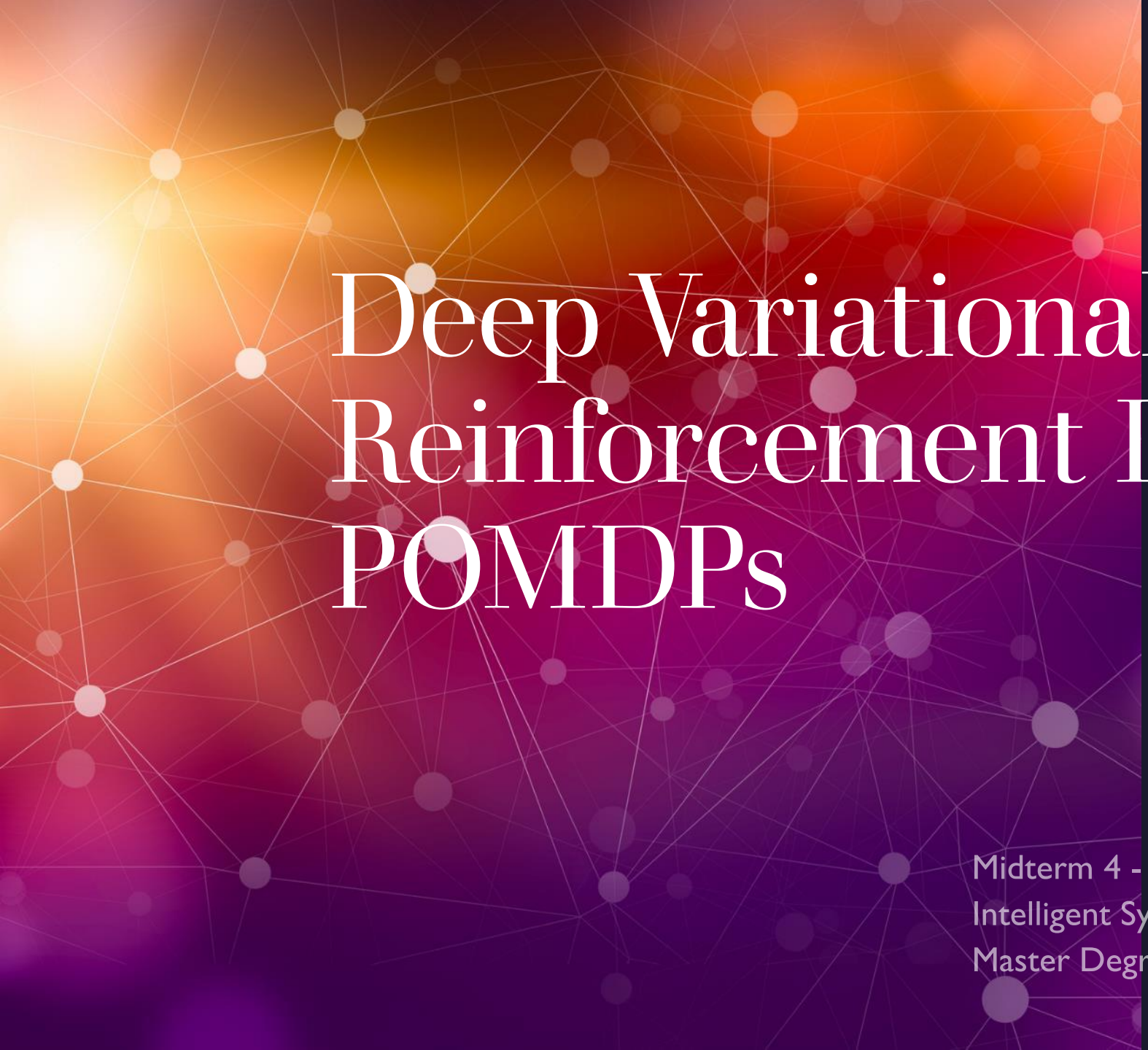


Deep Variational Reinforcement Learning for POMDPs

Midterm 4 - Gaetano Nicassio (658073)

Intelligent Systems for Pattern Recognition (760AA) A.Y. 22/23

Master Degree in Computer Science – Artificial Intelligence

Problem Introduction

Most **Deep Reinforcement Learning** (RL) methods assume that the state of the environment is **fully observable** at every time step. This assumption often does not hold in reality as occlusions and noisy sensor may limit the agent's perceptual abilities.

