Stochastic Video Prediction with Deep Conditional Generative Models

Rui Shu
Stanford University
ruishu@stanford.edu

Abstract

Frame-to-frame stochasticity remains a big challenge for video prediction. The use of feed-forward and recurrent networks for video prediction often leads to averaging of future states. This effect can be attributed to the networks' limited ability to model stochasticity. We propose the use of conditional variational autoencoders (CVAE) to model frame-to-frame transitions. In addition, we provide a novel extension of the standard VAE architecture by incorporating the use of a Gaussian mixture conditional prior and resolve the issue of the intractable Kullback-Leibler divergence between Gaussian mixtures via variational approximation. We tested our proposed Gaussian mixture conditional VAE (GM-CVAE) on a simple video-prediction task involving a stochastically moving object. Our architecture demonstrated superior performance, achieving significantly lower rates of blurring/averaging in comparison to a feed-forward network and a vanilla conditional VAE.

1. Introduction

Deep reinforcement learning has largely focused on the use of deep architectures with model-free reinforcement learning techniques to tackle complex, high-dimensional tasks. Most notably, Deep Q-Networks (DQN) have demonstrated remarkable success in matching (and sometimes surpassing) human-level control on a variety of Atari 2600 games [10, 5]. Recently, the combination of deep learning with Monte Carlo Tree Search (MCTS) has also garnered much success in games with long-term rewards [14]. However, the use of MCTS requires learning the transition model [9], making it challenging to use MCTS and other model-based approaches in vision-based reinforcement learning.

The difficulty stems from the fact that a model-based approach for vision-based reinforcement learning requires one to perform next-frame prediction conditioned on the current frame and the current action taken by the agent. Modeling video is in and of itself a challenging endeavor, and recent studies have focused on modeling simple video data where frame-to-frame transitions are largely deterministic [11]. Attempts to perform transition state modeling in vision-based RL via the use of action-conditional video prediction have successfully captured the movements of determnistic elements within the visual environment, but have great difficulty handling stochasticity [11].

It has previously been noted that "highly-structured" data are data that have high signal-to-noise ratio and contain a complex relationship between the underlying factors of variation and the observed data [1]. While the research done by [1] focused mainly on speech data, we believe the same characteristics are present in video data. The existence of stochasticity means that a deterministic network cannot be used. Furthermore, the structured nature of video and the means by which the underlying factors of variation affect the observed data suggest that sampling pixels independently (as is the case with a standard RNN) is not sufficient.

In this paper, we focus on the challenge of modeling stochastic elements in videos. We demonstrate that a conditional variational autoencoder can successfully handle the highly-structured yet stochastic nature of video data. We further demonstrate that the use of a conditional variational autoencoder with Gaussian posterior and Gaussian mixture prior can significantly improve sampling accuracy.

2. Related Work

2.1. Action-Conditional Video Prediction

[11] proposed the use of multiplicative action-conditional transformations to tackle the problem of modeling the transition function for vision-based reinforcement learning. Their architecture involved the use of an autoencoder framework. To perform an action-conditional transition, they perform their frame transition in the encoding space, generating a new encoding vector by using a multiplicative interaction between the action vector and the encoding of the current frame. In particular,

$$\mathbf{h}_t^{dec} = \mathbf{W}^{dec}(\mathbf{W}^{enc}\mathbf{h}_t^{enc} \odot \mathbf{W}^a \mathbf{a}_t) + \mathbf{b}, \qquad (1)$$

where $\mathbf{h}_t^{enc}$ is the encoding of the current frame, $\mathbf{h}_t^{dec}$ is the action-transformed encoding, $\mathbf{a}_t$ is the one-hot vector repre-

sentation of the action at time t, and $\mathbf{W}^{dec}$, $\mathbf{W}^{enc}$, $\mathbf{W}^{a}$, and $\mathbf{b}$ are learned weights for performing the multiplicative interaction. This approach relies on a deterministic transition in the encoding space and, as a result, is limited in its ability to deal with stochasticity. Most notably, their proposed architecture is unable deal with the stochastic movements of the ghosts in Ms. Pacman, as can be seen in their video at https://youtu.be/cy96rtUdBuE.

