Generative Image Models

Midterm Report, Group 11, 4 Oct 2016

Nirbhay Modhe (13444)   Vikas Jain (13788)
Department of Computer Science and Engineering
CS698N: Recent Advances in Computer Vision, Autumn 2016, Instructor: Prof Gaurav Sharma

1 Introduction

The problem of designing generative image models is important as it leads to more efficient representations of images. In this project, we studied the generative image model *DRAW* proposed by Gregor *et al.* [3]. In this model, probability distribution of the latent representation of images are learned using *Variational Autoencoders* [5]. The key features of the model include *progressive refinement* of the generated image and a *spatial attention* mechanism. The autoencoder is composed of two *Recurrent Neural Networks* and the attention mechanism is composed of *read* and *write* functions which direct the focus of the autoencoder on parts of the image rather than the whole.

In this project, we aim to experimentally analyze the *DRAW* model. We also aim to change (and possibly enhance) the model by changing the model architecture or incorporating other generative models (DARN [4], GAN [2], DCGAN [10], AE [8]). Till the making of this report *i.e.* before the mid-term, we have

- Experimentally analyzed the *DRAW* model by identifying the effects of reading attention, writing attention and the effect of allowing the encoder to look at past decoder output.
- Changed the model architecture to add *convolution* and *deconvolution layers* before the *RNN encoder* and after the *RNN decoder* respectively in the *DRAW* model.

2 Dataset Used

We have used the binarized *MNIST Dataset* [7] for the purpose of our experiments. The *DRAW* model also used this dataset, along with the Street View House Numbers (SVHN)[9] and CIFAR-10[6] datasets. The low dimensionality and compatibility of the MNIST dataset with our setup of image generation made it a suitable choice.

3 Existing Code and Libraries Used

Several open source implementations of the DRAW model are available on varied platforms. We have used the existing implementation of DRAW in TensorFlow documented and licensed by Eric Jang[1]. This implementation was limited to the binarized MNIST dataset. There were no other shortcomings in terms of replicating the published DRAW model.

4 Methodology

4.1 Experimental Study of the *DRAW* Model

We trained the original *DRAW* model for the parameters described in the paper [3]. It took nearly 10 minutes for 10000 training iterations each with a batch size of 100 images to train the original *DRAW* model with an encoder and decoder size of 256 hidden units each, 10 time steps, latent z vector of size 10, and a glimpse window size of 5x5. Listed below are the experiments done with different parameters of the model.

Generating original *DRAW* model results: We trained the original *DRAW* model with and without attention.

Varying time-step T of RNN encoder and decoder network We trained the model for $T = 1, 2, 5$ and 10. The case in which $T = 1$ reduces to a model similar to the variational autoencoder [5].

Removing decoder output from encoder input: In the original *DRAW* model, the output of decoder is made privy to the encoder with no explanation. We trained a model in without providing decoder ouput to the encoder. The original equation is:

$$h_t^{enc} = RNN^{enc}(h_{t-1}^{enc}, [r_t, h_{t-1}^{dec}])$$

And the modified equation afer removing decoder output is:

$$h_t^{enc} = RNN^{enc}(h_{t-1}^{enc}, [r_t])$$

Only error image $\hat{x}_t$ in encoder input: The original *DRAW* model takes as input both the original image x and the error image $\hat{x}_t = x - \sigma(c_{t-1})$.

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

Since $\hat{x}_{t-1}$ already contains information about x, we removed the original image x from the input for training the model.

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

4.2 Our Modification – Adding *Convolution* and *Deconvolution* Layers

The architecture of the *DRAW* model uses RNNs for encoding the output of their $read$ function as well as decoding of the *latent representation* of the image required by the $write$ function. This encoding and decoding part of the network resembles *variational autoencoders* with their standard loss – *reconstruction loss* and *latent loss*. Figure 1 shows the *DRAW* model unfolded temporally.

