A HIERARCHICAL APPROACH TO VIDEO PREDICTION

A Thesis

by

SUPRITH BALASUBRAMANIAN

Submitted to the Graduate and Professional School of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	P. R. Kumar
Committee Members,	Nima Kalantari
	Dileep Kalathil
	Srinivas Shakkotai
Head of Department,	Aniruddha Datta

May 2022

Major Subject: Computer Engineering

ABSTRACT

The guiding design principle behind humans building machines has been the repeated execution of a particular task in a precise and efficient manner. While we have systems that can solve tasks ranging from the relatively mundane like crunching numbers to highly complex tasks like recognizing objects and harvesting wheat, they are incredibly specific in that they perform the tasks that they are trained for and not good at generalization. For instance, a robotic arm that can weld two parts of a car together may be rendered completely useless in a situation that requires welding components in a computer motherboard. Developing agents that are endowed with human-like abilities to generalize in diverse scenarios is a core research topic in artificial intelligence.

In order for an agent to develop general-purpose skills in a completely self-supervised manner, learning rich representations of the world that it is embodied in as well as using these representations to adapt and learn more about the environment are useful. The prediction and anticipation of future events is a key component of such intelligent decision-making systems. Prediction serves as a useful means to learn useful concepts about the world even from a raw stream of sensory observations, such as images from a camera. If the agent can learn to predict raw sensory observations directly, it does not need to assume availability of low-dimensional state information or an extrinsic reward signal. This is beneficial in learning skills in real-world environments, where external reward feedback is extremely sparse or non-existent, and the agent has only indirect access to the state of the world through its senses. Images are high-dimensional and rich sources of information, underlying the potential of video prediction to extract meaningful representations of the underlying patterns in video data.

Video prediction refers to the problem of generating pixels of future frames given context information in the form of past frames of a video. When combined with planning algorithms, the agent is able to take actions towards a desired goal in an unsupervised manner by using data as its own supervision. Motivated by this objective of learning generalizable behavior in the real world, we introduce the Hierarchical Variational Autoencoder (HVAE), a model that leverages a hierarchy

of latent sequences to solve the task of video prediction.

DEDICATION

To my mother, father, and my late grandparents.

ACKNOWLEDGMENTS

I would like to take this opportunity to express my sincere gratitude to my advisors, mentors, friends, and family, without whose support and guidance, this work would not have been possible.

I would like to start by expressing sincere gratitude to my advisor, Dr. P. R. Kumar, for taking me in as his advisee and supporting me to undertake this project. Dr. Kumar's focus on fundamentals, his clarity of thought on a wide array of subjects and the ability to think at the big-picture level have been a source of inspiration for me. I hope to carry forward these learnings in my research journey.

I would also like to thank my committee members, Dr. Nima Kalantari, Dr. Dileep Kalathil, and Dr. Srinivas Shakkottai, for their support. I have enjoyed taking their classes and am thankful to them for all the intellectually stimulating discussions.

I am grateful to be a part of Dr. Kumar's research group. Special thank you to Akshay Mete, Jaewon Kim and Santosh Ganji for all the research and non-research related discussions. I would also like to thank my friends Deepankar Chanda and Avinash Paliwal for the brainstorming sessions along the way.

Last but not the least, I am grateful to my parents and family who have continued to support me while going through some incredibly difficult circumstances themselves.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis (or) dissertation committee consisting of Professor P. R. Kumar [advisor], Professor Dileep Kalathil and Professor Srinivas Shakkottai of the Department of Electrical and Computer Engineering and Professor Nima Kalantari of the Department of Computer Science.

All other work conducted for the thesis (or) dissertation was completed by the student independently.

Funding Sources

The research has been supported by **Texas A&M University** under the **President's Excellence Funds X Grants Program**.

NOMENCLATURE

CNN	Convolutional Neural Network
VAE	Variational Autoencoder
ELBO	Evidence Lower Bound
RNN	Recurrent Neural Network
HM-RNN	Hierarchical Multiscale Recurrent Neural Network
GAN	Generative Adversarial Network
HVAE	Hierarchical Variational Autoencoder
SSIM	Structural Similarity
PSNR	Peak Signal-to-Noise Ratio
LPIPS	Learned Perceptual Image Patch Similarity
SVG-LP	Stochastic Video Generation - Learned Prior

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

The prediction and anticipation of future events is a key component of intelligent decision-making systems. Biological agents can perform complex visual tasks, typically without requiring external supervision in the form of millions of labelled examples. They use perceptual systems for obtaining sensory information that enables them to act and accomplish their goals [1] [2]. The idea of video prediction has biological roots, and draws inspiration from the predictive coding paradigm [3] borrowed from the cognitive neuroscience field [4]. If an agent can learn to predict the future, it can take anticipatory actions, plan through its predictions, and use prediction as a proxy for representation learning. For example, robots can quickly learn manipulation skills when predicting the consequences of physical interactions. Also, an autonomous car can slow down or brake when predicting that a person is walking across the driving lane.

The task of video prediction is a generative modeling problem that uses self-supervision to predict future frames from raw video data. Videos provide complex transformations and motion information temporally, which means that computational models of perception based on egomotion (i.e. self motion) can be formulated for learning useful visual representations. Although training such models does not need annotated data, the models need to capture the complex dynamics of real-world phenomena to generate coherent sequences. Uncertainty is a key difficulty associated with prediction, as many plausible outcomes might occur from a sequence of observations. The future is by nature multimodal, and the assumption that all possible outcomes are reflected in the input data make prediction under uncertainty extremely challenging.

