

Recurrent Models of Visual Attention

Volodymyr Mnih, Nicolas Heess, Alex Graves, Koray Kavukcuoglu
Google DeepMind

NIPS 2014

Presented by Sanghyeon Lee

Motivation

Problem

1. CNN has very huge amount of computation
2. CNN has fixed input image size

Motivation

1. When person see the object, they dose not tend to process a whole scene in its entirety at once
2. Instead of 1, human focus on attention selectively on parts of the visual space to acquire information
3. By using RL's method, they considers attention-based processing of a visual scene as a control problem

** Human eye's movements has been studied in neuroscience and cognitive science literature*

Method

The Recurrent **A**ttention Model (RAM)

Notation

x_t : Image at time t (Observation of the environment by Agent)

$\rho(x_t, l_{t-1})$: Restriction of image area

$l_t \sim p(\cdot | f_l(h_t; \theta_l))$: Location

s_t : State

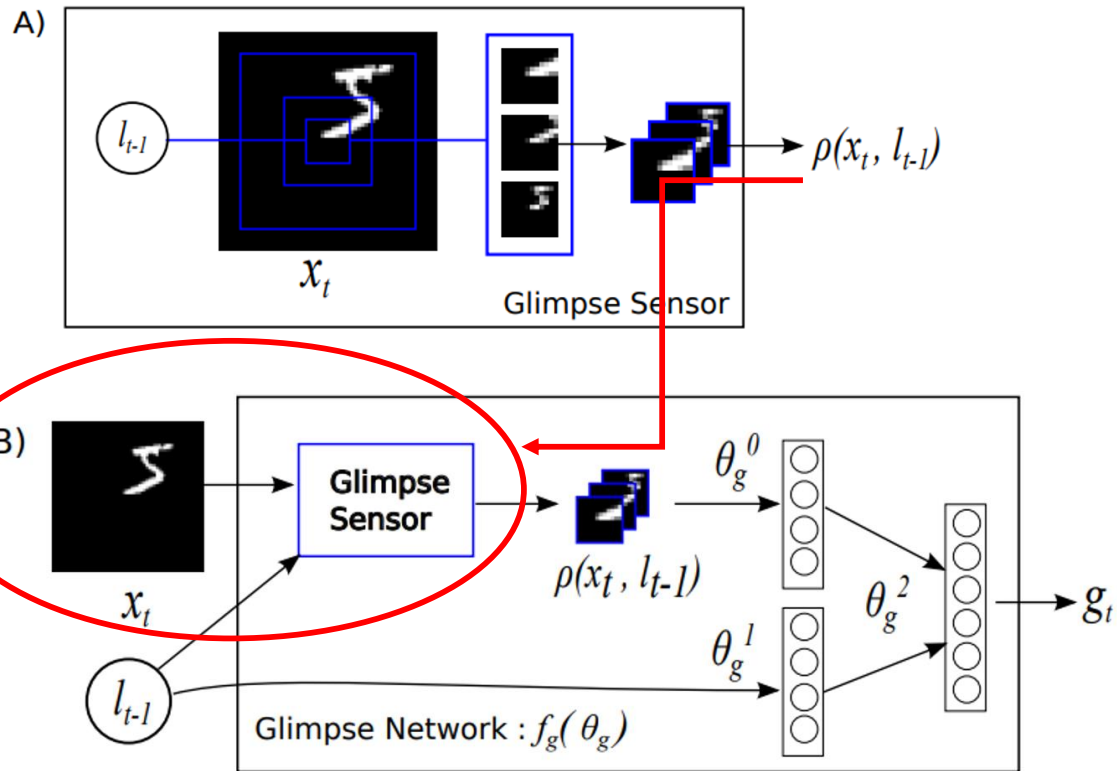
$a_t \sim p(\cdot | f_a(h_t; \theta_a))$: Action, (Using a softmax output)

r_t : Reward (If estimator == label : 1 else: 0)

$h_t = f_h(h_{t-1}, g_t; \theta_h)$: Hidden unit

Method

The Recurrent Attention Model (RAM)