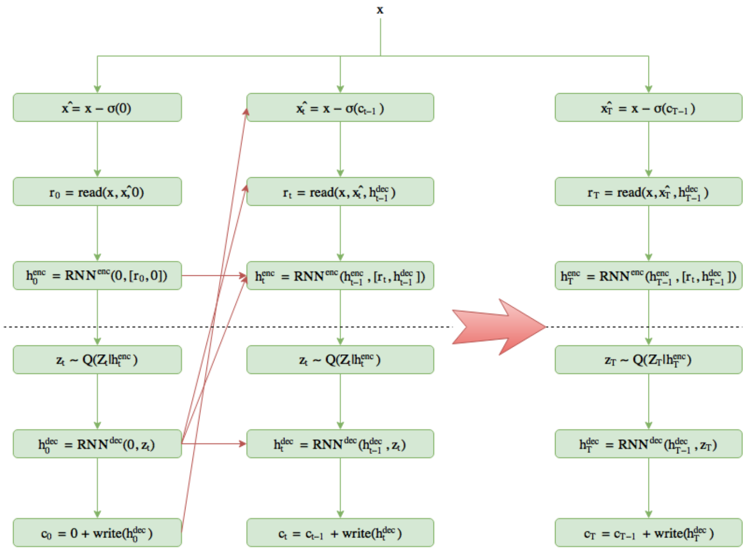

x
$\hat{x} = x - \sigma(0)$
$\hat{x_t} = x - \sigma(c_{t-1})$
$\hat{x_T} = x - \sigma(c_{T-1})$
$r_0 = read(x, \hat{x}0)$
$r_t = read(x, \hat{x_t}, h_{t-1}^{dec})$
$r_T = read(x, \hat{x_T}, h_{T-1}^{dec})$
$h_0^{enc} = RNN^{enc}(0, [r_0, 0])$
$h_t^{enc} = RNN^{enc}(h_{t-1}^{enc}, [r_t, h_{t-1}^{dec}])$
$h_T^{enc} = RNN^{enc}(h_{T-1}^{enc}, [r_t, h_{T-1}^{dec}])$
$z_t \sim Q(Z_t | h_t^{enc})$
$z_t \sim Q(Z_t | h_t^{enc})$
$z_T \sim Q(Z_T | h_T^{enc})$
$h_0^{dec} = RNN^{dec}(0, z_t)$
$h_t^{dec} = RNN^{dec}(h_{t-1}^{dec}, z_t)$
$h_T^{dec} = RNN^{dec}(h_{T-1}^{dec}, z_T)$
$c_0 = 0 + write(h_0^{dec})$
$c_t = c_{t-1} + write(h_t^{dec})$
$c_T = c_{T-1} + write(h_T^{dec})$

Figure 1: The original *DRAW* model of Recurrent Neural Networks unfolded temporally.

The modifications we made to the original model are as follows.

Convolution After Read: The read function (without attention) now passes the training image and error image through two convolution layers with 32 filters of size 5x5 in each layer. The flattened filter maps (outputs of the convolution layers) are then used subsequently in two possible way – they are concatenated with the source image (and error image respectively) or they are returned unchanged. The results for both these cases are tabulated in the next section.

Deconvolution Before Write: The decoder output is first passed through two deconvolution (also known as fractionally strided convolution) layers which restore the filter maps to their original image shape, which are then passed on to the write function (without attention).

