PPO with Compressed RGB Inputs for Robotic Grasping Problem

Kevin Wedage
University of Alberta
Faculty of Engineering
kwedage@ualberta.ca

Amr Marey
University of Alberta
Faculty of Engineering
amarey@ualberta.ca

Abstract

This paper investigates how effective variational auto-encoders (VAEs) and $\beta-$VAEs are in reducing the dimensionality of input states for a Proximal Policy Optimization (PPO) agent in a simulated robotic grasping problem. In this work, the authors train multiple VAEs with different parameters. The PPO agents are then trained off of the latent space observations generated by the VAEs. Results indicate that certain values for the latent space dimensions and β yield better performance compared to the baseline PPO agent trained solely on the RGB image. Hence, the authors show the potential of VAE state representations as observation states for general reinforcement learning algorithms.

1. Introduction

The interaction between the agent and the environment is a crucial aspect of reinforcement learning (RL), characterized by a dynamic and adaptive process. The agent learns to approximate the optimal actions through iterative experimentation and contact based on what it is capable of observing from the environment state [7]. In the field of RL, the environment state is defined as the full description of the situation an agent has found itself in. It consists of all useful information for making decisions for an action at the current time-step and is based on a prediction of subsequent states and rewards [7]. In the context of model-based RL, where an agent maintains or learns a model of the environment's dynamics, the state is used to predict future states and rewards based on current actions. When the state fully describes the environment, this is deemed a Markov Decision Process (MDP) [7]. In general, the agent does not always have access to the full environment state. The agent's observation is what the agent perceives from the environment. This idea is especially relevant in partially observable Markov decision processes (POMDPs). The agent estimates what it can of the true environment state based on this observation. The agent then utilizes this observation to perform its actions. However, there is no guarantee of optimality with respect to the actions selected due to the agent's lack of fundamental information from the environment state [7].

The are a large number of RL problems where the environment state can be described by a single image. If the agent observes the entire image environment state, in theory it is capable of analyzing the full dynamics of the environment to obtain the optimal policy for the MDP [5]. However, based on the dimensions and pixel density of the images used, the resulting state representations can be quite large. This can result in a large computational requirement for policy learning. In addition, a large amount of the state representation may be redundant due to repetition and high detail found in the images [5].

There are several techniques in the literature that aim to overcome the issue of high-dimensionality of single image observation states. One technique involves the utilization of convolutional neural networks (CNN) [5]. CNNs are designed to automatically learn spatial hierarchies of features through back-propagation. Multiple building blocks, such as convolutional layers, pooling layers, and fully connected layers are used to extract different abstractions of features such as edges, corners, and objects found in the images [5]. Furthermore, one can use an auto-encoder to learn efficient and compact representations of the high-dimensional image data. An auto-encoder consists of two structures, which are the encoder and the decoder. The encoder and decoder subsystems are trained together, by forcing the encoder to learn a compressed latent representation and requiring the decoder to reconstruct the original image from this compressed data [2]. This is accomplished by forcing the output dimension of the encoder to be smaller that the input, and training the encoder and decoder subsystems using images with a reconstruction loss, which compares the similarity between the original image and the resulting image generated by the decoder. In theory, the encoder should learn to preserve the most significant information that can fit within the constrained latent representation [2].

Two popular variants of the auto-encoder are the variational auto-encoder (VAE) and the $\beta -$ VAE. These two variants encode the image data using a prior that assumes that the latent space follows a multi-dimensional Gaussian [2]. This is achieved by requiring the model to output mean and variances (log variances), and furthermore introduces a similarity loss that encourages the resulting latent space to be similar to a multi-dimensional normal distribution. Formally, these models are trained based on the image reconstruction loss and the KL divergence between the learned distribution and a prior distribution. The β parameter in the $\beta -$ VAE allows for the relative importance of each constituent loss [2].

This paper aims to explore the utilization of VAEs/$\beta -$ VAEs as state-representations in an RL setting and determine if they provide significant advantages as opposed to full image states. We consider the following questions. **Does utilizing VAEs allow for better performance or faster policy convergence on the Robotic Grasping Problem? Is there an optimal latent representation dimension for VAEs/$\beta -$ VAEs? Can $\beta -$ VAEs out perform VAEs? Similarity, how does adjusting the β value affect the performance of $\beta -$ VAE.**

The rest of this paper is divided as follows. A review of VAEs is discussed in section 2. The methodology and experiment design is discussed in 3. The results are presented and discussed in 4. Section 5 presents some future work to consider. Finally, the paper is concluded in section 6. The authors show the code used for this report in section 7.

