Incorporating Attention in World Models for Improved Dynamics Modeling

Parth Chadha *
NVIDIA

Deepika Bablani *
IBM Research

Abstract

Recurrent world models have recently been shown to perform well on reinforcement learning tasks, where spatio-temporal representations are learned in an unsupervised manner in simulated game environments. This work is an attempt to improve the world model by incorporating a content-based attention mechanism in the dynamics modeling module which is used to simulate a training environment. The proposed attention mechanism uses the dynamics from the recurrent neural network (RNN) with the encoded visual information to create a focused spatial representation of the input. Further, a gating network is used during training of the visual encoding module which helps in predicting descriptive factorized latent feature vectors, which tend to generate more realistic states when combined with attention in an unsupervised manner. The efficacy of the approach is demonstrated through experiments in VizDoom game environment.

1 Introduction

There has been growing interest in model based reinforcement learning (RL) with a focus on learning descriptive abstract representations of sensory input. Task specific spatio-temporal abstractions of inputs from the environment form a key component for agents trained using these techniques to act optimally. The ability of the agent to efficiently utilize these learned abstractions to train in a simulated environment with no external supervision [1], analogous to “dreaming" in humans, depends directly on the effectiveness of the predictive abstraction in capturing rich, disentangled representations of raw sensory data.

The focus of this work is on improving the richness and disentanglement of learned sensory abstractions used for reconstruction by using disentangled visual embeddings along with selective spatial attention and global context. Using a gating principle similar to [2] during the training of the visual encoding module on sequences of images encourages the latent representation to capture disentangled features of variation. Furthermore, focusing explicitly on spatial content relevant to the transition dynamics leads to a richer spatial embedding. This is achieved by utilizing the dynamics from the RNN with the encoded visual information as the context, to attend to relevant details of the visual embedding. This allows focusing only on those visual transformations which are a consequence of the chosen action. Augmenting global content of the visual input with fine-grained focus on specific parts allows for more a detailed encoding without compromising significantly on the big picture.

Combined, these local and global contexts lead to significantly improved world models, as demonstrated in the experiments in VizDoom [3] environment in Section 4.

*Both authors contributed equally.

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

2 Background

2.1 World Model

Several recent papers have focused on learning accurate models of the environment using model based reinforcement learning [1], [4], [5], [6] . Recurrent world models introduced in [1] use an internal world model to train an agent entirely on a simulated environment and transfer the learned policy effectively to the real world. The authors propose an approach and an architecture inspired by how humans perceive and act in the world, through learning of abstract spatio-temporal representations. They use a simple model composed of three distinct components which are trained independently, viz., a visual sensory component V, a memory component M and a decision making component C.

V is a Variational Auto-Encoder (VAE) [7] and is used to compress the visual information the agent receives at each time step from the environment. Given a compressed representation z from V, M is used to model the dynamics and predict the next z. A Mixed Density Network [8] combined with a recurrent neural network (MDN-RNN) is used as M, which models $P(z_{t+1}|a_t, z_t, h_t)$, where a_t is the action taken at time t and h_t is the hidden state of the RNN at time t. Instead of predicting z directly, M models the probability density function that generates z to capture the stochastic variation of the environment. C is a RL controller that uses the h_t from the RNN to determine the best action to be chosen to maximize cumulative reward in an episode. One of the key contributions of this work is that M allows the controller to be trained on simulated environment roll-outs using the internal environment model, and hence to be trained in a completely unsupervised manner. The policies learned in the agent's mental model are then transferred to the real environment. The authors use Co-variance Matrix Adaptation Evolution Strategy (CMA-ES) [9], [10] to train C.