2.2. Deep Generative Models

In order to develop an accurate transition model for vision-based reinforcement learning with stochastic elements, it is critical that the predicted next-frame must be generated probabilistically in alignment with the distribution of the true transition model. Given the high-dimensionality of frame-to-frame transitions, it is natural to consider the use of deep generative models.

Two prominent approaches have gained much popularity in the recent literature on deep generative models. [3] proposed the use of generative adversarial networks (GAN), which employs an adversarial framework to train a generative model that mimics the true transition model. GANs, however, are well-known to be difficult to train, and the application of GANs as conditional generative models is still in its infancy [2, 12]. Since transition models for vision-based reinforcement learning requires conditioning on the current frame (a very high-dimensional variable), we believe that a variational autoencoder is more suited for the task.

In contrast to generative adversarial networks, variational autoencoders were developed as a variational Bayesian approach to deep generative models [7, 8]. At its core, the variational autoencoder is a directed graphical model involving a latent variable $\mathbf{z}$ generated from the prior distribution $p_\theta(\mathbf{z})$ and the data $\mathbf{y}$ generated from $p_\theta(\mathbf{y} \mid \mathbf{z})$ (which we denote the generative distribution).

In general, computing the posterior $p_\theta(\mathbf{z} \mid \mathbf{y})$ (also called the recognition distribution) is intractable. We therefore compute an approximation $q_\phi(\mathbf{z} \mid \mathbf{y})$ using variational Bayesian inference. Under the variational Bayesian framework,

$$\log p_\theta(\mathbf{y}) \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{y})}[-\log q_\phi(\mathbf{z} \mid \mathbf{y}) + \log p_\theta(\mathbf{y}, \mathbf{z})] \quad (2)$$

$$= -\mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z} \mid \mathbf{y}) \| p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{y})}[\log p_\theta(\mathbf{y} \mid \mathbf{z})] \quad (3)$$

where the RHS serves as the variational lowerbound, $\mathcal{L}_{\text{VAE}}$. [8] proposes to use the variational lowerbound as a surrogate objective function, thus allowing the parameters of the VAE to be learned via stochastic gradient descent.

In practice, neural networks are used to model the recognition and generative distributions. The expectation term in the variational lowerbound is then approximated with Monte Carlo estimation. Assuming Gaussian latent variables, the empirical estimate of the objective for Gaussian-VAE is then written as,

$$\tilde{\mathcal{L}}_{\text{VAE}}(\mathbf{y}; \theta, \phi) = -\mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z} \mid \mathbf{y}) \| p_\theta(\mathbf{z})) + \frac{1}{L} \sum_{l=1}^{L} \log p_\theta(\mathbf{y} \mid \mathbf{z}^{(l)}), \quad (4)$$

where $\mathbf{z}^{(l)}$ is a random sample drawn from the recognition distribution $q_\phi(\mathbf{z} \mid \mathbf{y})$. Suppose the recognition distribution parameterizes $q_\phi(\mathbf{z} \mid \mathbf{y})$ as $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$. We can introduce a function $g_\phi(\mathbf{y}, \epsilon^{(l)})$ that computes the parameters $\boldsymbol{\mu}, \boldsymbol{\sigma}$ for given $\mathbf{y}$ and performs the transformation $g_\phi(\mathbf{y}, \epsilon^{(l)}) = \boldsymbol{\mu} + \boldsymbol{\sigma}\epsilon^{(l)}$. This allows the use of the reparameterization trick, exploiting the fact that sampling $\mathbf{z}^{(l)} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ can be rewritten as $\mathbf{z}^{(l)} = g_\phi(\mathbf{y}, \epsilon^{(l)})$, where $\epsilon^{(l)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. This trick allows backpropagation through the Gaussian latent variables, which is crucial when training the VAE. We elaborate on the reparameterization trick in §4.1.

