CS698N State of the Art Presentation

DRAW: A Recurrent Neural Network For Image Generation by Gregor et. al. (Google DeepMind)

Nirbhay Modhe   Vikas Jain

Department of Computer Science
IIT Kanpur

December 22, 2016

Outline

Problem Statement

- Learning generative model for images.
- Inspired from **DRAW: A Recurrent Neural Nerwork for Image Generation** by Gregor et al. (Google DeepMind)

Outline

DRAW Model

- *Encoder* : compresses the real images presented during training into **latent codes**
- *Decoder* : **reconstitutes** images after receiving codes
- A pair of **Recurrent Neural Networks** for both the encoder and decoder networks.
- Family of *variational auto-encoders*
- Encoder iteratively accumulate the modifications emitted by the decoder network.

DRAW Model Network

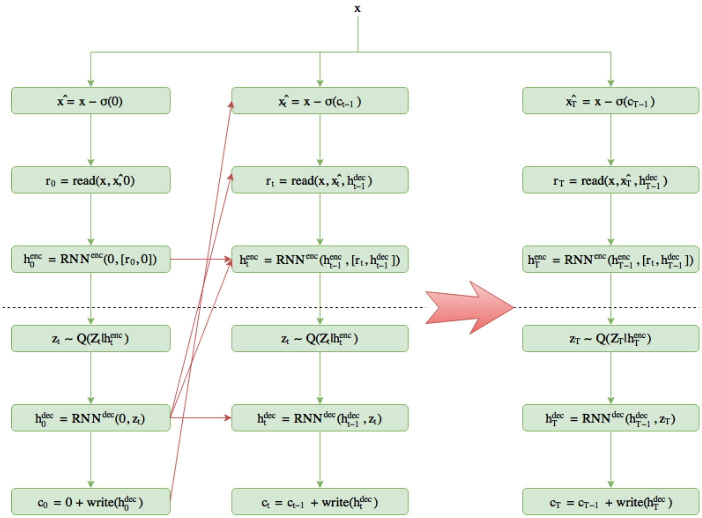


Figure: *DRAW* Network Architecture

Outline

Overall Idea of DRAW

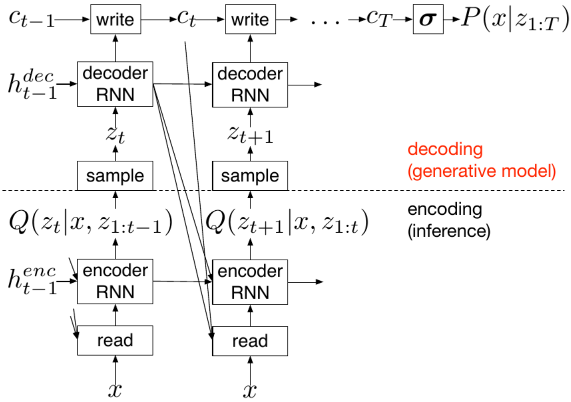
- *Encoder* : compresses the real images presented during training into **latent codes**
- *Decoder* : **reconstitutes** images after receiving codes
- A pair of **Recurrent Neural Networks** for both the encoder and decoder networks.
- Family of *variational auto-encoders*
- Encoder iteratively accumulate the modifications emitted by the decoder network.

Outline

The Draw Network - Architecture

Encoder Equations:

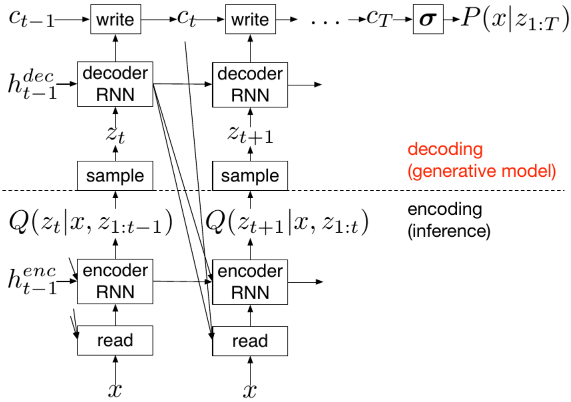
- Input image x
- $\hat{x}_t = x - \sigma(c_{t-1})$
- $r_t = read(x, \hat{x}_t, h^{dec}_{t-1})$
- $h^{enc}_t = RNN^{enc}(h^{enc}_{t-1}, [r_t, h^{dec}_{t-1}])$