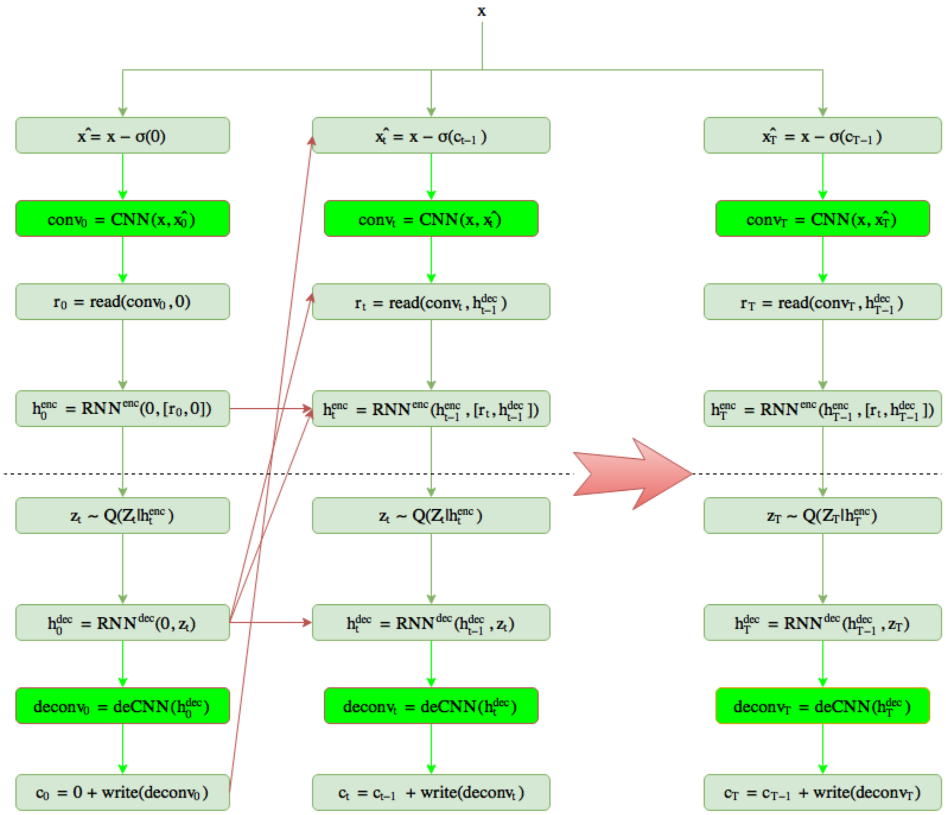

x
$\hat{x} = x - \sigma(0)$
$\hat{x_t} = x - \sigma(c_{t-1})$
$\hat{x_T} = x - \sigma(c_{T-1})$
$conv_0 = CNN(x, \hat{x_0})$
$conv_t = CNN(x, \hat{x_t})$
$conv_T = CNN(x, \hat{x_T})$
$r_0 = read(conv_0, 0)$
$r_t = read(conv_t, h^{dec}_{t-1})$
$r_T = read(conv_T, h^{dec}_{T-1})$
$h^{enc}_0 = RNN^{enc}(0, [r_0, 0])$
$h^{enc}_t = RNN^{enc}(h^{enc}_{t-1}, [r_t, h^{dec}_{t-1}])$
$h^{enc}_T = RNN^{enc}(h^{enc}_{T-1}, [r_t, h^{dec}_{T-1}])$
$z_t \sim Q(Z_t | h^{enc}_t)$
$z_t \sim Q(Z_t | h^{enc}_t)$
$z_T \sim Q(Z_T | h^{enc}_T)$
$h^{dec}_0 = RNN^{dec}(0, z_t)$
$h^{dec}_t = RNN^{dec}(h^{dec}_{t-1}, z_t)$
$h^{dec}_T = RNN^{dec}(h^{dec}_{T-1}, z_T)$
$deconv_0 = deCNN(h^{dec}_0)$
$deconv_t = deCNN(h^{dec}_t)$
$deconv_T = deCNN(h^{dec}_T)$
$c_0 = 0 + write(deconv_0)$
$c_t = c_{t-1} + write(deconv_t)$
$c_T = c_{T-1} + write(deconv_T)$

Figure 2: Our modified model with changes highlighted in green boxes – adding *convolution* and *deconvolution* layer

5 Results and Discussions

5.1 Experimental Study of the *DRAW* Model

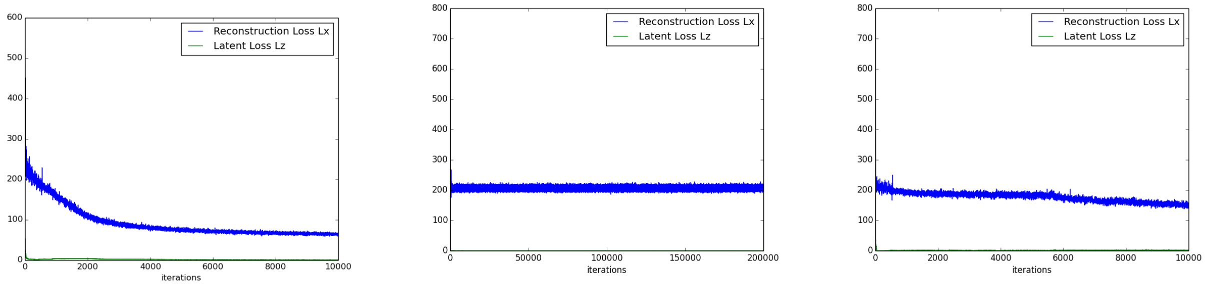
We calculated value of *Negative log-likelihood* of 1000 generated images for the original model and the modified models described in the section 4.1.

	Average Error	Standard Deviation	Maximum Error (≤)
***DRAW*(Attention)**	**643.09**	**264.67**	1370.03
***DRAW*(No-Attention)**	644.61	268.30	**1337.80**
***DRAW*(T=1)**	653.31	290.03	1724.63
***DRAW*(T=2)**	698.17	276.37	1402.27
***DRAW*(T=5)**	796.44	302.79	1853.58
***DRAW*(no-privy[1])**	809.61	298.31	2546.65
***DRAW*(only error image)**	648.66	302.79	1853.58

It can be observed that ***DRAW* model with attention** is performing better than the other models which is the proposed model of the paper [3].

