Notes on Recurrent Models of Visual Attention (2015)

> *We present a novel recurrent neural network model that is capable of extracting information from an image or video by adaptively selecting a sequence of regions or locations and only processing the selected regions at high resolution.*

Glossary: -

A RNN that can be trained using reinforcement learning methods to learn task-specific policies.

Main aim is to reduce cost of computation when finding (images) or tracking (video) a object. Computational complexity is linear in the number of pixels. Compare this to the human visual system that "extracts" areas of "importance" and gives them more attention.[1] This reduces complexity of the task (where do I look?) and ignores noise (whats relevant for me?). Both number of parameters and amount of computation can be controlled independently of the size of input.

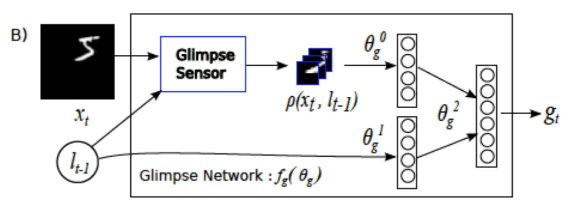
Litterature from neuroscience relies on saliency[2] and general attention.[3] **Should be expanded.**

Authors frame it as a control problem by implementing a RNN that processes input (frames) sequentially by attending to different parts of the frame and incrementally combines this information to build a dynamic interal representation. The next location is chosen at time t based on past information $X_{\leftarrow t}$ and the policy. The procedure uses backpropagation to train the neural network components and policy gradient.

Previous work has focussed on sliding window paradigmes for cascades,[4,5] branch and bound approach[6] or proposition of candidate windows based on their likelihood to contain objects[7,8] or salience as mentioned above. These methods prioritise "interesting" regions of the image but fail to integrate information across fixations. Others have used a sequential decision tasks.[9–14]

Essentially this is a POMDP. At each t the agent gets input $x_t = \{x_{\leftarrow t}, x_t\}$ through a bandwidth limited sensor $\mathcal{M}$ (**How is the sensor implemented?**) that senses the (hidden) environmental state s_t (full image or current frame and past information). If $x_t = s_t$ the agent has "full information". The agent decides where to place the sensor (attention) and can change the true (hidden) state of the environment by taking action a_t. Since $\mathcal{M} : x_t \rightarrow s_t$ is only partial the agent needs to build a internal representation $p(s_t, x_t | x_{\leftarrow t})$ using past information effectively. At each t the agent gets a scalar reward r_t that depends on the actions "and can be delayed" (**What is meant by delay?**). The agent chooses the policy π_i that maximises $\sum r_t$.

$\mathcal{M}(x_t, l_{t-1})$ where l_{t-1} is last location of sensor. Region around l is high-res but progressively gets lower res further from l. This is low the dimensionality of s_t is reduced to *glimpse* g_t. This is used in f_g (**What is this subscript?**) that produces a vector $g_t = f_g(x_t, l_{t-1}; \theta_g)$ (**What does ; mean here?**) where $\theta_g = \{\theta_g^1, \theta_g^2, \theta_g^3\}$ (**Is this raised to power 1 2 3?**).



The internal (hidden state) $h_t = f_h(h_{t-1}, g_t; \theta_h)$ or *core network* summarises past information in h_{t-1} and must therefore be Markovian. The external input is g_t. At each t the agent performs two actions: Deploy sensor a_s and a environmental specific action a_t that depends on the task. Locations are chosen from a distribution parameterised by the location network $f_l(h_t; \theta_l)$ where $l_t \sim p(\cdot | f_l(h_t; \theta_l))$

and environment action $a_t \sim p(\cdot|f_a(h_t;\theta_a))$. For classification they use softmax and the exact formulations of the dynamic environment depends (as said) on the task (motor control, joystick, etc.). The model can be augmented to a cost-sensitive classisifier by adding negative reward for each additional glimpse. As said, the agent maxmises $\sum r_t$ or alternatively $\sum \gamma r_t$ where gamma is a discount factor. E.g. $r_T = 1$ if the object is classified correctly after T time $r_T = 0$ other ways.

> Thus the agent needs to learn a stochastic policy $\pi((l_t, a_t)|s_{\leftarrow t};\theta)$ where θ maps the history of past interactions with the environment $s_{\leftarrow t} = x_1, a_1, l_1, \ldots, x_{t-1}, l_{t-1}, a_{t-1}, x_t$ to a distribution over actions for the current time step, subject to the constraint of the sensor. (**What the f*** does that mean?**)

Parameters of interest are given by the glimpse network, core network (figure below) and action network $\theta = \{\theta_g, \theta_h, \theta_a\}$ that are learned by maximising total (**Expected?**) reward. The policies of the agent in combination with environment gives a distribution over possible interaction sequences $\phi_{1:N}$. Maximise under

$$J(\theta) = \mathbb{E}_{p(\phi_{1:T};\theta)}\left[\sum_{t=1}^{T} r_t\right] = \mathbb{E}_{p(\phi_{1:T};\theta)}\left[R\right] \quad (1)$$

where $p(\phi_{1:T};\theta)$ depends on policy.