2.3. Conditional Variational Autoencoder

Because the transition model involves conditioning on the current state and action, the variational autoencoder cannot be applied directly. Instead, we use a natural extension of the framework called the conditional variational autoencoder. The CVAE framework extends VAE simply by including the additional factor that the prior, recognition, and generative models may change when conditioned on a third variable $\mathbf{x}$. The new exact and empirical objective now take

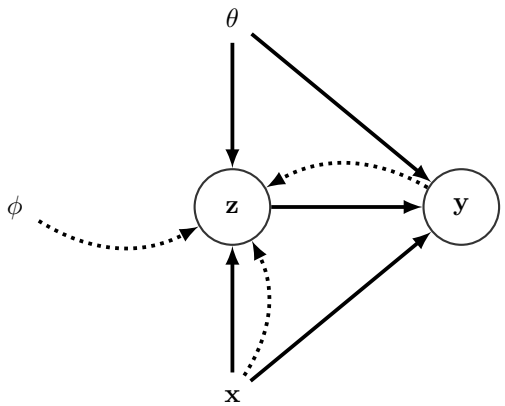


Figure 1. The directed graphical model associated with the conditional variational autoencoder architecture. Solid lines denote the generative model $p_\theta(\mathbf{z} \mid \mathbf{x})p_\theta(\mathbf{y} \mid \mathbf{z}, \mathbf{x})$. Dashed lines denote the variational approximation $q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{y})$ to the intractable posterior $p_\theta(\mathbf{z} \mid \mathbf{x}, \mathbf{y})$. The variational ϕ and generative θ parameters are learned jointly.

their respective forms,

$$\begin{aligned}\mathcal{L}_{\text{CVAE}}(\mathbf{x},\mathbf{y};\theta,\phi) =& -\mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}\mid\mathbf{x},\mathbf{y})\|p_\theta(\mathbf{z}\mid\mathbf{x})) \\ &+\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})}[\log p_\theta(\mathbf{y}\mid\mathbf{x},\mathbf{z})] \end{aligned} \tag{5}$$

$$\begin{aligned}\tilde{\mathcal{L}}_{\text{CVAE}}(\mathbf{x},\mathbf{y};\theta,\phi) =& -\mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}\mid\mathbf{x},\mathbf{y})\|p_\theta(\mathbf{z}\mid\mathbf{x})) \\ &+\frac{1}{L}\sum_{l=1}^{L}\log p_\theta(\mathbf{y}\mid\mathbf{x},\mathbf{z}^{(l)}), \end{aligned} \tag{6}$$

Note the high degree of similarity between (4) and (6). Furthermore, if $\mathbf{x}$ is simply a non-informative constant, (6) reduces to (4). The graphical model associated with CVAE is illustrated in Figure 1.

3. Evaluation of Gaussian-CVAE

By far the most common set-up is to assume that both the prior and recognition models are Gaussian distributions [1, 4, 15]. However, not only is it not necessary to assume a Gaussian distribution, it is also not required for the prior and recognition distribution to come from the same family of distributions. Within the context of encoding images (and, by extension, video frames), it is therefore important to critically evaluate the appropriateness of both the Gaussian prior and the Gaussian posterior.

3.1. Gaussian Posterior

Broadly speaking, the conditional variational autoencoder can be thought of as: given $\mathbf{x}$, try to encode $\mathbf{y}$ into $\mathbf{z}$. It seems intuitively desirable for $\mathbf{y}$ to have exactly one encoding $\mathbf{z}$. Thus, the use of a unimodal Gaussian distribution for the recognition model feels most natural, with the hope that recognition model learns to generate a distribution $q_\phi(\mathbf{z}\mid\mathbf{x},\mathbf{y})$ that is simultaneously singly-peaked and has small variance.

3.2. Gaussian Prior

The use of a Gaussian prior, however, is noticably limiting. As part of the variational Bayesian framework, it is vital that we have a good variational approximation of the posterior. If it is true that we have a reliable variational approximation of the posterior, it follows that,

$$p_\theta(\mathbf{z}\mid\mathbf{x},\mathbf{y}) \approx q_\phi(\mathbf{z}\mid\mathbf{x},\mathbf{y}) \tag{7}$$

$$p_\theta(\mathbf{z}\mid\mathbf{x}) \approx \int_{\mathbf{y}} p_\theta(\mathbf{y}\mid\mathbf{x})q_\phi(\mathbf{z}\mid\mathbf{x},\mathbf{y})d\mathbf{y}. \tag{8}$$

Since the RHS is the true encoding distribution learned by our autoencoder, (8) essentially describes that a necessary condition is for the prior to match the encoding distribution. This makes intuitive sense too; because we draw from the prior distribution during sampling time, it is crucial that prior distribution reflects the actual encoding distribution that our autoencoder has learned. As such, the use of the Gaussian prior imposes the strict requirement that the encoding distribution cannot be multi-modal. Otherwise, the trained CVAE will perform poorly when used as a generative model. This poses a serious limitation, as there are many situations where a multi-modal encoding distribution arises naturally. To address these issues, we propose an alternative to the Gaussian-CVAE.