Figure 1.1 shows video predictions on the Moving MNIST dataset using our algorithm for a time horizon of 1000 time steps using 36 context frames as input. Our main contributions in this work are:

- We introduce a simple hierarchical video prediction model that leverages different transition speeds of latent variables per level to learn long-term dependencies in video.

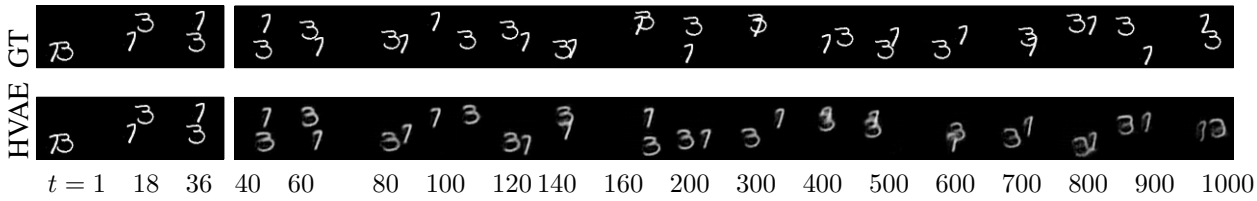

Figure 1.1: **Example results** We show example predictions of a video sequence using our method. The input sequence consists of two digits, 3 and 7 moving in an enclosed space and bouncing off the edges. Our method is able to preserve the structural integrity of the digits throughout the prediction horizon, and is able to do so by learning useful representations of the data rather than a distribution over the pixels.

- We demonstrate that our method improves over previous work on accurate long-term predictions, and also support this claim by an empirical evaluation.

2. VIDEO PREDICTION

2.1 Background

Predicting video frames from past frames through classical methods and deep learning has been an important aspect of video compression for long [5] [6]. However, a lot of progress has been made in video prediction with the advent of deep image generation models [7]. A simple approach to the problem has been to use temporally autoregressive models which predict one image after another, based on previously observed or generated images [8] [9]. However, these can be computationally expensive as they operate in the high-dimensional image space and at the frame rate of the dataset. Initially, the problem has been tackled by a deterministic model [10] [11]. Variational Autoencoders were later adopted to model the stochasticity of future visual observations [12] [13] [14]. The issue of blurry predictions that arise from using a trajectory-based latent variable model to model the stochasticity of the real world has then been addressed by two lines of orthogonal work- VAE-GANs [15] and timestep-based latent variable models [9]. While these models address the issue of blurriness on small-scale video datasets like the BAIR robot action dataset [16], they suffer from severe underfitting in large-scale datasets.

Latent dynamics models, which are typically trained using variational inference, predict a sequence of learned latent states forward that is then decoded into the video, without feeding generated frames back into the model [17]. Hierarchical latent variable models are used to better represent complex data and learn multiple levels of features. Some examples include Ladder VAE [18], VLAE [19], NVAE [20], and very deep VAEs which are used for generating static images [21]. Castrejon et al. [22] apply dense connections to hierarchical VAEs to address the optimization challenge of fitting hierarchical variational video prediction models, but are unable to scale up its hierarchical VAE. Deep video prediction models which use hierarchical latents and can learn to separate high-level details, such as textures, from low-level details, have also been used. However, all these models fail to predict very far into the future and are mainly focused on short-term video

prediction of under 100 frames.

Latent dynamics models that use temporal abstraction predict learned features at a slower frequency than the input sequence. Learning spatio-temporal structure hierarchically in sequential data is critical for an intelligent agent exploring an environment as it can enable efficient option-learning and jumpy future imagination, which are essential to solve the sample efficiency problem. Variational Temporal Abstraction(VTA) models videos using two abstraction levels, where the abstraction level with faster transitions decides when the slow states should tick [23]. In this work, we use a temporally abstract latent dynamics model to learn long-term dependencies in videos.

2.2 Temporal Information in Videos

Videos provide complex transformations and motion patterns in the time dimension, which is not the case in static images. For instance, if we look at the big picture, consecutive frames appear to be visually different due to occlusions or changes in the lighting conditions, but semantically coherent. However, at a fine granularity, a small patch at the same spatial location across consecutive time steps appears to have a wide range of visually similar local deformations due to temporal coherence. This information can be used by predictive models to extract spatio-temporal correlations depicting the dynamics in a video sequence. For instance, egomotion, i.e. self motion is shown to be a useful source of intrinsic supervision for visual feature learning in mobile agents [24].

2.3 Stochasticity of Observations

A model that can accurately predict future observations of complex sensory modalities such as vision must inherently learn a representation of the complex dynamics of real-world objects and people, and subsequently such a representation can be used to solve a variety of visual perception tasks, like object tracking and action recognition [25]. However, there are a number of spatio-temporal factors of variation that make up the dynamics of how video frames change through time.

3. GENERATIVE MODELING

Generative modeling is a broad area of machine learning which deals with models of distributions $p(X)$, defined over datapoints X in some potentially high-dimensional space $\mathcal{X}$. Images are high-dimensional sources of data where each image has a number of dimensions (pixels), and the generative model's goal is to capture the dependencies between the pixels, e.g. that nearby pixels have similar color, and are organized into objects. In the context of video prediction, generative models become useful to deal with the multimodal nature of the problem, where the space of possibilities diverges beyond a few frames. Methods that use deterministic models and loss functions such as mean-squared error (MSE) are unequipped to handle this uncertainty, and average together possible futures, resulting in blurry predictions.