Maximising $J(\cdot)$ is nontrivial as the expectation is over all possible high-dimensional interactions between policy and (unknown) environment. Applying gradient decent on a RL algorithm gives

$$\begin{aligned}\nabla_\theta J &= \sum_{t=1}^{T} \mathbb{E}_{p(\phi_{1:T};\theta)}\left[\nabla_\theta log\pi(u_t|\phi_{1:T};\theta)\cdot R\right] \\ &\approx \tfrac{1}{M}\sum\sum_{i=1,t=1}^{M,T} \nabla_\theta log\pi(u_t^i|\phi_{1:T}^i;\theta)\cdot R^i\end{aligned} \quad (2)$$

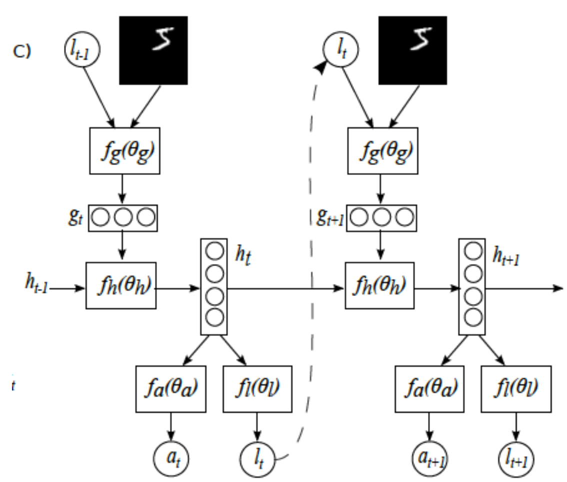
where each ϕ^i denotes a intertaction sequence obtained by running the current agent for $i = 1\ldots M$ episodes of T length. u_t denotes action at time t. For each run θ is adjusted to such that the log probability of actions that lead to high cumulative reward is increased, while actions that produce low cumulative reward is reduced. $\nabla_\theta log\pi(u_t^i|\phi_{1:T}^i;\theta)$ is solved by backpropagation.

Equation 2 is an unbiased estimtae of the gradient but may have high variance. It is common to solve this by considering

$$\frac{1}{M}\sum\sum_{i=1,t=1}^{M,T} \nabla_\theta log\pi(u_t^i|\phi_{1:T}^i;\theta)\cdot(R^i - b_t) \quad (3)$$

where R^i is total reward following action u_t^i and b_t is a baseline that depends on $\phi_{1:T}^i$ (via. h_t^i) but *not* on the action itself u_t^i (**Kind of cryptic. How is this baseline calculated?**). Taking expectations to equation 3 is equal to equation 2 but may have lower variance (**Why?**). Natural to select $b_t = \mathbb{E}(R_t)$ **this baseline is simply a value function.** Equation 3 results in an algorithm that ascribes higher log-probability to actions that were followed by larger-than-expected rewards (essentially prediction error) and vice versa for smaller rewards. The baseline is learned by reducing squared error between R_t^i and b_t.

If we have labeled data we can train on $log\pi(a_t^*|\phi_{1:T}^i;\theta)$ where a_t^* denotes labeled ground truth.



References

1 dd Ronald A. Rensink. The dynamic representation of scenes. Visual Cognition, 7(1-3):17–42, 2000.

[2] L. Itti, C. Koch, and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence, 20(11):1254–1259, 1998.

[3] Mary Hayhoe and Dana Ballard. Eye movements in natural behavior. Trends in Cognitive Sciences, 9(4):188 – 194, 2005.

[4] Pedro F. Felzenszwalb, Ross B. Girshick, and David A. McAllester. Cascade object detection with de- formable part models. In CVPR, 2010.

[5] Paul A.Violaand Michael J.Jones. Rapid object detection using a boosted cascade of simple features. In CVPR, 2001.

[6] Christoph H. Lampert, Matthew B. Blaschko, and Thomas Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In CVPR, 2008.

[7] KE Avande Sande, J.R.R. Uijlings, T Gevers, and A.W.M. Smeulders. Segmentationas Selective Search for Object Recognition. In ICCV, 2011.

[8] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. What is an object? In CVPR, 2010.

[9] Bogdan Alexe, Nicolas Heess, Yee Whye Teh, and Vittorio Ferrari. Searching for objects driven by context. In NIPS, 2012.

[10] Nicholas J. Butko and Javier R. Movellan. Optimal scanning for faster object detection. In CVPR, 2009.

[11] Misha Denil, Loris Bazzani, Hugo Larochelle, and Nando de Freitas. Learning where to attend with deep architectures for image tracking. Neural Computation, 24(8):2151–2184, 2012.

[12] Hugo Larochelle and Geoffrey E. Hinton. Learning to combine foveal glimpses with a third-order boltz- mann machine. In NIPS, 2010.

[13] Lucas Paletta, Gerald Fritz, and Christin Seifert. Q-learning of sequential attention for visual object recognition from informative local descriptors. In CVPR, 2005.

[14] M. Ranzato. On Learning Where To Look. ArXiv e-prints, 2014.

[15] R.J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning, 8(3):229–256, 1992.