World Models

Ganga Meghanath
EE15B025
gangamegha29@gmail.com

Topics in Reinforcement Learning
Indian Institute of Technology Madras

September 27, 2018

Motivation

Motivation

- Humans develop a mental model of the world based on what they are able to perceive with their limited senses
- Brain learns abstract representation of both spatial and temporal aspects of the information. Eg : Baseball
- Artificial agent can benefit from having a good representation of past and present states, and a predictive model of the future

Perceive time spatially

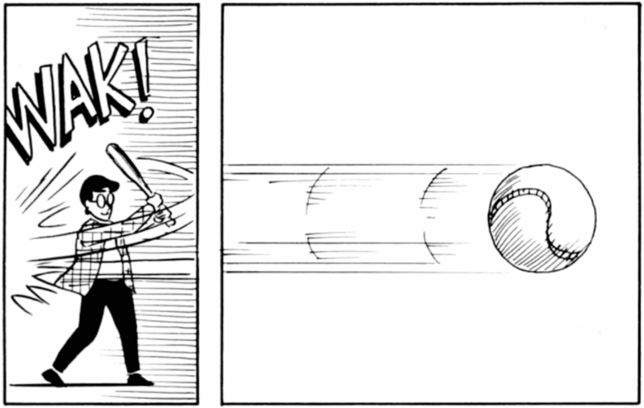


Figure: *"In the world of comics, time and space are one and the same"* - Scott McCloud, cartoonist and comics theorist

Model

Theory

- Generative Neural Network models for learning a compressed spatial and temporal representation of the environment
- Divide agent into a large world model and a small controller model
- Model is learned in latent space
- Optimize Controller : Covariance- Matrix Adaptation Evolution Strategy (CMA-ES)
- A small controller lets the training algorithm focus on the credit assignment problem on a small search space, while not sacrificing capacity and expressiveness via the larger world model

World Model

At each time step, our agent receives an **observation** from the environment.

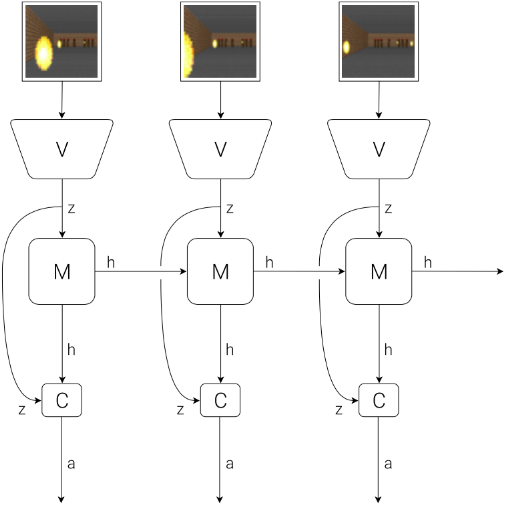
World Model

The **Vision Model (V)** encodes the high-dimensional observation into a low-dimensional latent vector.

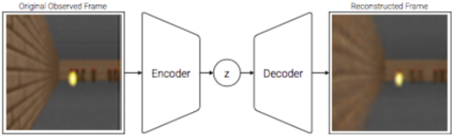
The **Memory RNN (M)** integrates the historical codes to create a representation that can predict future states.

A small **Controller (C)** uses the representations from both **V** and **M** to select good actions.

The agent performs **actions** that go back and affect the environment.



Visual Model : VAE



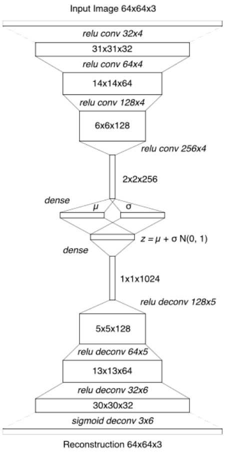


Figure: Convolutional Variational Autoencoder

Memory Model : MDN-RNN

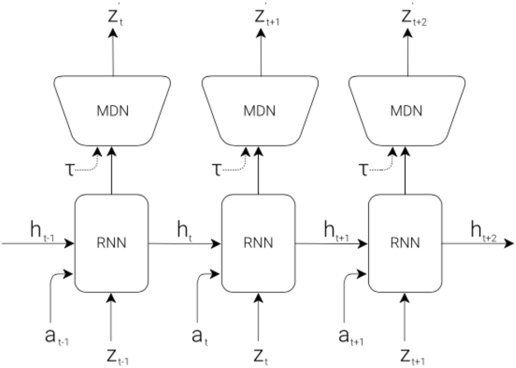
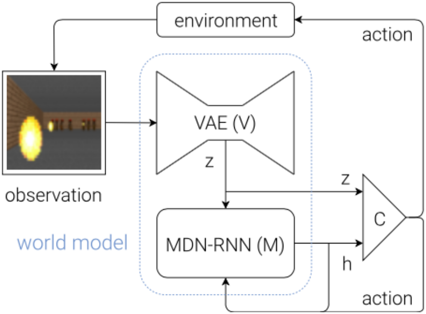


Figure: RNN with a Mixture Density Network layer : Outputs parameters of a mixture of Gaussian distribution - used to sample a prediction of the next latent vector z.