Problem formalization: Partially Observed Markov Decision Processes (**POMDPs**) that are notoriously hard to solve because :

- Information must be aggregated over time
- The entire story must be taken into account

Previous works like Deep Recurrent Q-network (**DRQN**) (Hauskenecht & Stone, 2015) and Action-specific Deep Recurrent Q-network (**ADRQN**) (Zhu et al., 2017) relies on RNN. These are completely model-free, place a heavy burdern on RNN, that summarize the history either by remembering features of the past or by computing simple heuristics instead of actual belief states.

Work's objectives

The aim of the authors's work is to allow an agent to learn models of latent state representation or transition and observation functions, and infer belief state using these learned models.

The paper propose the Deep Variational Reinforcement Learning (**DVRL**) model that provide a helpful inductive bias to the agent, can learn an internal generative model and use it to performe approximate inference to update the belief state. The authors develop a new approximation of the **ELBO**, based on autoencoding sequential Monte Carlo (**AESMC**) , allowing joint optimization with the **n-step** policy gradient update. It extend the RNN-based approach to explicity support belief inference.

Model Description

The model use different neural network architectures, a **Variational Auto Encoder** for time series, with a **new ELBO** approximation based on **Sequential Monte Carlo** to allow faster learning. To learn the parameter of the policy, it uses **n-step learning with A2C**, a synchronous semplification of Asynchronous Advantage Actor-Critic (A3C) with a modified implementation of BPTT in the case of policies with latent states. The n-step performs n_s consecutive steps in n_e parallels environment. The gradient update is based on mini-batch of size n_s x n_e

Update Rules

$$u_{t-1}^k \sim \text{Discrete}\left(\frac{w_{t-1}^k}{\sum_{j=1}^K w_{t-1}^j}\right)$$

$$z_t^k \sim q_\phi(z_t^k | h_{t-1}^{u_{t-1}^k}, a_{t-1}, o_t)$$

$$h_t^k = \psi_\theta^{\text{RNN}}(h_{t-1}^{u_{t-1}^k}, z_t^k, a_{t-1}, o_t)$$

$$w_t^k = \frac{p_\theta(z_t^k | h_{t-1}^{u_{t-1}^k}, a_{t-1}) p_\theta(o_t | h_{t-1}^{u_{t-1}^k}, z_t^k, a_{t-1})}{q_\phi(z_t^k | h_{t-1}^{u_{t-1}^k}, a_{t-1}, o_t)}$$

DVRL extend the latent state to be a set of K particles, with each particle consists of the triplet $(h_t^k\ z_t^k, w_t^k)$ with:

- $\boldsymbol{h_t^k}$**:** latent state of an RNN
- $\boldsymbol{z_t^k}$: an additional stochastic latent state to allows us to learn stochastic transition models
- $\boldsymbol{w_t^k}$: assigns each particle an important weight that measure how likely each new latent stae is under the model and how well it explains the current observation

- $p_\theta(z_t | h_{t-1}, a_{t-1})$: stochastic transition model
- $q_\emptyset(z_t |\ h_{t-1}, z_t, o_t)$: encoder
- $p_\theta(o_t | h_{t-1}, z_t, a_{t-1})$: decoder
- $h_t = \psi_\theta^{RNN}(h_{t-1}, z_t, a_{t-1}, o_t)$: deterministic transaction function denoted with Dirac delta distribution δ

Approximated Posterior Distribution

$$p_\theta(h_{\leq T}, z_{\leq T}, o_{\leq T} | a_{<T}) = p_\theta(h_0) \prod_{t=1}^{T} \Big(p_\theta(z_t | h_{t-1}, a_{t-1}) \\ p_\theta(o_t | h_{t-1}, z_t, a_{t-1}) \delta_{\psi_\theta^{\text{RNN}}(h_{t-1}, z_t, a_{t-1}, o_t)}(h_t) \Big), \quad (14)$$

Method

1. Resample particles based on their weight by drawing ancestor indices u_{t-1}^k based on the previous weight $w_{t-1}^{1:K}$

2. Pick the ancestor value $h_{t\_1}{}^{u_{t-1}^k}$ and use it to sample a new stochastic latent value z_t^k from the encoder $q_\emptyset$ conditioned on the ancestor value and the last actions a_{t-1} (with **reparametrization trick**).