4. Proposed Architecture

In this section, we provide an example of an architecture that solves the issue mentioned in §3.2. Our proposed model, the Gaussian mixture CVAE (GM-CVAE), uses a Gaussian for the recognition model but a Gaussian mixture for the conditional prior model. Following [8], the choice of distribution for the generative model can either be Gaussian or Bernoulli for continuous and binary data respectively. For simplicity, we focus on the use of binary data.

Let the conditional prior model be a neural network that outputs parameters parameterizing a mixture of k Gaussians with m dimensions. For simplicity, we assume all component Gaussians to have diagonal covariance structure. Thus,

$$p_\theta(\mathbf{z}\mid\mathbf{x}) = \sum_{i=1}^{k}\boldsymbol{\pi}_i\mathcal{N}(\mathbf{z};\boldsymbol{\mu}_i,\boldsymbol{\sigma}_i^2), \tag{9}$$

where $\boldsymbol{\mu}_i \in \mathbb{R}^m, \log\boldsymbol{\sigma}_i^2 \in \mathbb{R}^m, \boldsymbol{\pi} \in \mathbb{R}^k$ are the output of the conditional prior network. To ensure $\sum_{i=1}^{k}\boldsymbol{\pi}_i = 1$, we use a softmax layer to generate $\boldsymbol{\pi}$. The recognition network generates an m-dimensional Gaussian distribution, and thus follows a similar set-up described in [8],

$$q_\phi(\mathbf{z}\mid\mathbf{x},\mathbf{y}) = \mathcal{N}(\mathbf{z};\boldsymbol{\mu},\boldsymbol{\sigma}^2), \tag{10}$$

where $\boldsymbol{\mu} \in \mathbb{R}^m, \log\boldsymbol{\sigma}^2 \in \mathbb{R}^m$ are the output of the recognition network. Finally, the generative model models each pixel using an independent Bernoulli distribution. For an image with n pixels,

$$p_\theta(\mathbf{y}\mid\mathbf{x},\mathbf{z}) = \mathbf{p}, \tag{11}$$

where $\mathbf{p} \in \mathbb{R}^n$ is the output of the generative network denoting the Bernoulli parameter for each pixel.

4.1. Reparameterization Trick

In the original VAE architecture proposal by [8], several other families of distributions were considered for the posterior and prior. The use of a Gaussian mixture, however, was not mentioned. We suspect that this is in part because the use of a Gaussian mixture for the posterior distribution makes the application of the reparameterization trick challenging. The reparameterization trick is a vital component of the algorithm, because it allows one to easily take the

derivative of $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})}[\log p_\theta(\mathbf{y} \mid \mathbf{x}, \mathbf{z})]$ with respect to ϕ. It does so by noting that,

$$\mathbb{E}_{\mathcal{N}(\mathbf{z};\boldsymbol{\mu},\boldsymbol{\sigma}^2)}[f(\mathbf{z})] = \mathbb{E}_{\mathcal{N}(\epsilon;\mathbf{0},\mathbf{I})}[f(\boldsymbol{\mu} + \boldsymbol{\sigma}\epsilon)] \tag{12}$$

$$\approx \frac{1}{L}\sum_{l=1}^{L} f(\boldsymbol{\mu} + \boldsymbol{\sigma}\epsilon^{(l)}), \tag{13}$$

where $\epsilon^{(l)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, $f(\mathbf{z}) = \log p_\theta(\mathbf{y} \mid \mathbf{x}, \mathbf{z})$. If the posterior distribution were to be a Gaussian mixture, however, it is not readily apparent how the reparameterization trick can be applied. Fortunately, our proposed model restricts the use of the Gaussian mixture to the prior and retains the use of a Gaussian distribution for the posterior. Thus, we are able to use the original reparameterization trick as-is.