World Model



- V and M has no knowledge about the actual reward signals from the environment
- Only the Controller (C) Model has access to the reward information from the environment

Pseudo-code

Algorithm 1 Rollout

```
procedure def rollout(controller) :
    obs ← env.reset()
    h ← rnn.initial_state()
    done ← False
    cumulative_reward ← 0
    while not done do
        z ← vae.encode(obs)
        a ← controller.action([z, h])
        obs, reward, done ← env.step(a)
        cumulative_reward ← cumulative_reward + reward
        h ← rnn.forward([a, z, h])
    return cumulative_reward
```

Experiments

Observation captured by World Model

Play/Pause

Media Caption

Procedure

Algorithm 2 Car-Racing

1: Collect 10,000 rollouts from a random policy
2: Train VAE (V) to encode frames into $z \epsilon R^{32}$
3: Train MDN-RNN (M) to model $P(z_{t+1}|a_t, z_t, h_t)$
4: Define Controller (C) as $a_t = W_c[z_t h_t] + b_c$
5: Use CMA-ES: W_c & b_c : maximize expected cumulative reward

Model	Parameter Count
VAE	4,348,547
MDN-RNN	422,368
Controller	867

Controller

Table: Controller models

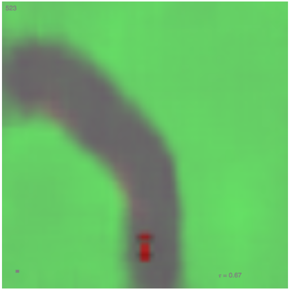
Model	**Method**	**Avg. Score**
I	V Model	632 ± 251
II	V model with Hidden layer	788 ± 141
III	Full World Model	906 ± 21

I: Wobbles around and misses the tracks on sharper corners

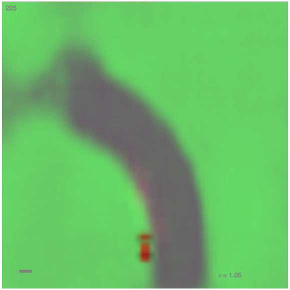
II: Performance improves; but not enough

III: Driving is more stable; agent is able to tackle sharp corners

Car Racing Dreams



(a) $\tau = 0.67$



(b) $\tau = 1.05$

- Hallucinated car racing environment generated by M
- Adjust τ to control the uncertainty of the environment generated by M

VizDoom

Play/Pause | Media Caption

VizDoom

- Learning Inside of a Dream
- Substitute the actual environment with the world model
- Making the game more challenging in the Dream environment
- Transfer Policy to Actual Environment

Temperature τ	Virtual Score	Actual Score
0.10	2086 ± 140	193 ± 58
0.50	2060 ± 277	196 ± 50
1.00	1145 ± 690	868 ± 511
1.15	918 ± 546	1092 ± 556
1.30	732 ± 269	753 ± 139

Procedure

Algorithm 3 VizDoom

1: Collect 10,000 rollouts from a random policy
2: Train VAE (V) to encode frames into $z \epsilon R^{64}$
3: Train MDN-RNN (M) to model $P(z_{t+1}, d_{t+1} | a_t, z_t, h_t)$
4: Define Controller (C) as $a_t = W_c[z_t h_t]$
5: Use CMA-ES : W_c : max(expected survival time) : Virtual envt
6: Use learned policy from (5) on actual environment

Model	Parameter Count
VAE	4,446,915
MDN-RNN	1,678,785
Controller	1,088

Observation captured by World Model

Play/Pause

Media Caption

Cheating the World Model

- Controller has access to all of the hidden states of M
- Adversarial policy : M model never shoots a fireball

Measures :

- MDN-RNN : dynamics model - models the distribution of possible outcomes
- τ : tradeoff between realism and exploitability.