3. Compute h_t^k from the first RNN and w_t^k as the following formula:.

4. Aggregate all the K values into the new belief state $\hat{b}_t$ and summarize into a vector representation $\hat{h}_t$ using a second RNN

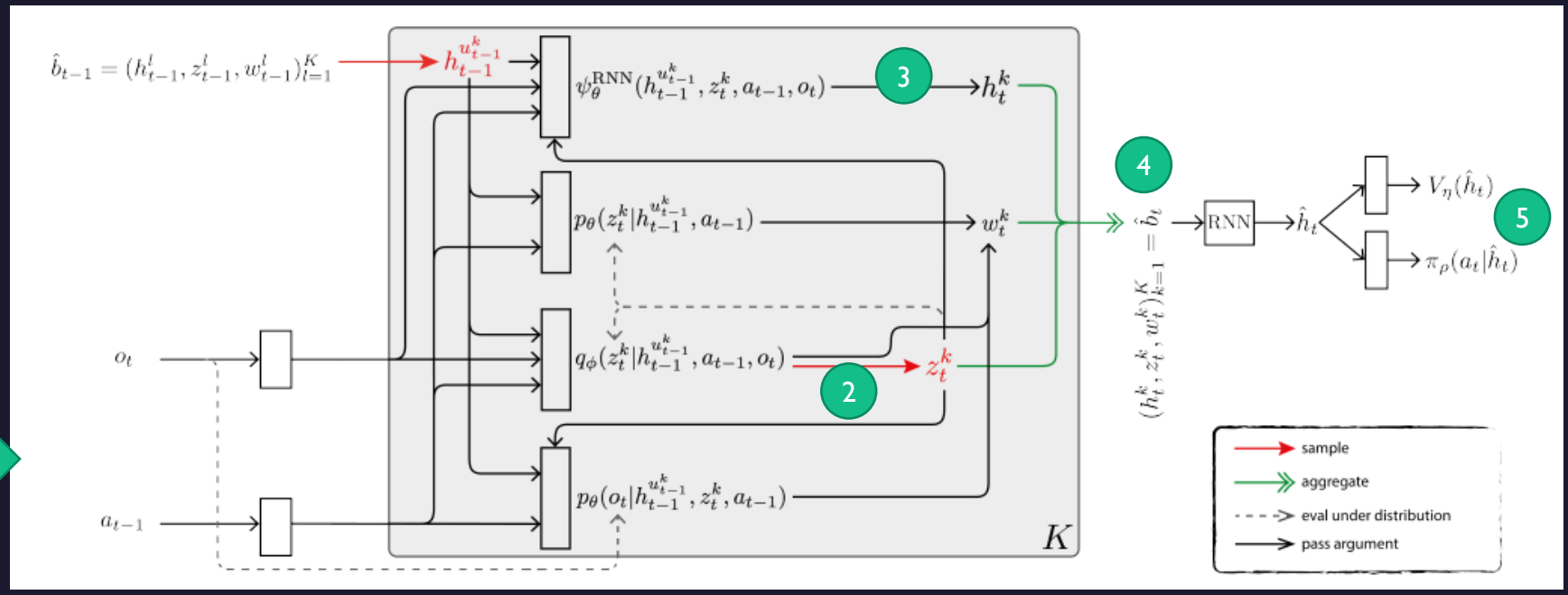
5. Actor-Critic can now condition on $\hat{h}_t$ and $\hat{b}_t$ is used as input for the next iteration timestep.

Steps 1,2,3 are iterated K time, with K number of particles

1

$$u_{t-1}^k \sim \text{Discrete}\left(\frac{w_{t-1}^k}{\sum_{j=1}^K w_{t-1}^j}\right)$$

Boxes indicates Neural Networks. Distributions are normal or Bernoulli distributions whose parameters are outputs of the neural network



Loss Function

To encourage learning a model, the authors include the term $L_t^{ELBO}(\theta,\phi) = -\frac{1}{n_e n_s}\sum_{envs}^{n_e}\sum_{i=0}^{n_s-1} log\left(\frac{1}{K}\sum_{k=1}^{K} w_{t+i}^k\right)$ (1) in each gradient update every ns steps. The general overall loss is then:

$$\mathcal{L}_t^{\text{DVRL}}(\rho,\eta,\theta,\phi) = \mathcal{L}_t^A(\rho,\theta,\phi) + \lambda^H \mathcal{L}_t^H(\rho,\theta,\phi) + \lambda^V \mathcal{L}_t^V(\eta,\theta,\phi) + \lambda^E \mathcal{L}_t^{\text{ELBO}}(\theta,\phi)\,.$$

The loss depend on the encoder parameter $\emptyset$ and model parameter θ. By introducing the n-step approximation L_t^{ELBO}, the model **can learn to jointly** optimize L_t^{ELBO} and the RL loss $L_t^A + \lambda^H L_t^H + \lambda^V L_t^V$.