The Draw Network - Architecture

Output of the encoder h_t^{enc} is used to parameterize the distribution $Q(Z_t|h_t^{enc})$ over the latent vector z_t

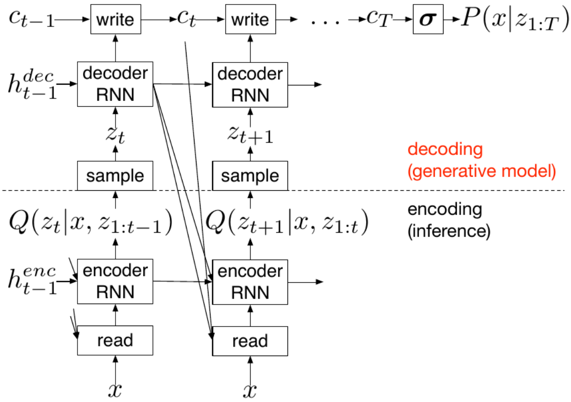
- $Q(Z_t|h_t^{enc}) = \mathcal{N}(Z_t|\mu_t, \sigma_t)$
- $\mu_t = W(h_t^{enc})$
- $\sigma_t = exp(W(h_t^{enc}))$



The Draw Network - Architecture

Decoder Equations:

- Input latent code z_t
- $z_t \sim Q(Z_t | h_t^{enc})$
- $h_t^{dec} = RNN^{dec}(h_{t-1}^{dec}, z_t)$
- $c_t = c_{t-1} + write(h_t^{dec})$
- c_t is canvas matrix



The DRAW Network - Loss Function

- The final canvas matrix c_T is used to parameterise a model $D(X|c_T)$ of the input data
- D is a Bernoulli distribution with means given by $\sigma(c_T)$
- Two types of losses are added:
  - *Reconstruction* Loss:

$$\mathcal{L}^x = -logD(x|c_T)$$

  - *Latent* Loss:

$$\mathcal{L}^z = \Sigma_{t=1}^{T} KL(Q(Z_t|h_t^{enc})\|P(Z_t))$$

$$= \frac{1}{2}(\Sigma_{t=1}^{T}\mu_t^2 + \sigma_t^2 - log\sigma_t^2) - \frac{T}{2}$$

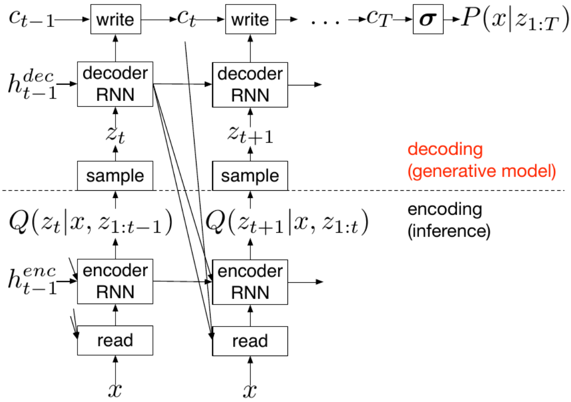
- Total Loss $\mathcal{L}$:

$$\mathcal{L} = \langle L^x + L^z \rangle_{z\sim Q}$$

The DRAW Network - Stochastic Data Generation

Using Decoder Network alone:

- $\tilde{z}_t \sim P(Z_t)$
- $\tilde{h}_t^{dec} = RNN^{dec}(\tilde{h}_{t-1}^{dec}, \tilde{z}_t)$
- $c_t = c_{t-1} + write(\tilde{h}_t^{dec})$
- $\tilde{x} \sim D(X|\tilde{c}_t)$