Cheating the World Model

Play/Pause | Media Caption

Iterative Training Procedure

- Sophisticated Environments : Strategical Navigation - parts of the environment revealed at a time
- Curriculum Learning : allow the C–M model to develop a natural hierarchical way to learn

Algorithm 4 Iterative Training Procedure

1: Initialize M, C with random model parameters
2: Rollout to actual environment N times
3: Save all actions a_t and observations x_t during rollouts to storage
4: Train M to model $P(x_{t+1}, r_{t+1}, a_{t+1} d_{t+1} | x_t, a_t, h_t)$
5: Train C to optimize expected rewards inside of M
6: Go back to (2) if task has not been completed

Implementation details

VAE

- Trained the model for 1 epoch over the data collected from a random policy

MDN-RNN

- Probability distribution of z_{t+1} modeled as a Mixture of Gaussian distribution : 5 Gaussian mixtures
- Trained for 20 epochs on the data collected from a random policy agent
- LSTM hidden units :
  - Car Racing task : 256 hidden units
  - Doom task : 512 hidden units

Implementation details

Controller

- Action space was bound to appropriate ranges.
- Car Racing
  - Steering wheel : [-1,1]
  - Acceleration pedal : [0, 1]
  - Brakes : [0, 1]
  - Agent is rewarded for visiting as many tiles as possible in the least amount of time
- VizDoom
  - Left, Stay, Right : [-1,1]
  - Cumulative reward : number of time steps the agent manages to stay alive during a rollout

Points to Critique

- Not end-to-end trainable
- Dynamics of the environment are not considered in the VAE
- Temperature parameter τ is a tunable parameter which has significant effects on the model performance
- Random policy used to generate sample trajectories for V & M
- For Car Racing task, N_z is 32 while for the Doom task N_z is 64
- Does not require running the actual Doom game engine ; used VizDoom only for the purpose of collecting training data
- Controller access to all of the hidden states of M

Related Work

Hallucination with RNNs

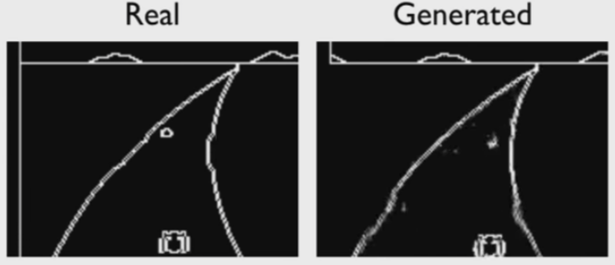


Figure: Probabilistic model of Enduro (Atari)

- Guest lecture by Alex Graves of Google Deepmind at the University of OXFORD
- Trained RNNs to learn the structure of the game and hallucinate similar game levels on its own

From Pixels to Torques

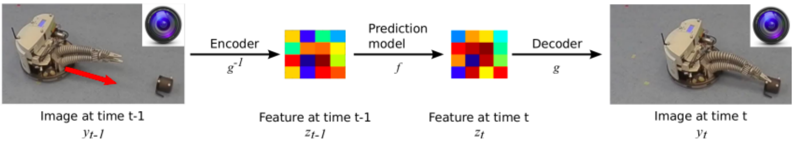


Figure: Feature learning and prediction model in feature space

- Agent must learn a closed-loop control policy from pixel information only
- Use deep autoencoders to learn a low-dimensional embedding of images jointly with a predictive model
- Joint learning ensures that not only static but also dynamic properties of the data are accounted for

PILCO

- PILCO : Probabilistic Inference for Learning COntrol
- Learning a probabilistic dynamics model and explicitly incorporating model uncertainty into long-term planning
- Reduce model bias; cope with very little data
- Gaussian process (GP) model is used to learn the system dynamics, and sample trajectories for training the controller

$$P(x_t|x_{t-1}, u_{t-1}) = \mathcal{N}(x_t|\mu_t, \Sigma_t)$$

References

- Marc Deisenroth and Carl E Rasmussen. "PILCO: A model-based and data-efficient approach to policy search". In: *Proceedings of the 28th International Conference on machine learning (ICML-11)*. 2011, pp. 465–472.
- David Ha and Douglas Eck. "A neural representation of sketch drawings". In: *arXiv preprint arXiv:1704.03477* (2017).
- David Ha and Jürgen Schmidhuber. "World models. arXiv preprint". In: *arXiv preprint arXiv:1803.10122* (2018).
- Niklas Wahlström, Thomas B Schön, and Marc Peter Deisenroth. "From pixels to torques: Policy learning with deep dynamical models". In: *arXiv preprint arXiv:1502.02251* (2015).

THANK YOU !!

Hope it was informative :)

CMA-ES

- Stochastic, derivative-free methods for numerical optimization of non-linear or non-convex continuous optimization problems
- Covariance matrix of the distribution is updated such that the likelihood of previously successful search steps is increased : facilitates a possibly much faster variance increase of favorable directions
- The step-size control aims to make consecutive movements of the distribution mean orthogonal in expectation and effectively prevents premature convergence yet allowing fast convergence to an optimum.