If we assume that observations and actions are drawn from the stationary state distribution induced by the policy π_ρ, then $-L_t^{ELBO}$ is a stochastic approximation to the action-conditioned ELBO

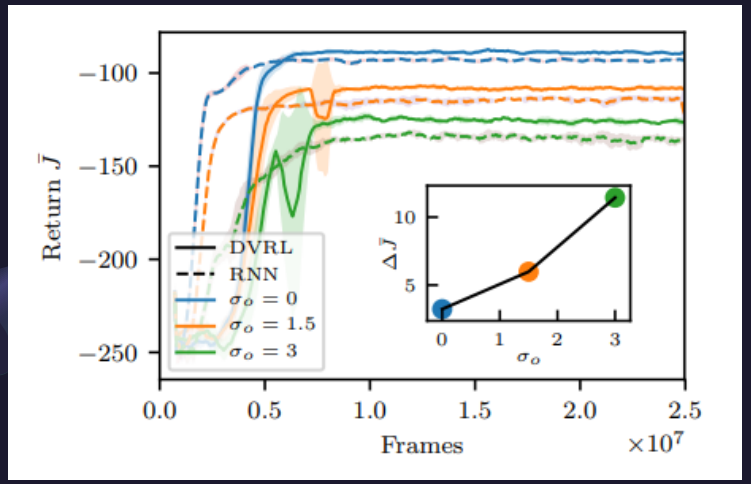
$$\frac{1}{T}\mathbb{E}_{p(\tau)}\,\text{ELBO}_{\text{SMC}}(o_{\leq T}|a_{<T}) = \frac{1}{T}\mathbb{E}_{p(\tau)}\mathbb{E}\left[\sum_{t=1}^{T}\log\left(\frac{1}{K}\sum_{k=1}^{K} w_t^k\right)\middle| a_{\leq T}\right] \quad (2)$$

To make equation (2) tractable, the authors approximate the expectation over $p(\tau)$ by using sampled trajectories from n_e environments. Furthermore, because we assume a stationary state distribution, we can take the sum over T outside of both expectations. This allows us to perform a stochastic gradient update that is based on only n_e summands instead of all T, leading to equation (1) which includes an additional minus sign to account for its minimization.

Key Result

DVRL has been evaluated on **Mountain Hike** and on **flickering Atari**. The paper shows that DVRL deals better with noisy or partially occluded observations and scales to high dimensional and continuous observation spaces (e.g. images, comples tasks). The experiments was also perform a series of **Ablation studies**, showing the importance of using many particles, including the ELBO training objective in the loss function and jointly optimising the ELBO and RL losses. The RNNs used are gated recurrent network (GRUs).

The main difficulty in Mountain Hike is to correctly estimate the current position. Consequently, the achieved return reflects the capability of the network to do so. **DVRL outperforms RNN based policies**, especially for higher levels of observation noise σ_o.



Flickering Atari was used as benchmark since it was previously used to evaluate the performance on ADRQN and DQRN. The authors, test the same subset of games of ADRQN and DQRN. The table below that **DVRL significantly outperforms the RNN-based policy** on five out of ten games and narrowly underperforms significantly on only one. This shows that **DVRL is viable for hig dimensional observation spaces with complex environmental models**.