Outline

Read and Write Operations - Without Attention

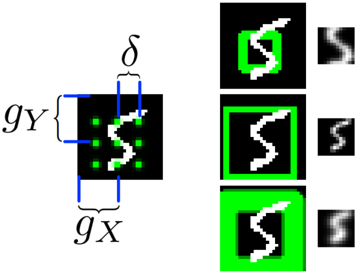
DRAW Without Attention

- The entire input image is passed to the encoder at every time-step
$$read(x, \hat{x}_t, h_{t-1}^{dec}) = [x, \hat{x}_t]$$
- The decoder modifies the entire canvas matrix at every time-step
$$write(h_t^{dec}) = W(h_t^{dec})$$

it does not provide the network with an **explicit selective attention mechanism**, which is believed to be crucial to large scale image generation

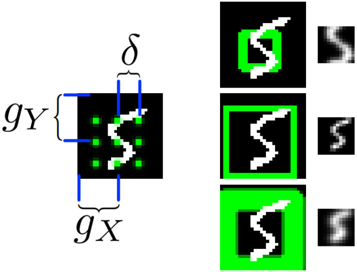
Read and Write Operations - Selective Attention Model

- 2D Gaussian Filter of $N \times N$ grid
- Grid Center (g_X, g_Y) and stride δ determines mean location μ_X^i, μ_Y^j of the filter at row i, column j in the patch as follows:
  $\mu_X^i = g_X + (i - N/2 - 0.5)\delta$
  $\mu_Y^j = g_Y + (j - N/2 - 0.5)\delta$
- $(\tilde{g}_X, \tilde{g}_Y, log\sigma^2, log\tilde{\delta}, log\gamma) = W(h^{dec})$
- For input image of size $A \times B$:
  $g_X = \frac{A+1}{2}(\tilde{g}_X + 1)$
  $g_Y = \frac{B+1}{2}(\tilde{g}_Y + 1)$
  $\delta = \frac{max(A,B)-1}{N-1}\tilde{\delta}$



Read and Write Operations - Selective Attention Model

- the horizontal and vertical filterbank matrices F_X and F_Y (dimensions $N \times A$ and $N \times B$ respectively) are defined as follows:
  $F_X[i,a] = \frac{1}{Z_X} exp(-\frac{(a-\mu_X^i)^2}{2\sigma^2})$
  $F_Y[j,b] = \frac{1}{Z_Y} exp(-\frac{(b-\mu_Y^j)^2}{2\sigma^2})$
- $read(x, \hat{x}_t, h_{t-1}^{dec}) = \gamma[F_Y x F_X^T, F_Y \hat{x} F_X^T]$
- $w_t = W(h^{dec})$
  $write(h_t^{dec}) = \frac{1}{\hat{\gamma}} \hat{F}_Y^T w_t \hat{F}_X$

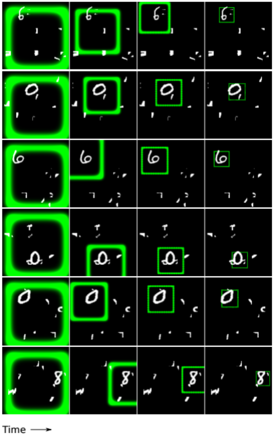


Outline

Experimental Results - Cluttered MNIST Classification

Table 1. Classification test error on 100 × 100 Cluttered Translated MNIST.

Model	Error
Convolutional, 2 layers	14.35%
RAM, 4 glimpses, 12 × 12, 4 scales	9.41%
RAM, 8 glimpses, 12 × 12, 4 scales	8.11%
Differentiable RAM, 4 glimpses, 12 × 12	4.18%
Differentiable RAM, 8 glimpses, 12 × 12	**3.36%**



Cluttered MNIST classification with attention. Each sequence shows a succession of four glimpses taken by the network while classifying cluttered translated MNIST. The green rectangle indicates the size and location of the attention patch, while the line width represents the variance of the filters.