4.2. Objective Function

Another issue with the introduction of a Gaussian mixture stems from the intractability of the KL-divergence term when a Gaussian mixture is involved. One solution, proposed by [8], is to perform Monte Carlo estimation of the KL-divergence term (they denote this as the generic Stochastic Gradient Variational Bayes estimator $\tilde{\mathcal{L}}^A$). This approach, however, introduces significant variance into the estimator and causes the gradients to become very noisy, curtailing the effectiveness of gradient descent. Fortunately, [6] showed that it was possible to derive a closed-form variational approximation of the KL-divergence term between two Gaussian mixtures f and g,

$$\mathrm{D}_{\mathrm{KL}}(f\|g) \approx \sum_i \boldsymbol{w}_i \log \frac{\sum_{i'} \boldsymbol{w}_{i'} \exp(-\mathrm{D}_{\mathrm{KL}}(f_i\|f_{i'}))}{\sum_j \boldsymbol{\pi}_j \exp(-\mathrm{D}_{\mathrm{KL}}(f_i\|g_j))}. \tag{14}$$

where $f_i, f_{i'}, g_j$ indexes the component Gaussians of the respective Gaussian mixtures, and $\boldsymbol{w}$ and $\boldsymbol{\pi}$ denote the component weights. Because only our prior distribution uses a Gaussian mixture, we set f to be Gaussian and the expression simplifies to,

$$\mathrm{D}_{\mathrm{KL}}(f\|g) \approx \log \frac{1}{\sum_j \boldsymbol{\pi}_j \exp(-\mathrm{D}_{\mathrm{KL}}(f\|g_j))} \tag{15}$$

$$= \mathrm{D}_{\mathrm{vKL}}(f\|g). \tag{16}$$

We denote the RHS as $\mathrm{D}_{\mathrm{vKL}}(f\|g)$, the variational approximate KL-divergence. It is now possible to construct the objective function used by the GM-CVAE model,

$$\tilde{\mathcal{L}}_{\text{GM-CVAE}} = -\mathrm{D}_{\mathrm{vKL}}(q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{y})\|p_\theta(\mathbf{z} \mid \mathbf{x})) + \frac{1}{L}\sum_{l=1}^{L} \log p_\theta(\mathbf{y} \mid \mathbf{x}, \mathbf{z}^{(l)}), \tag{17}$$

where $q_\phi(\mathbf{z} \mid \mathbf{x}, \mathbf{y})$ is a Gaussian distribution produced by the recognition model, $p_\theta(\mathbf{z} \mid \mathbf{x})$ is a Gaussian mixture produced by the conditional prior model, and $p_\theta(\mathbf{y} \mid \mathbf{x}, \mathbf{z}^{(l)})$ is a multivariate Bernoulli distribution produced by the generative model. In practice, we set $L = 1$ when a sufficiently large mini-batch is used. This function is fully differentable and enables the use of backpropagation to efficiently tune the model parameters θ and ϕ.

5. Experiments

We propose two experiments to demonstrate the superiority of the GM-CVAE. The first experiment is a toy dataset that specifically reflects the inability of the Gaussian-CVAE to handle a multi-modal encoding distribution. The second experiment illustrates the application of GM-CVAE to modeling frame-to-frame transitions when the video contains a stochastically moving object.

5.1. Two-Image Dataset

We propose an image dataset consisting only of two distinct images (see Figure 2). As such, an optimal encoding will necessarily result in an encoding distribution with two peaks. For both the Gaussian-CVAE and GM-CVAE, we use only a single dimensional latent variable $\mathbf{z}$. Unlike the Gaussian-CVAE, the GM-CVAE is initialized to have three component Gaussians (note that we overestimate the number of component Gaussians).

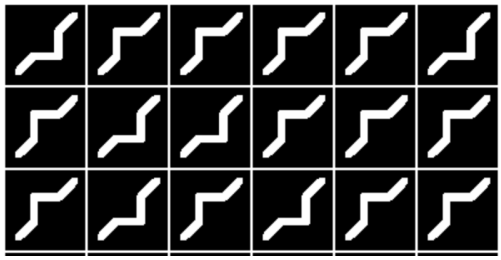


Figure 2. In this dataset, we seek to encode each image $\mathbf{y}$ into $\mathbf{z}$ using a non-informative constant for $\mathbf{x}$. An optimal encoding necessarily yields two peaks.