In effect, the goal in generative model training is to learn an unknown or intractable probability distribution from a typically small number of independent and identically distributed samples. We can formalize this setup by saying that we get examples X distributed according to some unknown distribution $p_{data}(X)$, and our goal is to learn a model p which we can sample from, such that p is as similar as possible to p_{data}.

3.1 Latent variable models

The task of training generative models becomes harder the more complicated the dependencies between the dimensions. For example, let us consider the problem of generating handwritten characters $0-9$. If the left half of the character contains the left half of a 5, then the right half cannot contain the left half of a 0, or the character will not look like a real digit. If the model develops an intuition to make a decision as to which character to generate before assigning a value to any pixel, it would be beneficial. This kind of decision is called a *latent variable*. In order to declare that the model is representative of the dataset, we need to ensure that for every datapoint X in the dataset, there exists a setting of the latent variables which causes the model to generate something very similar to X.

By defining a joint distribution over visible and latent variables, the corresponding distribution of the observed variables is then obtained by marginalization. Formally, say we have a vector of latent variables z in a space $\mathcal{Z}$ which can be sampled according to some PDF $p(z)$ defined over $\mathcal{Z}$. Then, say we have a family of deterministic functions $f(z;\theta)$, parameterized by a vector θ in some space Θ, where $f : \mathcal{Z}$ x $\Theta \mapsto \mathcal{X}$. f is deterministic, but if z is random and θ is fixed, then $f(z;\theta)$ is a random variable in the space $\mathcal{X}$. We wish to optimize θ such that we can sample z from $P(z)$ and, with high probability, $f(z;\theta)$ will be similar to the X's in our dataset. The aim is to maximize the probability of each X in the training set under the entire generative process, according to:

$$p(X) = \int p(X|z;\theta)p(z)dz. \tag{3.1}$$

Here, $f(z;\theta)$ is replaced by the distribution $p(X|z;\theta)$, making the dependence of X on z explicit using the law of total probability.

3.2 Variational Autoencoders

A variational autoencoder provides a probabilistic approach to describe an observation in latent space. The input data is converted into an encoding vector where each dimension represents some learned attribute about the data. VAEs approximately maximize Equation 3.1, according to the model shown in Figure 3.1.

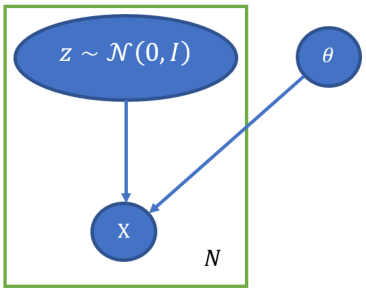

Figure 3.1: Standard VAE model represented as graphical model.

To solve Equation 3.1, VAEs answer two questions: how to define the latent variables z, and how to deal with the integral over z. Taking the handwritten digits example, the model needs to make complicated 'latent' decisions before drawing the digit, like choosing the angle of the digit or the stroke width. The key idea behind the variational autoencoder is to attempt to sample values of z that are likely to have produced X, and compute $p(X)$ just from those. The marginal density of the observations $p(X)$ is also called the *evidence*. This means that a new function $q(z|X)$ is needed which can take a value of X and output a distribution over z values that are likely to produce X.

The Kullback-Leibler divergence (KL divergence or $\mathcal{D}$) between $p(z|X)$ and $q(z)$, for some arbitrary q (which may or may not depend on X) is given by:

$$\mathcal{D}\left[q(z)\|p(z|X)\right] = \mathbb{E}_{z\sim q}\left[\log q(z) - \log p(z|X)\right]. \tag{3.2}$$

By applying Bayes rule to $p(z|X)$:

$$\mathcal{D}\left[q(z)\|p(z|X)\right] = \mathbb{E}_{z\sim q}\left[\log q(z) - \log p(X|z) - \log p(z)\right] + \log p(X). \tag{3.3}$$

Rearranging and expressing a part of $E_{z\sim q}$ as a KL-divergence term yields:

$$\log p(X) - \mathcal{D}\left[q(z)\|p(z|X)\right] = \mathbb{E}_{z\sim q}\left[\log p(X|z)\right] - \mathcal{D}\left[q(z)\|p(z)\right]. \tag{3.4}$$

Since we're interested in inferring $p(X)$, it makes sense to construct a q which depends on X and one which makes $\mathcal{D}\left[q(z)\|p(z|X)\right]$ small:

$$\log p(X) - \mathcal{D}\left[q(z|X)\|p(z|X)\right] = \mathbb{E}_{z\sim q}\left[\log p(X|z)\right] - \mathcal{D}\left[q(z|X)\|p(z)\right]. \tag{3.5}$$

Equation 3.5 serves as the core of the variational autoencoder. The left-hand side of the equation has the quantity we want to maximize. The right-hand side is something that we can optimize via stochastic gradient descent given the right choice of q. This framework is in a form similar to that

of an encoder, as q is "encoding" X into z, and p is "decoding" it to reconstruct X. The objective is to maximize $\log p(X)$ while simultaneously minimizing $\mathcal{D}\left[q(z|X)\|p(z|X)\right]$. The intractable $p(z|X)$ is made tractable by using $q(z|X)$.