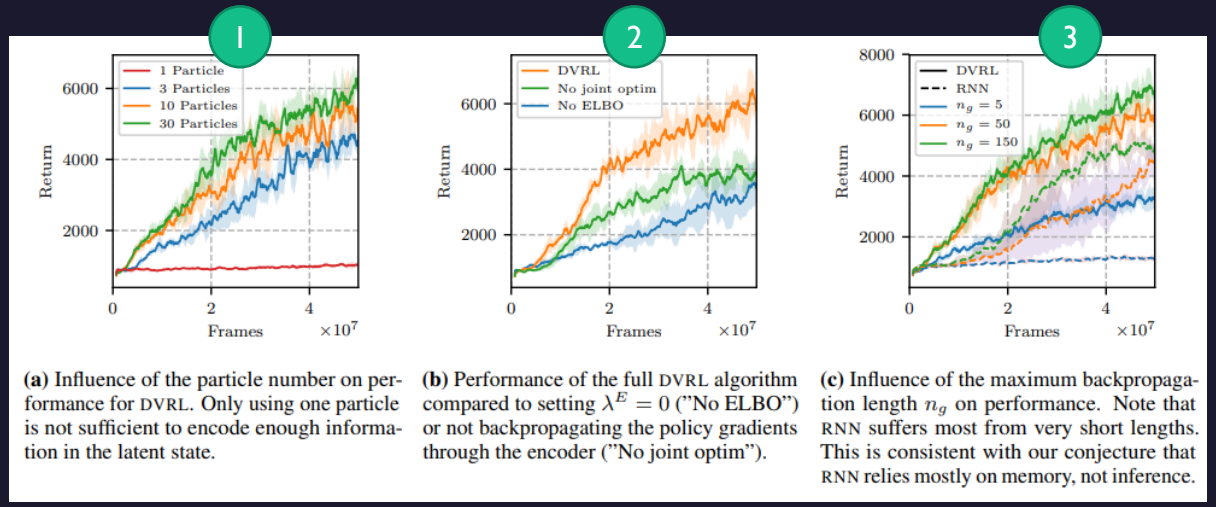
Env	DVRL($\pm std$)	RNN($\pm std$)
Pong	**18.17**($\pm$2.67)	6.33($\pm$3.03)
Chopper	**6602**($\pm$449)	5150($\pm$488)
MsPacman	2221($\pm$199)	2312($\pm$358)
Centipede	4240($\pm$116)	4395($\pm$224)
BeamRider	1663($\pm$183)	1801($\pm$65)
Frostbite	**297**($\pm$7.85)	254($\pm$0.45)
Bowling	29.53($\pm$0.23)	**30.04**($\pm$0.18)
IceHockey	**−4.88**($\pm$0.17)	−7.10($\pm$0.60)
DDunk	**−5.95**($\pm$1.25)	−15.88($\pm$0.34)
Asteroids	1539($\pm$73)	1545($\pm$51)

Key Result

1. Using more than one particle is important to accurately approximate the belief distribution over the latent state (z, h), so the empirical experiments shows that **higher particle numbers provide better information to the policy**, leading to higher returns. Also **resampling step is necessary, otherwise we cannot approximate ELBO on only n_s observations**.

2. The inclusion of L_t^{ELBO} encourage model learning for good performance. Furthermore, **not backpropagating the policy** gradients through the encoder **and only learning it based on the ELBO also deteriorates performance**.

3. **The backpropagation lengths on both the RNN and DVRL has influence**. While increasing n_g universally helps, the key here is that **a short lenght $n_g = 5$ has a stronger negative impact on both RNN and DVRL**. This is consistent with the notion that RNN is mainly performing memory based reasoning, for which BPTT is required. The belief update for DVRL is a one step update from b_t to b_{t+1} without the need to condition on the past actions and observations (can benefit of backpropagation lenght but is not necessary). This results support that **DVRL relies more on inference computations to update the latent state**.



(a) Influence of the particle number on performance for DVRL. Only using one particle is not sufficient to encode enough information in the latent state.

(b) Performance of the full DVRL algorithm compared to setting $\lambda^E = 0$ ("No ELBO") or not backpropagating the policy gradients through the encoder ("No joint optim").

(c) Influence of the maximum backpropagation length n_g on performance. Note that RNN suffers most from very short lengths. This is consistent with our conjecture that RNN relies mostly on memory, not inference.

Conclusions

DVRL is a method for solving POMDPs given only a stream of observations, withouth knowledge ot the latent space or the transitions and observation functions operating in that space.

DVRL leverages a new ELBO-based auxiliary loss and incorporates an inductive bias into the structure of the policy network, taking advantage of our prior knowledge that an inference step is required for an optimal solution.

At the end of the empirical process, the results support the author's claim that the latent state in DVRL approximates a belief distribution in a learned model.

The authors state that access to belief network can open to a several interesting reaserch direction, investigating the role of better generalization capabilities and the more powerful latent state representation on the policy performance. DVRL can give rise to further improvements and also likely to benefit from more powerful model architecture and disentangled latent state.

Furthermore, the uncertainty of the belief state can be used for exploration in environments with sparse rewards.