5.2. Stochastic Sprite Dataset

[11] has demonstrated the success of the action-conditional architecture for modeling the position of the agent in the next-frame, but not the position of a stochastic object. As such, we focus our transition modeling task solely on a stochastically moving object for simplicity (in other words, we deal with an actionless environment). We use a dataset consisting of a free-moving object that randomly chooses a direction when it reaches one of nine checkpoints spread uniformly across the video frame. Sample trajectories are shown in Figure 3. To apply the conditional variational autoencoder to next-frame prediction, simply set the current frame to be $\mathbf{x}$ and the next-frame to be $\mathbf{y}$ in (17).

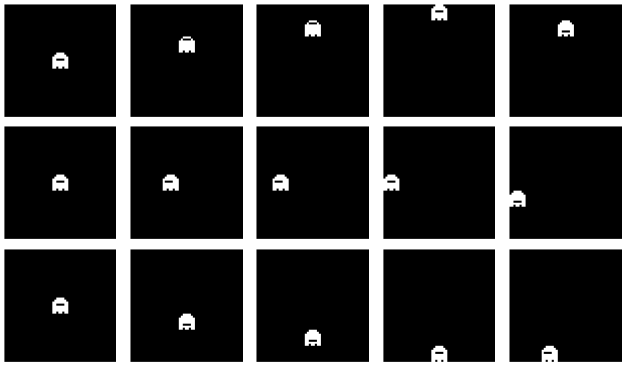


Figure 3. We use a dataset consisting of a sprite that pivots stochastically at nine locations spread uniformly across the frame. Three sampled trajectories are shown above, one in each row.

5.3. Details of Training

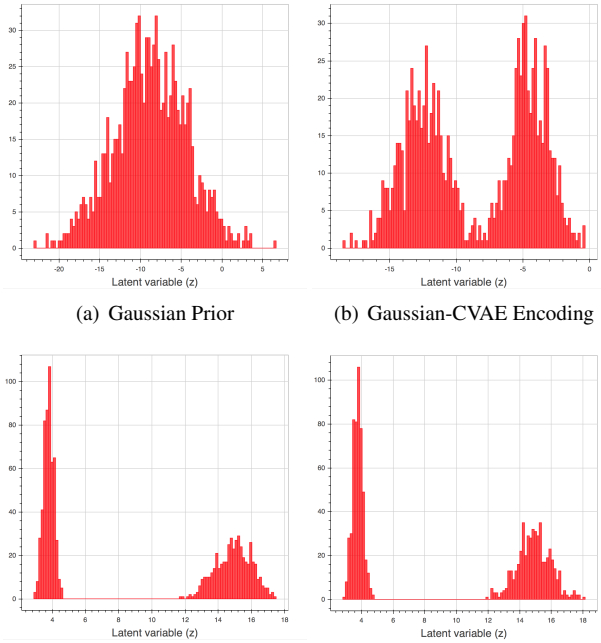
In all experiments a minibatch of 200 was used. The conditional prior, recognition, and generative networks are all constructed using two fully-connected layers. ReLU activations were applied to all hidden layers, and a learning rate of $\alpha = 0.001$ was used. The hidden layer is fixed to have 400 units. The number of units in the output layer is problem-dependent. For the two-image dataset, the Gaussian-CVAE uses a 1-dimensional Gaussian. Thus the prior and reconigition networks contain two output units (one for $\boldsymbol{\mu}$ and one for $\log \boldsymbol{\sigma}^2$). The GM-CVAE uses a 3-component 1-dimensional Gaussian mixture as the prior. Thus the prior network contains nine output units ($\boldsymbol{\mu}_{1:3}, \log \boldsymbol{\sigma}^2_{1:3}, \boldsymbol{\pi}$). The generative network has number of output units equal to the number of pixels.