3.2.1 The evidence lower bound

The right-hand side of Equation 3.5 is called the evidence lower bound (ELBO) and is given by

$$\text{ELBO}(q) = \mathbb{E}_{z\sim q}\left[\log p(X|z)\right] - \mathcal{D}\left[q(z|X)\|p(z)\right]. \tag{3.6}$$

The property of the ELBO is that it lower-bounds the (log) evidence, $\log p(X) \geq \text{ELBO}(q)$ for any $q(z)$ and hence the name. To see this, from Equations (3.5) and (3.8), we can express the evidence as

$$\log p(X) = \mathcal{D}\left[q(z|X)\|p(z|X)\right] + \text{ELBO}(q).$$

The bound then follows from the fact that the KL term $\mathcal{D}(\cdot) \geq 0$.

3.3 Recurrent State-Space Model

The recurrent state space model is a latent dynamics model that is used by PlaNet [26], a model-based agent that learns the environment dynamics from pixels and chooses actions through online planning in a compact latent space. The model explains the video sequence $x_{1:T}$ using a latent sequence of compact states $s_{1:T}$ and is autoregressive in latent space but not in image space, allowing for future predictions without generating images along the way,

$$p\left(x_{1:T}, s_{1:T}\right) = \prod_{t=1}^{T} p\left(x_t|s_t\right) p\left(s_t|s_{t-1}\right). \tag{3.7}$$

Given a training sequence of context frames, the model first individually embeds the frames using a CNN. A recurrent network with deterministic and stochastic components then summarizes the image embeddings. The stochastic component helps with generating multiple features, while the deterministic component helps remember information over multiple timesteps. The states are

then decoded using another CNN to provide a training signal,

$$\begin{aligned}
&\text{Encoder:} && e_t = \text{enc}(x_t) \\
&\text{Posterior transition } q_t: && q\left(s_t|s_{t-1}, e_t\right) \\
&\text{Prior transition } p_t: && p(s_t|s_{t-1}) \\
&\text{Decoder:} && p(x_t|s_t).
\end{aligned}$$

As the likelihood of the training data under the model cannot be computed in closed form, the evidence lower bound (ELBO) is used as the training objective. The ELBO contains a reconstruction term and a regularization term,

$$\max_{q,p} \sum_{t=1}^{T} E_{q_t}\left[\ln p\left(x_t|s_t\right)\right] - \sum_{t=1}^{T} E_{q_{t-1}}\left[\mathcal{D}\left[q_t||p_t\right]\right]. \tag{3.8}$$

All components jointly optimize Equation 3.8 using stochastic backpropagation, a modified form of gradient backpropagation that allows for the joint optimization of the parameters of both the generative and inference models [27]. Both the first and second term are stochastic with respect to the latent state random variable. The optimization uses a reparameterization trick which converts the representation of the latent state random variable into a stochastic and deterministic part, in order to provide a clear path for the backpropagation algorithm [28].

3.3.1 Variational bound

The variational bound for latent dynamics models $p(x_{1:T}, s_{1:T}) \doteq \Pi_{t=1}^{T}\, p\left(x_t \mid s_t\right) p\left(s_t \mid s_{t-1}\right)$ and a variational posterior $q\left(s_{1:T} \mid x_{1:T}\right) = \Pi_{t=1}^{T}\, q\left(s_t \mid s_{t-1}, x_t\right)$, follows from importance weight-

ing and Jensen's inequality as below,

$$
\begin{aligned}
\ln p\left(x_{1:T}\right) &\doteq \ln \mathbb{E}_{p(s_{1:T})}\left[\prod_{t=1}^{T} p(x_t \mid s_t)\right] \\
&= \ln \mathbb{E}_{q(s_{1:T} \mid x_{1:T})}\left[\prod_{t=1}^{T} p\left(x_t \mid s_t\right) p\left(s_t \mid s_{t-1}\right) / q\left(s_t \mid x_t, s_{t-1}\right)\right] \\
&\geq \mathbb{E}_{q(s_{1:T} \mid x_{1:T})}\left[\sum_{t=1}^{T} \ln p(x_t \mid s_t) + \ln p(s_t \mid s_{t-1}) - \ln q\left(s_t \mid x_t, s_{t-1}\right)\right] \\
&= \sum_{t=1}^{T}\left(\mathbb{E}_{q_t}\left[\ln p(x_t \mid s_t)\right] - \mathbb{E}_{q_{t-1}}\left[\mathcal{D}\left[q_t \| p_t\right]\right]\right)
\end{aligned}
\tag{3.9}
$$

3.4 Multiscale Recurrent Neural Networks

Multiscale RNNs have been useful for modeling hierarchical and temporal representation by grouping hidden units into multiple modules of different timescales. The key idea lies in updating the weights of neurons belonging to different layers of the stack, corresponding to the layers of hierarchies in the data. This method is able to efficiently deliver long-term dependencies with fewer updates at the higher layers, thereby mitigating the vanishing gradient problem. Implementations of these models include approaches like setting timescales as hyperparameters [29] and a model which can learn the hierarchical multiscale structure from temporal data without explicit boundary information [30].

3.5 Generative Adversarial Nets

Generative adversarial networks [7] are a framework similar to a minimax two-player game to estimate generative models via an adversarial process, in which we simultaneously train two models: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G. GANs enable machine learning to work with multi-modal outputs, making them very useful for tasks like image generation and video generation.