2. Auto-Encoder State Representation

An image state-representation problem can be described as the following: Given an RGB image I of dimension $H \times W \times 3$ as an environment state, what compact representation of I, defined as agent state s, is optimal to allow the agent to maximize expected return. Where the W is the width, H is the height and there are 3 different color channels. Hence the objective is to derive a function f such that [4]

$$s = f(I), dim(s) \ll H \times W \times 3. \tag{1}$$

2.1. Auto-Encoders

One popular method for learning state representations that are frequently utilized for dimensionality reduction is the use of auto-encoders (AE). AEs are adept at downsizing data dimensions by learning to compress (encode) the input into a more compact form before reconstructing (decoding) the output to closely resemble the original input. AEs are composed of two parts, which are the encoder and the decoder. The encoder condenses the input data into a compact representation. The decoder reconstructs the input data from this compact form. The AE adjusts its weights throughout the training phase in order to minimize the reconstruction error, which is commonly measured as the difference between the network's input data and output data. This approach enables the AE to recognize and pick up features unique to the dataset it is trained on in an unsupervised fashion [3].

2.2. Variational Auto-Encoders

VAEs incorporate probabilistic techniques to model the latent space more flexibly than classic AEs, such that the generative capabilities can improve. VAEs transform an input into a distribution across the latent space as opposed to encoding it as a single point in the latent space. Within this space, each input I is represented by two parameters: variance σ_I^2 and mean μ_I. The latent representation is sampled based on a Gaussian distribution that is defined by these parameters. By using a probabilistic approach, VAEs can model the input data in a way that better represents the data's underlying distribution. As a result, this makes it possible to produce the reconstructed data points that closely resemble the ones in the training set [3].

Fig. 1 Summarizes the structure of VAEs. The original input image is denoted by I. The reconstructed image is denoted as $\hat{I}$. The reconstructed image is reconstructed from the latent vector s as defined by

$$\hat{I} = g(s). \tag{2}$$

Where $g(.)$ denotes the nonlinear decoder mapping to be learned. The reconstruction loss L_r is defined as [3]

$$L_r = ||I - \hat{I}||^2 = ||I - g(s)||. \tag{3}$$

VAEs also incorporate a Kullback-Leibler (KL) divergence term in their loss function, referred to as the similarity loss L_s. This term imposes a penalty on the encoder if the distributions it learns deviates from a predefined prior distribution, usually a standard Gaussian $\mathcal{N}(0, I)$. This regularization component promotes a well-organized and continuous latent space, which is essential for producing coherent samples within the lower dimensional latent space [3]. This similarity loss can be written as

$$L_s = D_{KL}(\mathcal{N}(\mu_I, \sigma_I^2), \mathcal{N}(0, I)). \tag{4}$$

Thus, the total loss L_{VAE} being minimized can be defined by

$$L_{VAE} = L_r + L_s. \tag{5}$$

2.3. β-Variational Auto-Encoders

$\beta-$VAEs build upon the framework of traditional VAEs by introducing an adjustable hyper-parameter β that explicitly controls the significance between the reconstruction loss and the similarity loss [3]. The $\beta-$VAE operates in a similar fashion to the VAE except its loss function to be minimized is