A) Glimpse Sensor

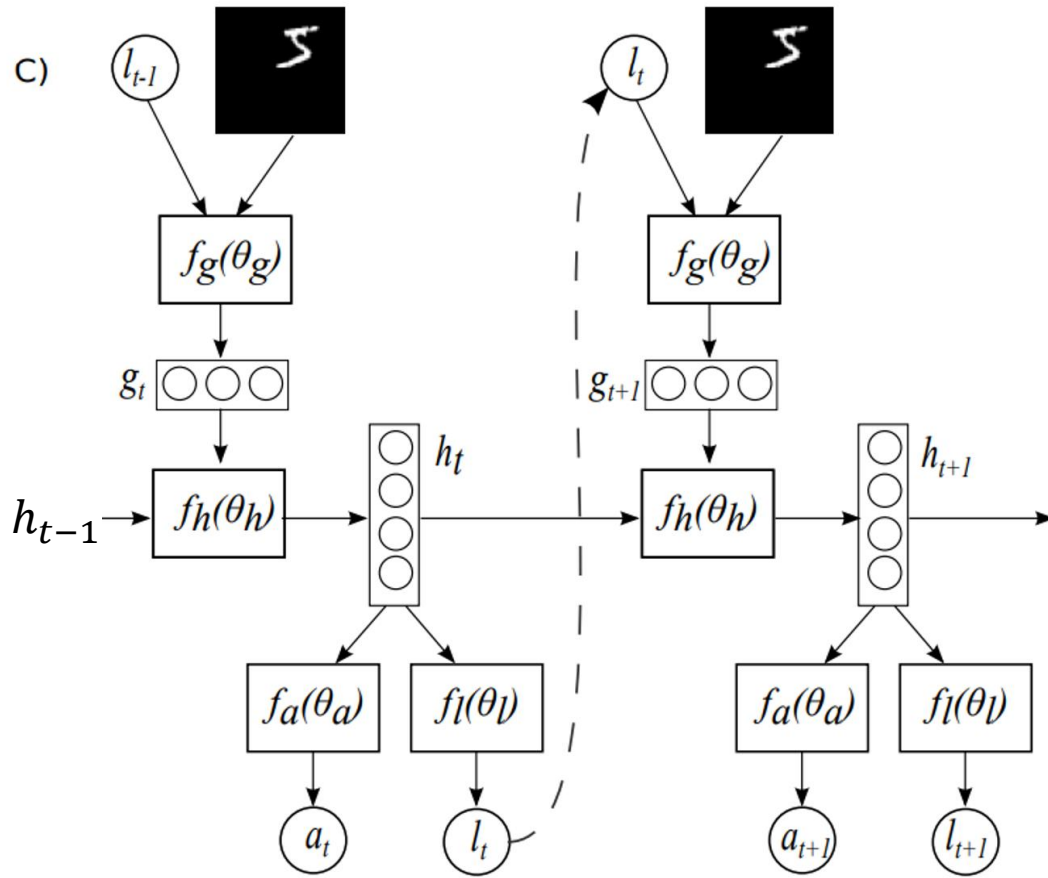
- Agent receives a observation of the env
- Agent does not have full information of image(Restricted by ρ)
- Getting a wider range of multiple image patch (center: l_{t-1} , $d1$, $d2$, $d3...$)

B) Glimpse Network

- Concatenate the output of A) and l_{t-1}
- The glimpse network f_g defines a trainable bandwidth limit (Has an effect of attention)
- Each layer has linear inner product layer (e.g $h_g = \text{RELU}[\text{Linear}(\rho(x_t, l_{t-1}))]$, $g_t = \text{RELU}(\text{Linear}(h_g) + \text{Linear}(h_l))$)

Method

The Recurrent Attention Model (RAM)



C) Model Architecture

- a_t and l_t is output of each RNN cell's
- a_t : Prob of classification label (e.g $P(x)$, $X=1,2,\dots,9$)
- Agent's object is maximize the total rewards

Method

Training Method – Policy Gradient

$$J(\theta) = \mathbb{E}_{p(s_{1:T};\theta)} \left[\sum_{t=1}^T r_t \right] = \mathbb{E}_{p(s_{1:T};\theta)} [R] \quad - \quad p(s_{1:T};\theta): \text{Policy s.t stationary distribution}$$

$$\nabla J(\theta) = \sum_{t=1}^T \mathbb{E}_{p(s_{1:T};\theta)} [\nabla_{\theta} \log \phi(u_t \mid s_{1:t}; \theta) R] \simeq \frac{1}{M} \sum_{i=1}^M \sum_{t=1}^T \nabla_{\theta} \log \phi(u_t^i \mid s_{1:t}^i; \theta) R^i$$

$$\frac{1}{M} \sum_{i=1}^M \sum_{t=1}^T \nabla_{\theta} \log \phi(u_t^i \mid s_{1:t}^i; \theta) (R_t^i - b_t), \text{ where } R_t^i = \sum_{t'_{prime}=1}^T r_{t'}^i \quad -\text{Reduce Variance}$$

a_t, l_t are trained by policy gradient

Baseline: MSE Loss

Action: NLE Loss


RL: Policy Gradient Loss

Method

Training Method – Policy Gradient

Unknown because the environment is typically unknown

$$\nabla \mathbb{E}[r(s, a)] = \nabla_{\theta} \sum_s d(s) \sum_a \pi_{\theta}(a|s) r(s, a)$$