The basic idea of a GAN is to set up a game between two players. The generator creates

samples that are intended to come from the same distribution as the training data p_{data}. The discriminator, on the other hand, examines samples to determine whether they are real or fake. Let X be a datapoint representing an image. $D(X)$ is the discriminator network which outputs the probability that X came from the training data rather than the generator and can be thought of as a traditional binary classifier. The generator function is represented by $G(z)$ which maps the latent variable z to the high-dimensional space of the input data. The goal of G is to estimate a distribution p_g from which it can generate fake samples that resemble images from the actual training data distribution p_{data}. The value function is given by

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}\left[\log D(x)\right] + \mathbb{E}_{z\sim p_z(z)}\left[\log\left(1 - D(G(z))\right)\right]. \tag{3.10}$$

Although GANs overcome the limitations of pixel-wise losses, they are notoriously susceptible to "mode collapse", where latent random variables are often ignored by the model. This makes them difficult to apply to the generation of diverse and plausible futures, conditioned on context frames, making them unsuitable to solve the long-horizon prediction problem.

4. PRACTICAL IMPLEMENTATION AND RESULTS

In this section, we discuss our algorithm and its performance on a few standard datasets.

4.1 Hierarchical Variational Autoencoder

We propose the Hierarchical Variational Autoencoder by using ideas from latent dynamics models like the one used by PlaNet and multiscale recurrent neural networks [30] to learn long-term correlations of videos. Our model predicts ahead on multiple timescales, as shown in Figure 4.1. HVAE consists of a hierarchy of recurrent latent variables, where each level transitions at a different frequency. The transitions slow down exponentially as we go up the hierarchy by a factor k, which we denote as the temporal abstraction factor. The latent state at time t and level l is denoted by s_t^l and the video frames by x_t. We define a set of active time steps $\mathcal{T}_l$ for each level $l \in [1, L]$ as those time instances for which the state transition generates a new latent state,

$$\mathcal{T}_l \doteq \{t \in [1, T] \mid t \mod k^{l-1} = 1\}. \tag{4.1}$$

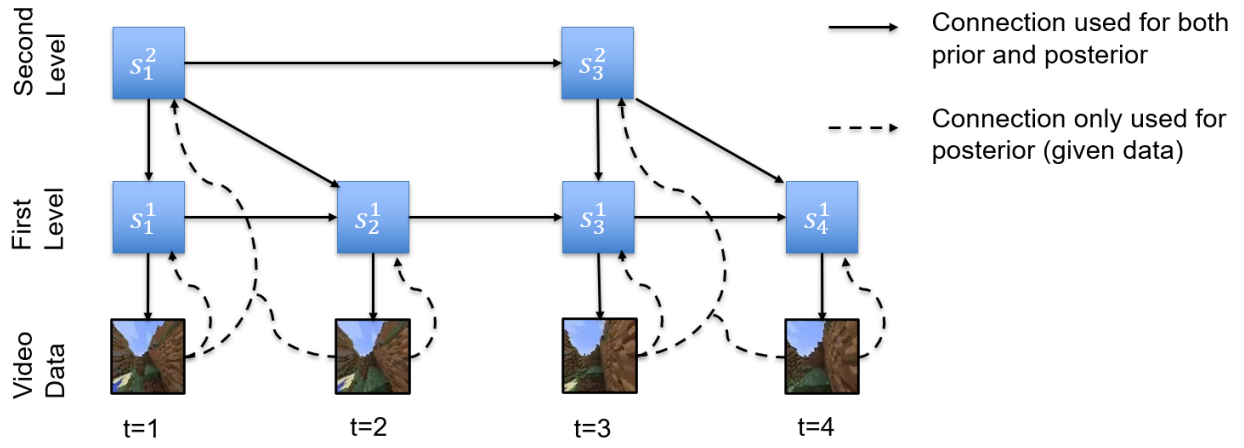

Figure 4.1: Hierarchical Variational Autoencoder.

At each level, we condition k consecutive latent states on a single latent variable in the level above. For example, in the model shown in Figure 4.1, with the temporal abstraction factor $k = 2$, $\mathcal{T}_1 = \{1, 2, 3, \ldots\}$, $\mathcal{T}_2 = \{1, 3, 5, \ldots\}$, both s_1^1 and s_2^1 are conditioned on the same s_1^2 from the previous level. Each level of the hierarchy of latent variables has a separate state variable per time step, with the previous state being updated every k^{l-1} time steps, and otherwise copying the previous state. So, $\forall t \notin \mathcal{T}_l$, the copied states are given by:

$$s_t^l \doteq s^l_{\max_\tau\{\tau \in \mathcal{T}_l | \tau \leq t\}}. \tag{4.2}$$

We factorize the joint distribution of a sequence of images and active latents at every level into two terms:

- the reconstruction terms of the images given the lowest latents, and
- state transitions at all levels that are conditioned on the previous latent and the latent above,

$$p\left(x_{1:T}, s_{1:T}^{1:L}\right) \doteq \left(\Pi_{t=1}^{T}\, p\left(x_t \mid s_t^1\right)\right)\left(\Pi_{l=1}^{L}\, \Pi_{t \in \mathcal{T}_l}\, p\left(s_t^l \mid s_{t-1}^l, s_t^{l+1}\right)\right). \tag{4.3}$$