$$L_{\beta-VAE} = L_r + \beta L_s, \beta > 0. \tag{6}$$

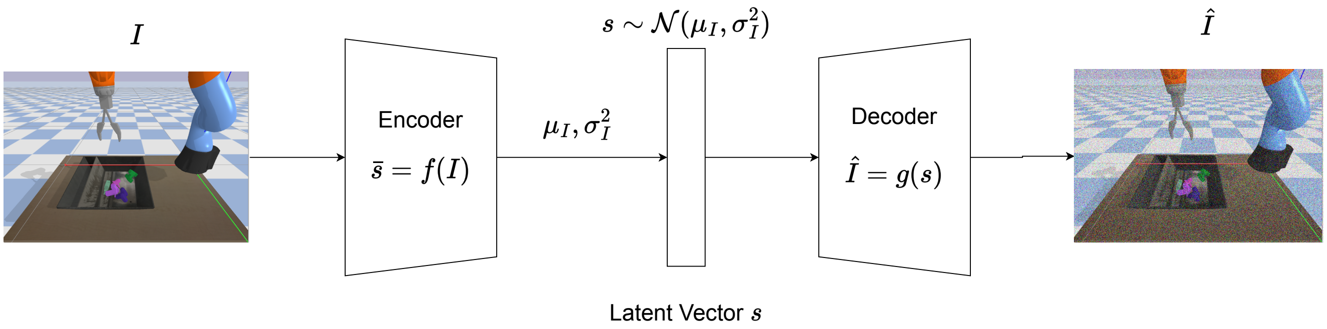


Figure 1. A high level overview of the structure of a VAE. The image I is inputted into the encoder to produce a mapping to the mean μ_I and variance σ_I^2. The latent vector is sampled from a Gaussian distribution with this mean and variance. The decoder uses this latent vector to produce a reconstructed image $\hat{I}$.

Increasing β places more relative importance on the similarity loss, forcing better disentanglement by driving the latent variables to be more independent. This tends to worsen the quality of the reconstruction, since the model cares more about organizing the latent space than making an accurate reconstruction of the data. Lower values of β reduce the impact of the similarity loss, allowing the model to focus more on minimizing the reconstruction error. This typically results in better reconstruction performance but less emphasis on the similarity loss [3].

One can note that a traditional VAE is simply a $\beta-$VAE with $\beta = 1$.

3. Methodology and Experimentally Design

In this work, we utilize the Kuka Diverse Object gym environment, released by PyBullet to simulate and train the RL agents for the Robotic Grasping Problem [1]. We propose training multiple auto-encoders with different latent dimensions and different β values on RGB observations obtained from the gym environment. Following this we train a PPO agent [6] using observations that are compressed using different auto-encoders. Finally, we evaluate each PPO agent and compare the number of objects successfully picked up over multiple runs. A high-level overview of the experimental set-up is presented in Fig. 2.

3.1. Task

The Robotic Grasping Problem is a task that involves training an RL agent (robotic arm) to move and pick up rigid body objects in a 3D scene. In this work, we allow the agent to take continuous motion actions and utilize RGB images, obtained from a simulated 2D camera, as input observations. The overall goal of the RL agent is to pick up a single object in the scene, and it can accomplish this by moving the robotic arm in 3D space and opening/closing the grasping hand. An example of the 3D scene and 2D observation is showcased in Fig. 3.

3.2. Training

3.2.1 Auto-Encoders

The dataset used for training the auto-encoders was obtained by saving the full RGB image observations for every time-step throughout the training of the baseline PPO agent. This consisted of 111k $48 \times 48 \times 3$ images. Each variational auto-encoder was trained with a distinct latent dimension and β value. The architecture of the auto-encoders consisted of encoder and decoder subsystems. The encoder was pre-trained with a ResNet18 backbone with the last two layers removed and two separate linear layers added. Each of the additional two layers produces a separate output vector corresponding to the mean and log-variances. Each of these vectors has a length equal to the latent dimension. During training, the batch size was specified to 64, the learning rate is 10^{-3} and the number of epochs is set to 10. We considered latent dimensions of 1, 2, 3, 4, 5, 7, 10, 25, 100, and 1000. For each latent dimension we considered β values of 0.1, 1, and 10.

3.2.2 PPO Agent

To train the PPO agents we utilized the stable_baseline3 PPO implementation [1]. We utilized a custom Kuka Diverse Object environment, which allowed us to update the observation space and corresponding get_observation method. This allows us to utilize the encoder of a previously trained auto-encoder, to produce a condensed observation of our specified latent dimension. We trained several PPO agents using different auto-encoders. Each PPO agent was trained using 100k time steps, a maximum of 20 steps per episode, a batch size of 1024 and a non-dense reward. Afterwards, we evaluated each trained agent using the stable_baseline3 evaluate_policy method, which averaged the reward over 100 evaluation episodes.