2.2 Learning Disentangled Representations

Generative models for tasks such as modelling visual representations or learning the dynamics of an environment have to deal with the problem of entangled representations, making it challenging to interpret these representations and also making it harder for the dynamic model to re-use representation components in future time steps. In recent work on continuation learning [2], the authors propose an architecture to produce factorized representations of a given visual image. A gating layer is used which receives input from two consecutive time steps, and selects a small set of gating units to characterize the dynamics between x_{t-1} and x_t. To learn such a gating function, the authors use scheduled weight sharpening [11], which creates a smooth, differentiable loss function to select gates. In their work, x_{t-1} and x_t are concatenated and passed through a linear layer. w_i are outputs of the linear layer and transformed in every iteration(γ) according to equation 1.

$$w_i^{'} = \frac{(w_i + \mathcal{N}(0, \sigma^2))^{\gamma}}{\sum_j w_j^{\gamma}} \tag{1}$$

3 Proposed Architecture

The architecture proposed in this work is in Figures 1 and 2. Roll-outs are first generated from the environment using a random policy (random action repeated for n iterations, where n is uniformly sampled from [1,10]) to generate sequences of images to train the VAE. The VAE is trained using a gated network as shown in Figure 1, where two consecutive frames are fed to the visual encoder and the generated embeddings are passed to a gating network to create a fused embedding which is then used by the decoder to reconstruct the image. The gating mechanism causes visual embeddings from the encoder to be disentangled, where each dimension of the embedding controls an important part of the scene independently.

After training the VAE module, the memory network M, implemented using Long Short-Term Memory (LSTM)[12] is trained. In addition to the LSTM network, a content based attention mechanism is introduced which uses the hidden outputs from the LSTM and the activation of an intermediate convolutional layer from the VAE encoder for context. The convolutional feature map from the conv3 layer of an encoder with 4 convolutional layers is used along with the hidden state of the LSTM to obtain attention weights, which when up-sampled back to the original image size produce the spatial region that the model decides to focus on. Further, the attention weighted encoding

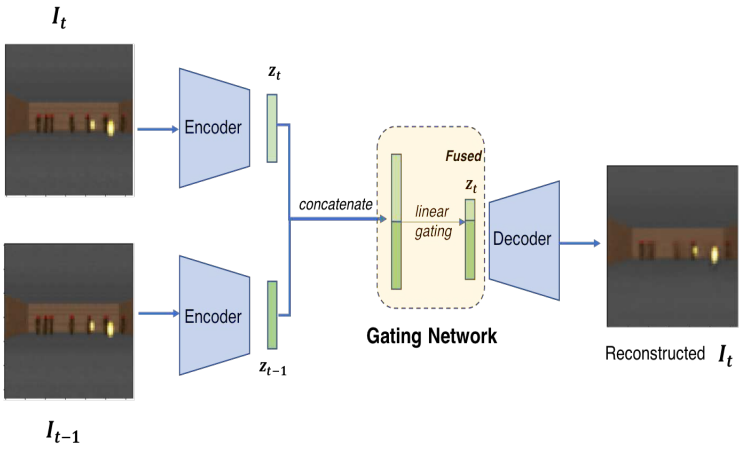

Figure 1: **Visual Gating Architecture**

is concatenated with the output from the LSTM and then passed through a linear layer to generate the parameters for a Mixture of Gaussians. The model learns to attend to the relevant portion of the input image while the MDN is trained to maximize the log-likelihood of the label z_t under the mixture of distributions generated by the network.

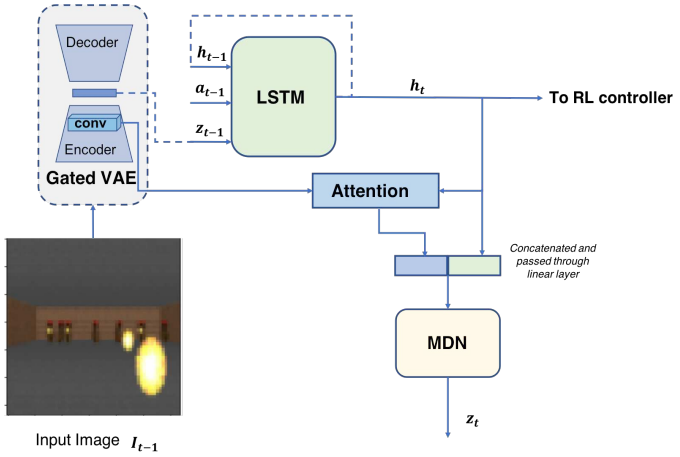

Figure 2: **MDN-RNN with Attention**: The MDN is trained to maximize the log-likelihood of z_t under the mixture of distributions generated by the network.