In order to implement the distribution in Equation 4.3, we use the following components, $\forall l \in [1, L], t \in \mathcal{T}_l$,

$$\begin{aligned} \text{Encoder:} \quad & e_t^l = e\left(x_{t:t+k^{l-1}-1}\right) \\ \text{Posterior transition } q_t^l: \quad & q\left(s_t^l \mid s_{t-1}^l, s_t^{l+1}, e_t^l\right) \\ \text{Prior transition } p_t^l: \quad & p\left(s_t^l \mid s_{t-1}^l, s_t^{l+1}\right) \\ \text{Decoder:} \quad & p\left(x_t \mid s_t^1\right). \end{aligned} \tag{4.4}$$

4.1.1 Inference

In the inference stage, the embeddings are computed using a CNN and the posterior transition function is then calculated. Each active latent state at level l receives image embeddings of its

corresponding k^{l-1} observations (dashed lines in Figure 4.1). The belief q_t^l is computed as a function of the input features, the posterior sample at the previous step, and the posterior sample above (solid lines in Figure 4.1). The belief is a Gaussian distribution with a diagonal covariance matrix.

4.1.2 Generation

During the generation stage, the prior p_t^l is computed by applying the transition function from the latent state at the previous time step in the current level, as well as the state belief at the level above (solid lines in Figure 4.1). Finally, the posterior samples at the lowest level are decoded into images using a transposed CNN, which is an upsampling operation on the latent space feature map to produce the image in the higher dimensional space.

4.1.3 Training Objective

We cannot compute the likelihood of the training data under the model in closed form. Instead, we use the evidence lower bound as the training objective, as defined in Section 3.3.1. This objective optimizes a reconstruction loss at the lowest level, and a Kullback-Leibler divergence regularization term at every level summed across active time steps,

$$\max_{e,q,p} \sum_{t=1}^{T} \mathbb{E}_{q_t^1} \left[\ln p \left(x_t \mid s_t^1 \right) \right] - \sum_{l=1}^{L} \sum_{t \in \mathcal{T}_l} \mathbb{E}_{q_{t-1}^l q_t^{l+1}} \left[\mathcal{D} \left[q_t^l \parallel p_t^l \right] \right]. \tag{4.5}$$

The function of the KL regularization terms is to restrict the amount of information about the images that enters via the encoder. This prioritizes information available from the previous and above latent levels, and extracts information from the input image only to the extent necessary. As the number of active time steps decreases higher up in the hierarchy, the number of KL terms per level decreases. This makes it easier for the model to store slowly changing information in the higher levels than to pay a KL penalty to repeatedly extract information from images in the lower level.

4.1.4 State Components

We split the state s_t^l into stochastic (z_t^l) and deterministic (h_t^l) parts, as shown in Figure 4.2. The deterministic state is computed using the top-down and temporal context, which then conditions the stochastic state at that level. A Gated Recurrent Unit (GRU) per level is used to update the deterministic variable at every active step.

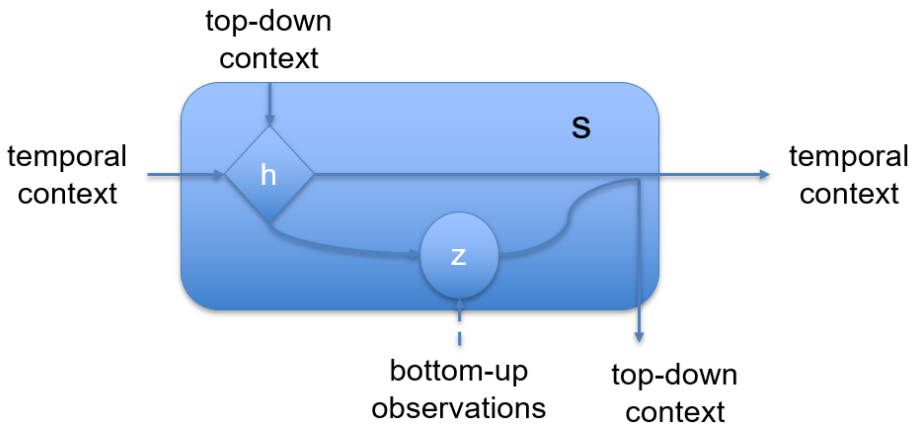

Figure 4.2: State components of HVAE.

The motivation for including a deterministic component is due to the fact that purely stochastic transitions make it difficult for the transition model to reliably remember information for multiple time steps. In theory, although the model could learn to set the variance to zero for some state components in the purely stochastic transition case, the optimization procedure may not find this solution.

4.2 Datasets

We choose two different video datasets for the benchmark. The MineRL Navigate dataset was crowdsourced for reinforcement learning applications by Guss et al. [31]. We process this data to create a long-horizon video prediction dataset by combining the *Navigate* and *Navigate Extreme* tasks, splitting them into non-overlapping sequences of length 500, and splitting them into training

and test sets. The Moving MNIST dataset contains $2\text{x}10^6$ frames where two digits move with velocities sampled in the range of 2 to 6 pixels per frame, and bounce within the edges of the image.

4.3 Quantitative Evaluation

We compare HVAE to two well-established video prediction models, and an ablation of our method where the states at all levels are updated at the fastest scale with a temporal abstraction factor of 1, which we denote as HVAE-Fast. PlaNet uses the recurrent state-space model, which is commonly used as a world model in reinforcement learning [26]. It predicts forward using a sequence of compact latents without generating images along the way. SVG-LP (or SVG for short) has been shown to generate predictions that are both varied and sharp on visually complex datasets [9]. It autoregressively feeds generated images back into the model, while also using a latent variable at every time step.