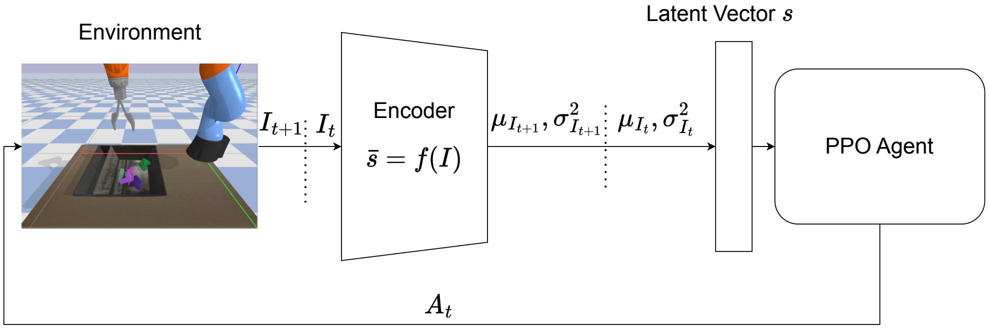
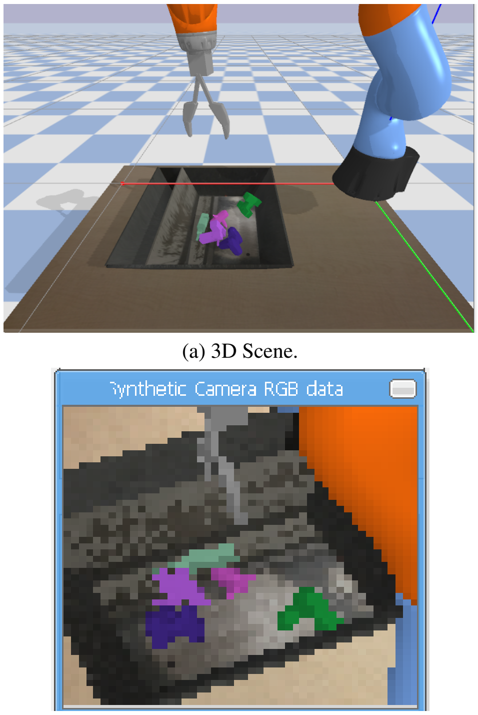


Figure 2. A high-level overview of the Gym environment. The environment state at time-step t is described by image I_t. This image is put into the encoder of a VAE to produce the latent vector s_t. The PPO agent learns to optimize its policy based on this latent state to produce action A_t. to change the environment state at time-step $t+1$. This cycle is repeated at time-step $t+1$ until the end of the episode.



(a) 3D Scene.

(b) 2D RGB observation.

Figure 3. Example of the 3D scene and corresponding RGB observation from the simulated 2D camera.

4. Results and Discussion

4.1. Baseline

We trained a baseline PPO agent using the original RGB images, each of dimension $48 \times 48 \times 3$. The training curves for the PPO agent are provided in Fig. 4 and Fig. 5. At the end of the training, the mean episode reward was 0.42. This is the baseline that has allowed the authors to compare the performance of the VAEs.

4.2. Overall Results

The overall results of using different combinations of latent dimensions and β values are shown in Fig. 8. Empirical results suggest that lower latent dimensions are more stable and less impacted by the choice of β value. Furthermore, larger latent dimensions performance degrades significantly with larger β values. The highest mean reward, of 0.54, was obtained with a latent dimension of 10 and a β value of 0.1. The training curves for the PPO agent using this VAE is shown in Fig. 6 and Fig. 7.