Based on the trained V and M modules, a controller C implemented as a simple linear layer is trained using CMA-ES algorithm to maximize the expected reward on the simulated environment generated by the M network.

4 Experiments

The proposed model was tested in the CarRacing environment from OpenAI Gym [13] and ViZDoom [3] (*DoomTakeCover-v0*). VizDoom was found to be relatively more challenging with scope for attending to different parts of images while generating episodes via simulation. For demonstration, results on the ViZDoom environment are presented in Figure 3. The input frame is shown in the left column and the middle column shows where the model chose to attend while generating the next frame. The right-most column shows the generated next frame. Crucially, these are examples from a simulated roll-out, demonstrating the detail which gets captured even during simulation.

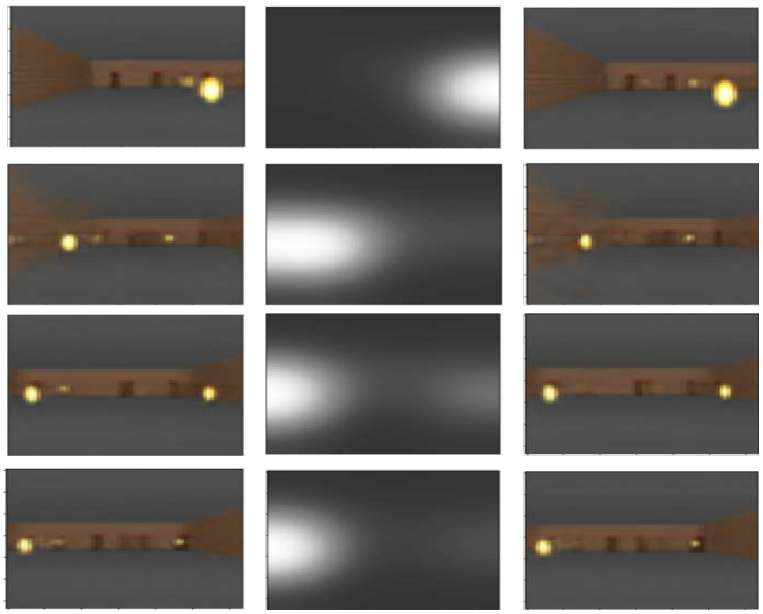

Figure 3: **Results on DoomEnv**: Left: Input frame, Middle: Attention map, Right: Simulated frame

References

[1] David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. *arXiv preprint arXiv:1809.01999*, 2018.

[2] William F Whitney, Michael Chang, Tejas Kulkarni, and Joshua B Tenenbaum. Understanding visual concepts with continuation learning. *arXiv preprint arXiv:1602.06822*, 2016.

[3] Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. Vizdoom: A doom-based ai research platform for visual reinforcement learning. In *Computational Intelligence and Games (CIG), 2016 IEEE Conference on*, pages 1–8. IEEE, 2016.

[4] Jürgen Schmidhuber. On learning to think: Algorithmic information theory for novel combinations of reinforcement learning controllers and recurrent neural world models. *arXiv preprint arXiv:1511.09249*, 2015.

[5] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in atari games. In *Advances in neural information processing systems*, pages 2863–2871, 2015.

[6] Alexey Dosovitskiy and Vladlen Koltun. Learning to act by predicting the future. *arXiv preprint arXiv:1611.01779*, 2016.

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

[8] Christopher M Bishop. Mixture density networks. Technical report, Citeseer, 1994.

[9] Nikolaus Hansen. The cma evolution strategy: A tutorial. *arXiv preprint arXiv:1604.00772*, 2016.

[10] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, 9(2):159–195, 2001.

[11] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

[12] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[13] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.