For the stochastic sprite dataset, a 4-component 1-dimensional Gaussian mixture is used as the prior for the GM-CVAE. All other architecture details resemble the aforementioned architecture for the two-image dataset. All methods were implemented with Torch and trained on an NVIDIA GRID K520 GPU [13]. The code is publicly accessible at https://github.com/RuiShu/cvae.

6. Results and Analysis

6.1. Two-Image Dataset

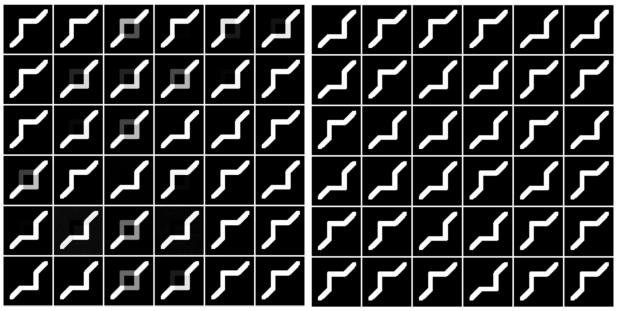
Our toy example confirms our hypothesis that the Gaussian-CVAE functions poorly when applied to a dataset whose encoding distribution is multi-modal. Not only is it easy to see a clear difference between the prior distribution versus the encoding distribution (Figure 4), but it is also possible to see the impact this has on sampling—causing the Gaussian-CVAE to be prone to averaging over the two images (Figure 5). In contrast, the GM-CVAE successfully matches the prior distribution with the encoding distribution and was far less likely to average over the two images during sampling.



(a) Gaussian Prior (b) Gaussian-CVAE Encoding

(c) Gaussian Mixture Prior (d) GM-CVAE Encoding

Figure 4. Histograms for the Gaussian prior and Gaussian-CVAE encoding shown in the first row. Using a Gaussian distribution for the prior fails to capture the multi-modality of the encoding distribution. Histograms for the Gaussian mixture prior and GM-CVAE encoding shown in the second row. Using a Gaussian mixture for the prior successfully captures the multi-modality of the encoding distribution. Note that the GM-CVAE encoding is not symmetric. This is because the solution space for the encoding distribution is non-unique. To address this issue, future work will consider regularizing the conditional prior network's output.



(a) Gaussian-CVAE Samples (b) GM-CVAE Samples

Figure 5. The Gaussian-CVAE is more likely to sample incorrectly in comparison to the GM-CVAE.

It is also important to note that the GM-CVAE produced a prior distribution with only two Gaussian components despite us initializing it with three components. We attribute this to the successful application of backpropagation, which allowed the network to learn to reduce one of the three com-

ponent contributions to zero.

6.2. Stochastic Sprite Dataset

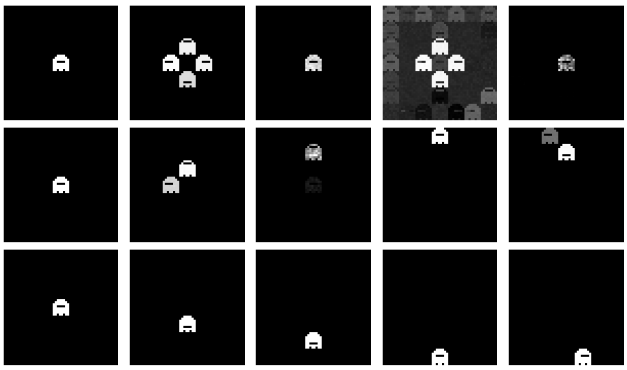


Figure 6. By row: sampled trajectories from a multi-layer perception (MLP), Gaussian-CVAE, and GM-CVAE respectively. The full video is available at: https://youtu.be/5fe30qSxW5s.

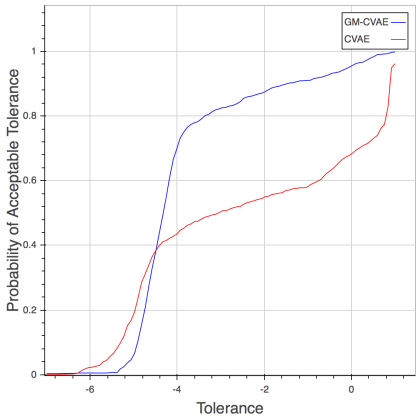


Figure 7. Semi-log plot of the probability that the generated frame is within some margin of error. The probability is plotted as a function of the tolerance value. x-axis is on $\log_{10}$ scale. The GM-CVAE framework demonstrates superior robustness, avoiding large errors when compared to Gaussian-CVAE.