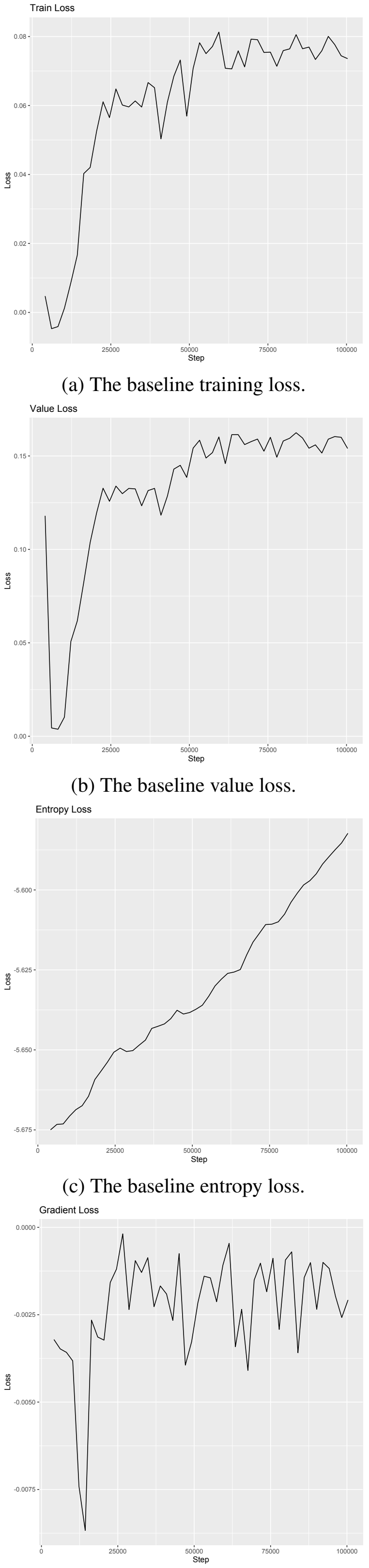
4.3. Discussion

Does utilizing VAEs allow for better performance or faster policy convergence on the Robotic Grasping Problem? Using VAEs allowed us to achieve better performance as measured by the average evaluation episode reward compared to our baseline.

We can use the average episode length and the average episode reward as an indicator to see how long it takes for a given PPO agent to converge. For the baseline, it appears to converge at around 65k step, as shown in Fig. 5. However, for the PPO agent trained using a VAE with a latent dimension of 10 and β of 0.1, the minimum episode length occurs at around the final step of 100k steps, as shown in Fig. 7. This implies that using a VAE with a smaller dimension does not guarantee a faster convergence.

Is there an optimal latent representation dimension for VAEs/$\beta -$ VAEs? The following VAEs achieved the highest performances: (latent_dim=2,β=1), (latent_dim=5, β=10) and (latent_dim=10,β=0.1). However, it appears that a latent dimension of 2 is the most stable and less impacted by the choice of β values.

Can $\beta -$ VAEs out perform VAEs? Similarity, how does adjusting the β value affect the performance of $\beta -$ VAE. In some instances $\beta -$ VAEs outperformed VAEs. This is shown with latent dimensions of 3, 5, 10, 25, 50, 100, and 1000. As mentioned prior, empirical results sug-



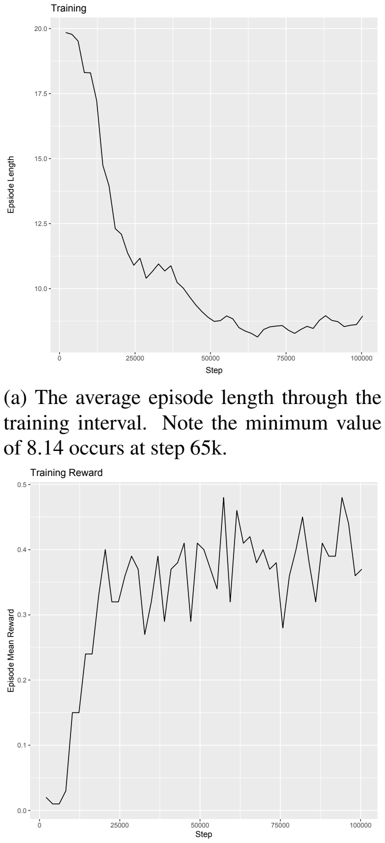
(a) The baseline training loss.

(b) The baseline value loss.

(c) The baseline entropy loss.

(d) The baseline gradient loss.

Figure 4. The baseline losses obtained by training the PPO agent solely through the RGB images without the VAE.



(a) The average episode length through the training interval. Note the minimum value of 8.14 occurs at step 65k.

(b) The mean reward through the training interval.