Outline

Experimental Results - MNIST Generation

Table 2. Negative log-likelihood (in nats) per test-set example on the binarised MNIST data set. The right hand column, where present, gives an upper bound (Eq. 12) on the negative log-likelihood. The previous results are from [1] (Salakhutdinov & Hinton, 2009), [2] (Murray & Salakhutdinov, 2009), [3] (Uria et al., 2014), [4] (Raiko et al., 2014), [5] (Rezende et al., 2014), [6] (Salimans et al., 2014), [7] (Gregor et al., 2014).

Model	$-\log p$	$\leq$
DBM 2hl [1]	$\approx$ 84.62	
DBN 2hl [2]	$\approx$ 84.55	
NADE [3]	88.33	
EoNADE 2hl (128 orderings) [3]	85.10	
EoNADE-5 2hl (128 orderings) [4]	84.68	
DLGM [5]	$\approx$ 86.60	
DLGM 8 leapfrog steps [6]	$\approx$ 85.51	88.30
DARN 1hl [7]	$\approx$ 84.13	88.30
DARN 12hl [7]	-	87.72
DRAW without attention	-	87.40
DRAW	-	**80.97**

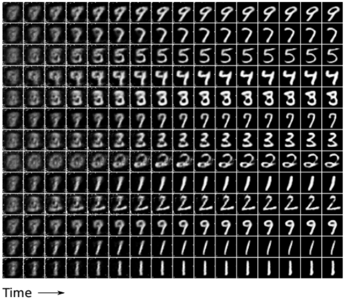


Figure 7. **MNIST generation sequences for DRAW without attention.** Notice how the network first generates a very blurry image that is subsequently refined.

Video - `https://www.youtube.com/watch?v=Zt-7MI9eKEo`

Outline

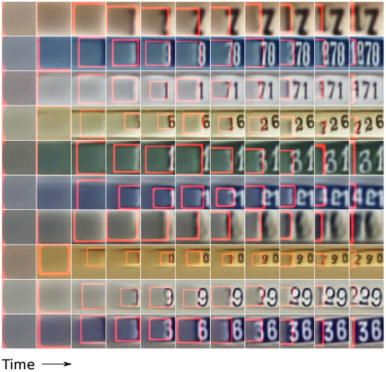
Experimental Results - MNIST Generation with Two Digits

DRAW Network is trained to generate images with two 28×28 MNIST images chosen at random and placed at random locations in a 60×60 black background

Video - `https://www.youtube.com/watch?v=Zt-7MI9eKEo`

Outline

Experimental Results - Street View House Number Generation

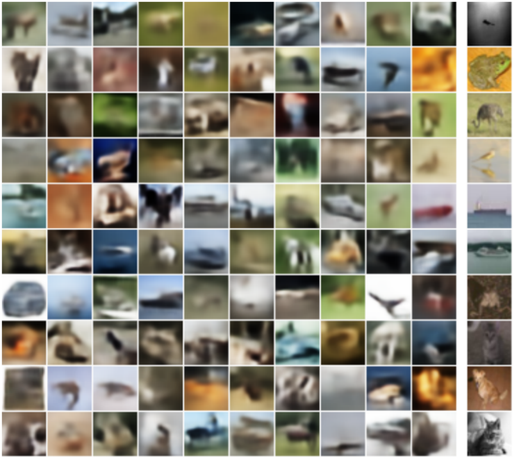


SVHN Generation Sequences. The red rectangle indicates the attention patch. Notice how the network draws the digits one at a time, and how it moves and scales the writing patch to produce numbers with different slopes and sizes.

Video - `https://www.youtube.com/watch?v=Zt-7MI9eKEo`

Outline

Experimental Results - Generating CIFAR Images



Generated CIFAR images. The rightmost column shows the nearest training examples to the column beside it.
Able to capture much of the shape, colour and composition of real photographs.

Conclusions

- Introduced the *Deep Recurrent Attentive Writer* (DRAW) neural network architecture
- demonstrated its ability to **generate highly realistic natural images** such as
  - photographs of house numbers
  - as well as improving on the best known results for binarized MNIST generation
- Introduced two-dimensional differentiable attention mechanism embedded in DRAW which is beneficial not only to **image generation**, but also to **image classification**