4.4 Training details

We train all models on all datasets for 300 epochs on training sequences of 100 frames of size 64 x 64 pixels. For the baselines, we tune the learning rate in the range $[10^{-4}, 10^{-3}]$. We use a temporal abstraction factor of 6 for HVAE.

4.4.1 Model Architecture

We use the discriminator and generator of DCGAN [32] for the convolutional frame encoders and decoders, respectively. To obtain input embeddings e_t^l at a particular level, k^{l-1} input embeddings are pre-processed using a feed-forward network and then summed to obtain a single embedding. We do not use any skip connections between the encoder and the decoder, which would bypass the latent states. The posterior and prior transition models are implemented using a recurrent neural network.

4.4.2 Hyperparameters

We keep the feature dimension of the encoder output at each level of HVAE as $|e_t^l| = 1024$, that of the stochastic states as $\left|p_t^l\right| = \left|q_t^l\right| = 20$ and that of the deterministic states as $\left|h_t^l\right| = 200$. The number of hidden layers in the RNN cell is set to 200. We train using the Adam optimizer [33] with a learning rate of 3 x 10^{-4} and $\epsilon = 10^{-4}$ using a batch size of 30 sequences with 100 frames each.

We use 36 context frames for video predictions, which is the minimum number of frames required to transition at least once in the highest level of the hierarchy. With 3 levels in the hierarchy and a temporal abstraction factor of 6, each latent state at the highest level corresponds to 36 images in the sequence, and thus its encoder network expects 36 images as input. This follows from Equation 4.4 where each active latent state at level l receives image embeddings of its corresponding k^{l-1} observations, where k is the temporal abstraction factor.

4.5 Evaluation Metrics

We use three different metrics to evaluate the open-loop predictions:

- Structural Similarity Index (SSIM): SSIM is a method for measuring the similarity between two images [34]. The SSIM index can be viewed as a quality measure of one of the images being compared, provided the other image is regarded as of perfect quality. The higher this quantity, the better the quality of prediction.

- Peak Signal-to-Noise Ratio (PSNR): PSNR is used as a quality measurement between the ground truth and test image. The higher the PSNR, the better the quality of the predicted image.

- Learned Perceptual Image Patch Similarity (LPIPS): LPIPS is a metric that evaluates the distance between image patches. A higher value indicates that the images are further or more different. A lower value indicates that the images are similar, and the prediction quality is good.

4.6 Results

We evaluate the 4 models on two datasets and three metrics. HVAE outperforms the existing methods, which we attribute to its hierarchical latents and temporal abstraction.

	MineRL Navigate			**Moving MNIST**		
	SSIM	PSNR	LPIPS	SSIM	PSNR	LPIPS
HVAE	**0.64**	**22.91**	**0.31**	**0.68**	**13.04**	**0.26**
PlaNet	**0.66**	**23.48**	**0.31**	0.64	**12.79**	0.28
SVG	0.56	17.22	**0.32**	0.64	11.73	0.31
HVAE-Fast	0.56	20.39	0.36	**0.66**	**12.93**	**0.26**

Table 4.1: Quantitative comparison of HVAE with other state-of-the-art video prediction models. The scores are averaged across frames of all evaluation sequences of a dataset. The best performing model and those within 5% of its performance are highlighted in bold.

4.6.1 Moving MNIST

We run our prediction algorithm on the Moving MNIST dataset as shown in Figure 4.3. We use the first 36 frames as context and generate open-loop prediction of 1000 frames. Here are some observations from running the models on this dataset:

- We observe that HVAE is able to remember digit identities across all 1000 frames of the prediction horizon, whereas the other models forget the digit identities at around 300 time steps.
- PlaNet outperforms SVG, but starts to forget digit identities much sooner than HVAE.
- HVAE predicts accurate positions of digits until around 100 steps, and predicts a plausible sequence thereafter.
- PlaNet also predicts correct location of digits for at least as long as HVAE, whereas SVG starts to lose track much sooner.

- The prediction models that predict purely in latent space are slightly blurry compared to those generated by SVG.

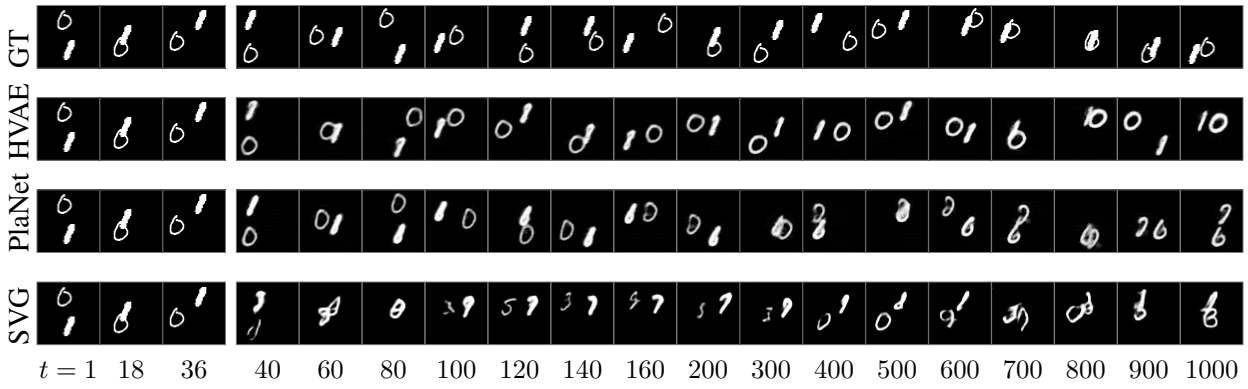

Figure 4.3: Long-horizon open-loop video predictions on Moving MNIST.