As expected, a simple feedforward network suffers when attempting to predict a stochastically moving object, resulting in the averaging of multiple future states. Visual inspection further reveals that the GM-CVAE models the true transition model more accurately than the Gaussian-CVAE. The GM-CVAE is less likely to sample a next-frame that contains an averaging of multiple future states than the Gaussian-CVAE. We believe that this is directly attributable to the GM-CVAE's ability to match the prior distribution with the multi-modal encoding distribution.

Quantifying the extent to which the GM-CVAE adhered to the true transition model in comparison to the Gaussian-CVAE is a non-trivial task. As a proxy, we decided to use an simpler metric that simply quantifies the extent to which each sample frame deviates from the nearest L_2 neighbor from the sample space of the true transition model.

Concretely, let $\mathbf{X}$ be the set of all possible frames that can be sampled from the true dataset, and let $\hat{\mathbf{x}}_t$ be the frame sampled by either the GM-CVAE or the Gaussian-CVAE at time t. We define the margin of error between the sampled frame and the true dataset to be,

$$d = \min_{\mathbf{x} \in \mathbf{X}} \|\mathbf{x} - \hat{\mathbf{x}}_t\|_2. \tag{18}$$

Using this metric, it was possible for us to quantitatively ascertain that GM-CVAE is less likely to generate an incorrect frame than Gaussian-CVAE (Figure 7).

7. Conclusion

Our results demonstrate that conditional variational autoencoders successfully handles stochastic elements that exist within video. By modifying the Gaussian-CVAE to instead use a Gaussian mixture prior, we were able to significantly reduce the likelihood of sampling an incorrect frame. To the best of our knowledge, this is the first time a Gaussian mixture has been suggested and implemented for variational autoencoders, as well as the first time variational autoencoders have been applied to stochastic video prediction.

Given the promising success of the CVAE with stochastic video data, it is now possible to consider the incorporation of CVAE's into an action-conditional framework, thus creating a transition model that can be used for vision-based reinforcement learning. In our future work, we will also evaluate whether the CVAE architecture can successfully handle a visual environment with multiple stochastic objects and whether the architecture remains effective as we transition to more refined time-scales.

References

[1] J. Chung, K. Kastner, L. Dinh, K. Goel, A. Courville, and Y. Bengio. A Recurrent latent Variable Model for Sequential Data. *ArXiv e-prints*, June 2015.

[2] E. Denton, S. Chintala, A. Szlam, and R. Fergus. Deep generative image models using a laplacian pyramid of adversarial networks. *Neural Information Processing Systems*, 2015.

[3] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Neural Information Processing Systems*, 2014.

[4] K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra. DRAW: A Recurrent Neural Network For Image Generation. *International Conference on Machine Learning*, 2015.

[5] M. Hausknecht and P. Stone. Deep Recurrent Q-Learning for Partially Observable MDPs. *Association for the Advancement of Artificial Intelligence*, 2015.

[6] J. R. Hershey and P. A. Olsen. Approximating the Kullback-Leibler divergence between gaussian mixture models. *International Conference on Acoustics Speech and Signal Processing*, 2007.

[7] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling. Semi-supervised learning with deep generative models. *Neural Information Processing Systems*, 2014.

[8] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *International Conference on Learning Representations*, 2013.

[9] M. Kochenderfer. Decision Making Under Uncertainty. *MIT Press*, 2015.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 2015.

[11] J. Oh, X. Guo, H. Lee, R. Lewis, and S. Singh. Action-conditional video prediction using deep networks in atari games. *Neural Information Processing Systems*, 2015.

[12] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional enerative adversarial networks. *International Conference on Learning Representations*, 2016.

[13] C. Ronan, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. *NIPS Workshop*, 2011.

[14] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature*, 2016.

[15] K. Sohn, X. Yan, and H. Lee. Learning structured output representation using deep conditional generative models. *Neural Information Processing Systems*, 2015.