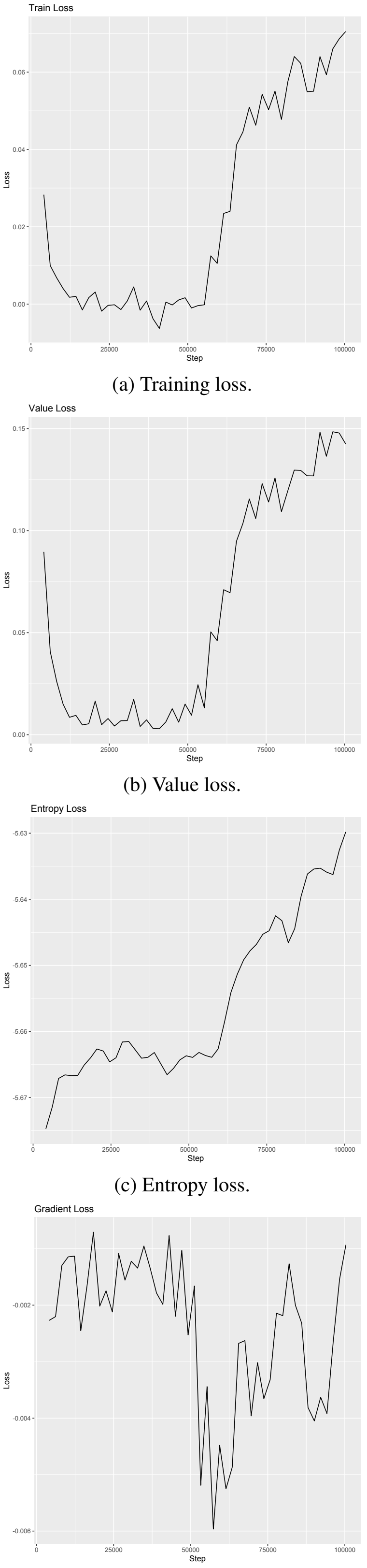
Figure 5. Information regarding mean episode length and reward throughout the training process.

gest that for larger latent dimensions, smaller β values are preferred. Specifically for latent dimension 2, there is no clear pattern of performance gain with respect to changes in β values, as shown in Fig. 9. This reflects the finding that the latent dimension of 2 is stable.

5. Future Work

There are multiple ways this work could be built on. Firstly, the authors only explored the utilization of VAEs within the PPO policy learning algorithm. There are no guarantees that the results presented in this paper would generalize to other policy learning algorithms such as REINFORCE or TPRO [6]. Hence, the authors plan on replicating the results of these experiments to using other policy learning algorithms.

Secondly, the simulated robot grasping task considered within this paper is relatively simple compared to other RL tasks considered within the literature. An example of several more complicated tasks include playing Atari video games, navigating certain maps, etc. Furthermore, how robust are the results in this paper when conducted in different



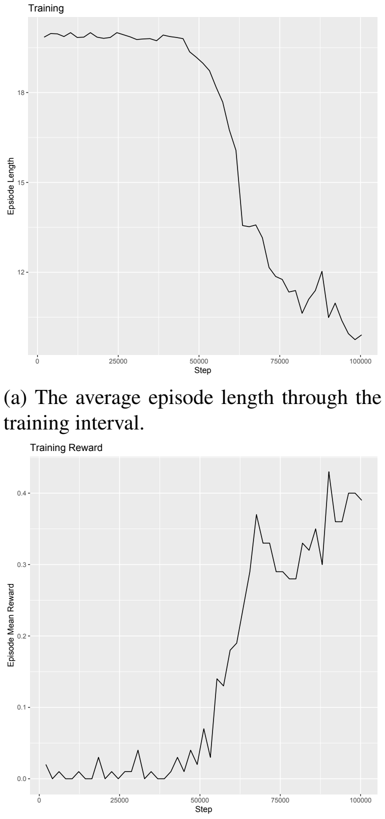
(a) Training loss.

(b) Value loss.

(c) Entropy loss.

(d) Gradient loss.

Figure 6. The training losses obtained by training the PPO agent using a VAE with a latent dimension of 10 and β of 0.1.



(a) The average episode length through the training interval.

(b) The mean reward through the training interval.

Figure 7. Information regarding mean episode length and reward throughout the training process of the PPO agent using a VAE with a latent dimension of 10 and β of 0.1.

environment conditions? A question can be asked if one can replicate similar results for real robots.