$$= \sum_s d(s) \sum_a \nabla \pi_{\theta}(a|s) r(s, a)$$

$$= \sum_s d(s) \sum_a \pi_{\theta}(a|s) \frac{\nabla \pi_{\theta}(a|s)}{\pi_{\theta}(a|s)} r(s, a)$$

$$= \left(\sum_s d(s) \sum_a \pi_{\theta}(a|s) \right) \nabla \log \pi_{\theta}(a|s) r(s, a)$$

score function

$$= \mathbb{E}[\nabla_{\theta} \log \pi_{\theta}(a|s) r(s, a)]$$

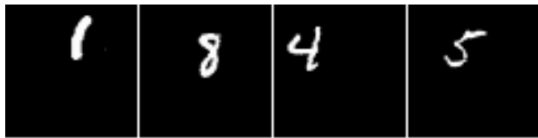


Calculating by MC Approximation

Experiments

(a) 28x28 MNIST		(b) 60x60 Translated MNIST	
Model	Error	Model	Error
FC, 2 layers (256 hidden each)	1.69%	FC, 2 layers (64 hidden each)	6.42%
Convolutional, 2 layers	1.21%	FC, 2 layers (256 hidden each)	2.63%
RAM, 2 glimpses, 8×8 , 1 scale	3.79%	Convolutional, 2 layers	1.62%
RAM, 3 glimpses, 8×8 , 1 scale	1.51%	RAM, 4 glimpses, 12×12 , 3 scales	1.54%
RAM, 4 glimpses, 8×8 , 1 scale	1.54%	RAM, 6 glimpses, 12×12 , 3 scales	1.22%
RAM, 5 glimpses, 8×8 , 1 scale	1.34%	RAM, 8 glimpses, 12×12 , 3 scales	1.2%
RAM, 6 glimpses, 8×8 , 1 scale	1.12%		
RAM, 7 glimpses, 8×8 , 1 scale	1.07%		

Table 1: Classification results on the MNIST and Translated MNIST datasets. FC denotes a fully-connected network with two layers of rectifier units. The convolutional network had one layer of $8 \times 10 \times 10$ filters with stride 5, followed by a fully connected layer with 256 units with rectifiers after each layer. Instances of the attention model are labeled with the number of glimpses, the number of scales in the retina, and the size of the retina.



(a) Translated MNIST inputs.



(b) Cluttered Translated MNIST inputs.

Figure 2: Examples of test cases for the Translated and Cluttered Translated MNIST tasks.

Experiments

(a) 60x60 Cluttered Translated MNIST		(b) 100x100 Cluttered Translated MNIST	
Model	Error	Model	Error
FC, 2 layers (64 hiddens each)	28.58%	Convolutional, 2 layers	14.35%
FC, 2 layers (256 hiddens each)	11.96%	RAM, 4 glimpses, 12×12 , 4 scales	9.41%
Convolutional, 2 layers	8.09%	RAM, 6 glimpses, 12×12 , 4 scales	8.31%
RAM, 4 glimpses, 12×12 , 3 scales	4.96%	RAM, 8 glimpses, 12×12 , 4 scales	8.11%
RAM, 6 glimpses, 12×12 , 3 scales	4.08%	RAM, 8 random glimpses	28.4%
RAM, 8 glimpses, 12×12 , 3 scales	4.04%		
RAM, 8 random glimpses	14.4%		

Table 2: Classification on the Cluttered Translated MNIST dataset. FC denotes a fully-connected network with two layers of rectifier units. The convolutional network had one layer of $8 \times 10 \times 10$ filters with stride 5, followed by a fully connected layer with 256 units in the 60×60 case and 86 units in the 100×100 case with rectifiers after each layer. Instances of the attention model are labeled with the number of glimpses, the size of the retina, and the number of scales in the retina. All models except for the big fully connected network had roughly the same number of parameters.

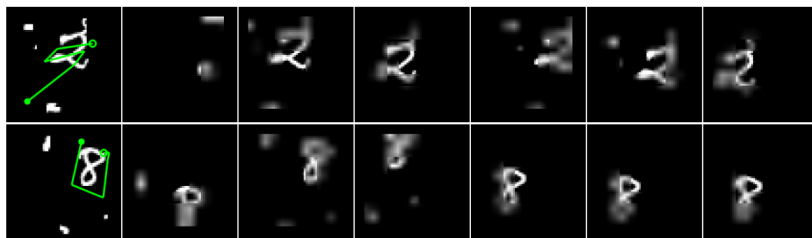


Figure 3: Examples of the learned policy on 60×60 cluttered-translated MNIST task. Column 1: The input image with glimpse path overlaid in green. Columns 2-7: The six glimpses the network chooses. The center of each image shows the full resolution glimpse, the outer low resolution areas are obtained by upscaling the low resolution glimpses back to full image size. The glimpse paths clearly show that the learned policy avoids computation in empty or noisy parts of the input space and directly explores the area around the object of interest.