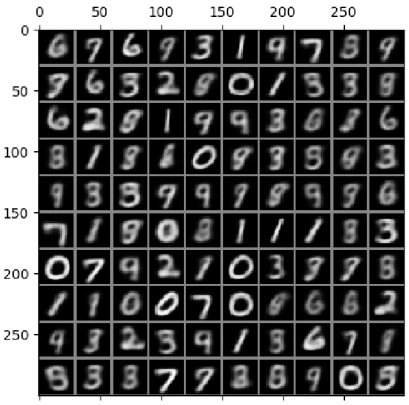
(a) MNIST data generation at time step $t = 1$



(b) MNIST data generation at time step $t = 4$



(c) MNIST data generation at time step $t = 6$



(d) MNIST data generation at time step $t = 10$

Fig. 5: MNIST data generation with no attention in Read

D. Data Generation

Data is generated by the decoder using latent samples of distribution $z_t \sim q(z_t|h_t^{enc})$. The output of the decoder updates canvass matrix C_t at each time step t and after T time steps the data is generated using $D(\hat{x}|C_T)$. Where $\hat{x}$ is the newly generated image. There is no contribution of the encoder network while generating the images.

IV. Experimental Results

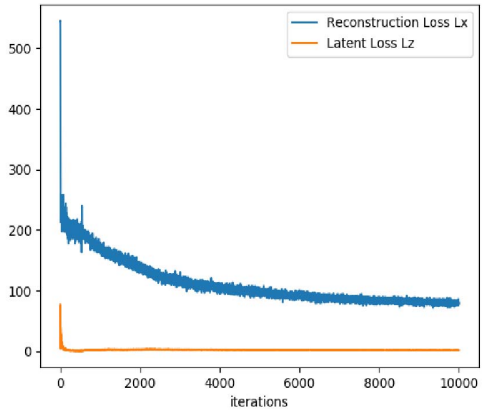
In order to evaluate the proposed architectures, they were trained to encode and then decode (regenerate) the MNIST dataset images. As the sampling operation performed at each time step t during the training of architecture was always unique, therefore the images generated are also novel and indistinguishable from training samples of MNIST dataset. The reconstruction loss used was binary cross-entropy. The network parameters and losses, i.e. *reconstruction loss* and *regularization loss* for each network are given in Table I.

Experimental Hyper Parameters and Losses				
Architecture	LSTM No of Hidden Units	Read Size	Write Size	Losses
Convolutional and Recurrent Writer with attention	256	7×7	5×5	$L_x = 70.168$ $L_z = 1.5468$
Convolutional and Recurrent Writer without attention in Read	256	-	5×5	$L_x = 147.24$ $L_z = 31.065$

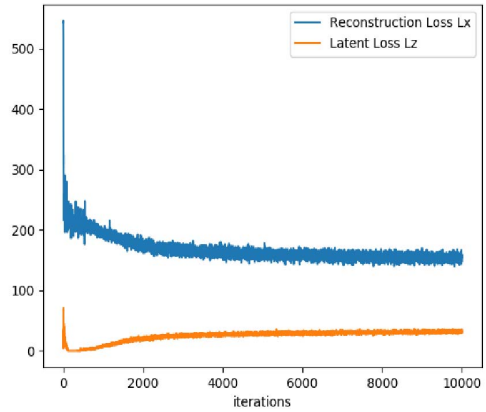
TABLE I: Networks Hyper-Parameters and Losses

A. MNIST Data Generation using Convolutional and Recurrent Network with Attention

Convolutional and Recurrent Neural network is trained end to end as the generative model on MNIST dataset. MNIST is the large database with 60,000 training images and 10,000 test images commonly used for image processing experiments. Once the Convolutional Recurrent Neural network architecture



(a) Reconstruction and Regularization loss for DCRW



(b) Reconstruction and Regularization loss for DCRW with no attention in read

Fig. 6: Reconstruction loss (L_x) and Regularization loss (L_z)Plots

is trained data generation operation is performed, the regenerated MNIST images at time steps $t = 1, t = 4, t = 6, t = 10$ are shown in Fig. 4 . We can observe from the generated images that although the architecture was simple with only two Convolutional and two pooling layers in encoder network and Recurrent Neural Network on the decoder side, it was able to reconstruct the readable images of MNIST numbers. The plot of regularization and reconstruction loss is also shown in Fig. 6a and its values are given in Table I. We can clearly observe that losses decrease with the increase in the number of iterations and reaches to the values comparable with DRAW architecture.

B. MNIST Data Generation using Convolutional and Recurrent Network without Attention in Read

Final experiment of MNIST generation is performed using the Convolutional and recurrent networks without attention in encoding process as shown in Fig.3. Although in this architecture input is encoded at single time step using Convolutional neural network and samples are decoded using RNN in T time steps. We can observe from the Figure 5 that the network was able to reconstruct the images, although not as sharpened as with attention mechanism. Another observation made from the Figure 5 is that images are updated globally when the attention mechanism is removed from encoder, while in the DCRW with attention at both read and write images they were updated locally i.e. sequentially. Further, loss plots for the deep Convolutional Recurrent writer (DCRW) without attention in reading are shown in Fig. 6b. Similar to the results of regeneration in Fig. 5 the loss plot also shows that the loss of DCRW model without attention mechanism at reading is higher and hence the images are a bit blurrier than those of with attention mechanism.

V. Conclusion

In this paper, we proposed deep Convolutional and Recurrent writer architecture (DCRW) which is the modification of the deep Recurrent attentive writer (DRAW). We replaced the encoder of DRAW from Recurrent neural network to Convolutional Neural Network and applied attention mechanism on CNN. The logic behind changing encoder from RNN to CNN is that CNNs are the state of art for image processing in deep learning applications and their architecture is biologically inspired from the visual cortex. The experimental results show that DCRW gave the comparative results in the experiments.

VI. Acknowledgment

This work was supported by the ICT R&D program of MSIP/IITP [2016-0-00563, Research on adaptive machine learning technology development for intelligent autonomous digital companion].

References

[1] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[2] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," *arXiv preprint arXiv:1401.4082*, 2014.

[3] I. Goodfellow, Y. Bengio, and A. Courville, "Deep learning," 2016, book in preparation for MIT Press. [Online]. Available: http://www.deeplearningbook.org

[4] C.-Y. Liou, J.-C. Huang, and W.-C. Yang, "Modeling word perception using the elman network," *Neurocomputing*, vol. 71, no. 16, pp. 3150–3157, 2008.

[5] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra, "Draw: A recurrent neural network for image generation," *arXiv preprint arXiv:1502.04623*, 2015.

[6] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[8] K. P. Körding and D. M. Wolpert, "Bayesian decision theory in sensorimotor control," *Trends in cognitive sciences*, vol. 10, no. 7, pp. 319–326, 2006.

[9] T. Griffiths and A. Yuille, "Technical introduction: A primer on probabilistic inference. ucla. department of statistics papers no. 2006010103. ucla, los angeles, ca," 2006.