Thirdly, further exploration is needed on the interpretation of the latent spaces in the VAEs. A question can be asked on what the learned σ_I^2 and μ_I signify in relation to the robot grasping task. For example, does the latent space represent the 2D location of the objects? Furthermore, this paper explored the effect of different values of β for the $\beta-$VAE and presented the results. However, there is no explanation for why these results came to be. In the future, the authors should plan a detailed data analysis on the latent space of the VAEs variants to make sense of the results presented in this paper.

6. Conclusion

This paper explores the utilization of variational autoencoders (VAE) and $\beta-$VAEs as state-representation methods for a Proximal Policy Optimization (PPO) agent. One of the applications of VAEs include the representation of high-dimensional data into a lower-dimensional latent space. The

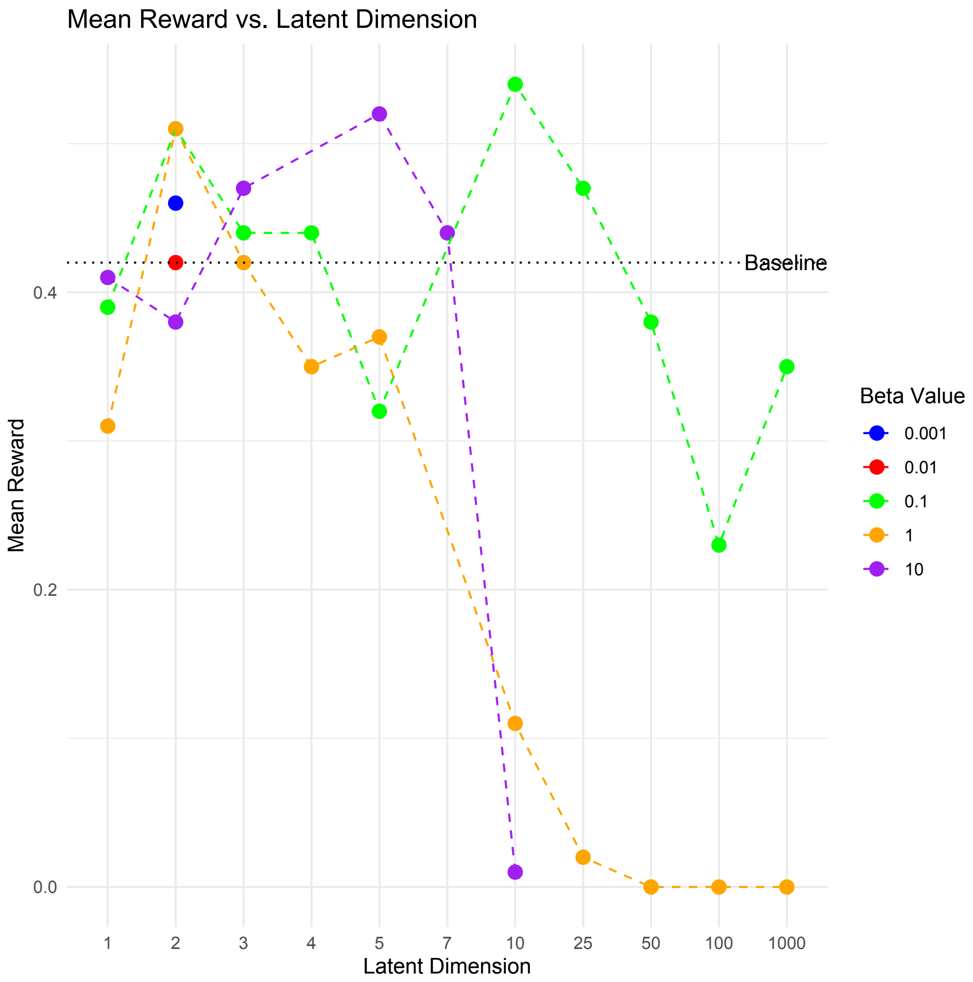


Figure 8. The arithmetic mean reward over 100 evaluation episodes for each distinct latent dimension and β combination. The highest mean reward was obtained by latent dimension 10 with a β value of 0.1 at 0.54.

authors considered a simple simulated robot grasping task with a PPO agent to highlight the application of VAEs within the RL framework. The experiments showed promising results compared to the baseline RGB image state representation. Generally, the mean reward was similar to the baseline. In the future, the authors plan on exploring the application of VAEs with other policy learning RL agents.