5.2 Generated images and convergence graphs

Figure 4 shows samples of randomly generated images for all of our modifications to DRAW. Convergence plots for a few models have been showin in Figure 3. The subfigures 4i,4h show the generated images for our models. It can be seen from the generated images and convergence plot3b that the network was unable to learn any representation given just the convolution filter outputs for encoding and decoding.



(a) DRAW (Original)

(b) CONV - No Concat

(c) CONV - Concat

Figure 3: Training loss convergence plots for selected models

6 Future Work

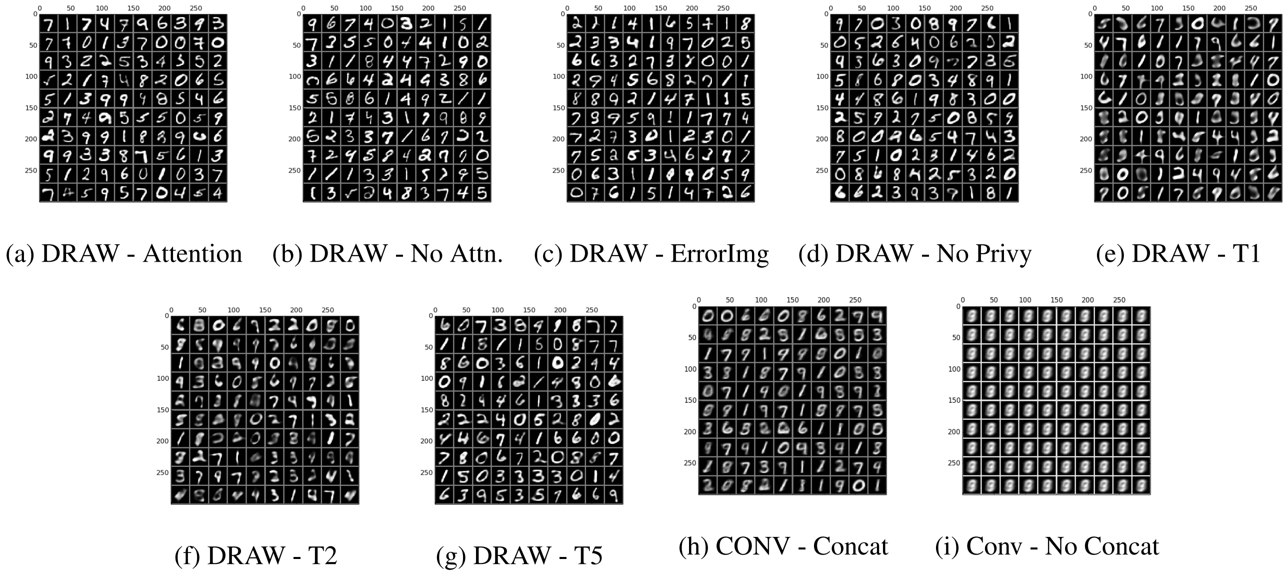
In the future work, we wish to study and understand other generative image models (DARN [4], GAN [2], DCGAN [10], AE [8]). The variational autoencoding framework upon which the DRAW model's autoencoder is based has been studied and improved upon soon after the DRAW model was published. We are interesed in trying to incorporate these improvements into the DRAW network. We would also like to test our current modified model as well as the original *DRAW* model on a more complex dataset (SVHN [9] or CIFAR-10 [6] datasets) whose probability distributions are much more difficult to learn.

References

[1] Eric jang: Understanding and implementing deepmind's draw model. `http://blog.evjang.com/2016/06/understanding-and-implementing.html`. Accessed: 2016-10-2.

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[1] decoder output is not made privy to the encoder



(a) DRAW - Attention (b) DRAW - No Attn. (c) DRAW - ErrorImg (d) DRAW - No Privy (e) DRAW - T1

(f) DRAW - T2 (g) DRAW - T5 (h) CONV - Concat (i) Conv - No Concat

Figure 4: Generated images using the original DRAW model, its parameter variations and our modifications. (a) to (g) consists of the original DRAW's paramater variations, (h) and (i) are our modifications. T refers to the number of time steps, no-privy means the previous decoder output is not made privy to the encoder input. In CONV, concat and no-concat refer our model's read output - the filter map with and without the source image concatenated to it respectively

[3] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

[4] Karol Gregor, Ivo Danihelka, Andriy Mnih, Charles Blundell, and Daan Wierstra. Deep autoregressive networks. *arXiv preprint arXiv:1310.8499*, 2013.

[5] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[6] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

[7] Yann LeCun, Corinna Cortes, and Christopher JC Burges. The mnist database of handwritten digits, 1998.

[8] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian Goodfellow. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

[9] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[10] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.