7. Code

Code can be found at RL-Robot-Grasping-Experiments.

References

[1] Gymnasium - a farama foundation project. `https://gymnasium.farama.org/`. Accessed: 2023-05-01. 3

[2] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders. *CoRR*, abs/2003.05991, 2020. 1, 2

[3] Shuangshuang Chen and Wei Guo. Auto-encoders in deep learning—a review with new perspectives. *Mathematics*, 11 (8), 2023. 2, 3

[4] Jun Jin, Masood Dehghan, Laura Petrich, Steven Weikai Lu, and Martin Jagersand. Evaluation of state representation methods in robot hand-eye coordination learning from demonstration, 2019. 2

[5] Ngan Le, Vidhiwar Singh Rathour, Kashu Yamazaki, Khoa Luu, and Marios Savvides. Deep reinforcement learning in computer vision: A comprehensive survey. *CoRR*, abs/2108.11510, 2021. 1

[6] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. 3, 5

[7] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018. 1

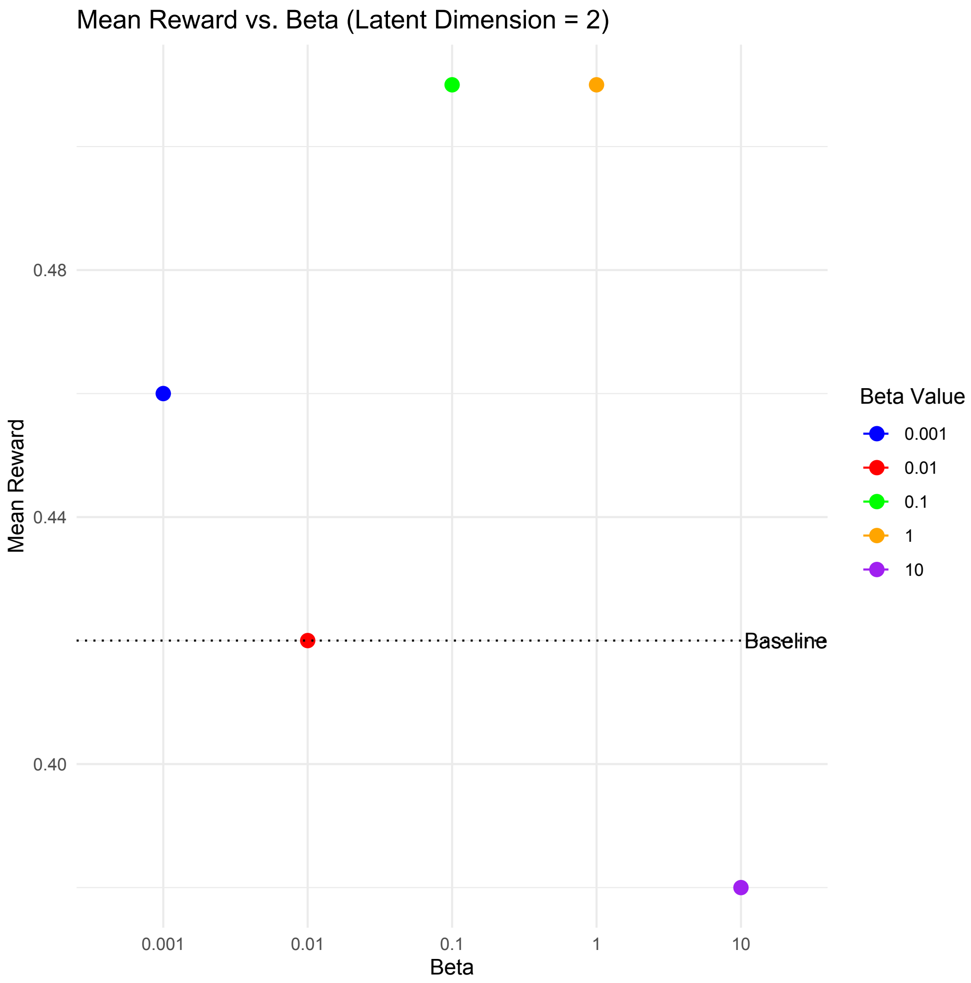


Figure 9. The arithmetic mean reward over 100 evaluation episodes for different β values for a latent dimension of 2.