Figure 4.4 shows a grid of predicted video frames for HVAE for 1000 time steps. The predictions are able to maintain the structural integrity of the two digits 5 and 8 throughout the prediction horizon.

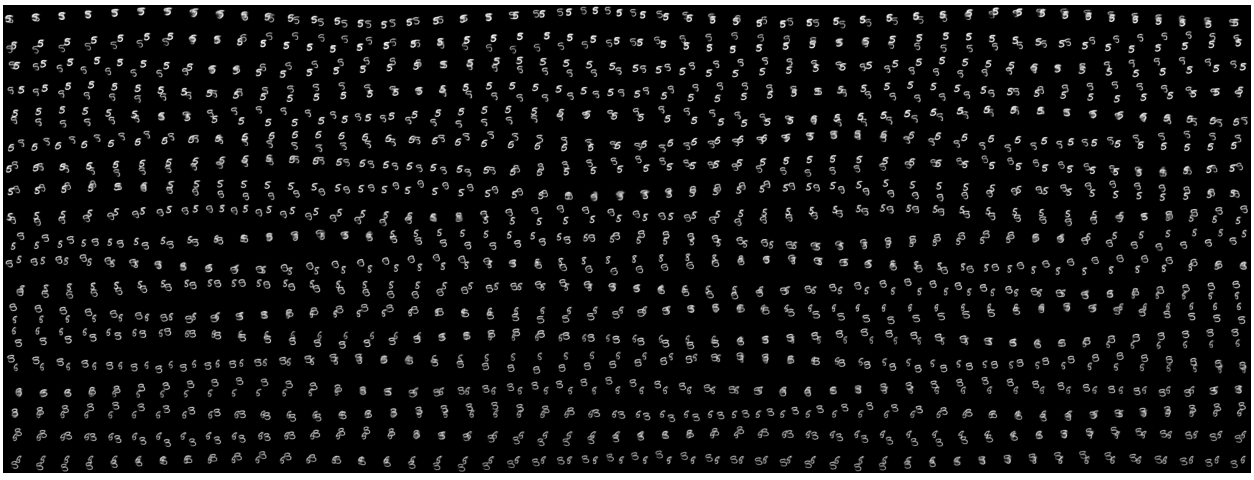

Figure 4.4: Open-loop video prediction using HVAE for 1000 frames shown as a grid.

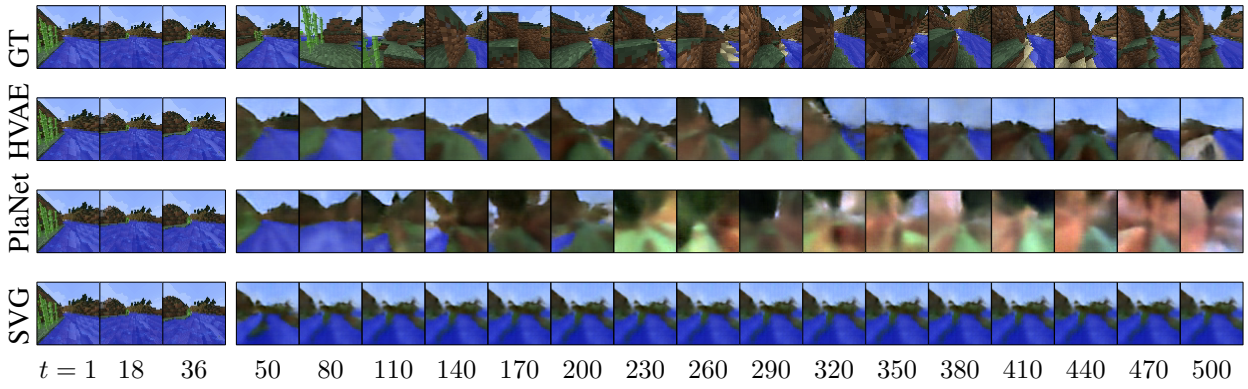

Figure 4.5: Long-horizon video predictions on the MineRL dataset.

4.6.2 MineRL Navigate

Video predictions for sequences of 500 frames are shown in Figure 4.5. Given a canyon on the horizon and a body of water as context, HVAE correctly predicts that the player will enter the canyon (as the player typically navigates straight ahead in the dataset). Here are some other observations from the predictions:

- HVAE predicts diverse variations in the terrain, such as grass, rocks, trees, and a body of water.
- PlaNet generates plausible images but fails to capture long-term dependencies, such as consistent movement towards the island.
- SVG does not have variations after the first few frames.
- The predictions are blurry as the model capacity is small.

4.6.3 Temporal Abstraction Factor

We compare the quality of open-loop video predictions on Moving MNIST for HVAE with temporal abstraction factors 2, 4, and 6 - all with equal number of model parameters, as shown in Figure 4.6. We observe that increasing the temporal abstraction factor directly increases the duration for which the predicted frames are accurate.

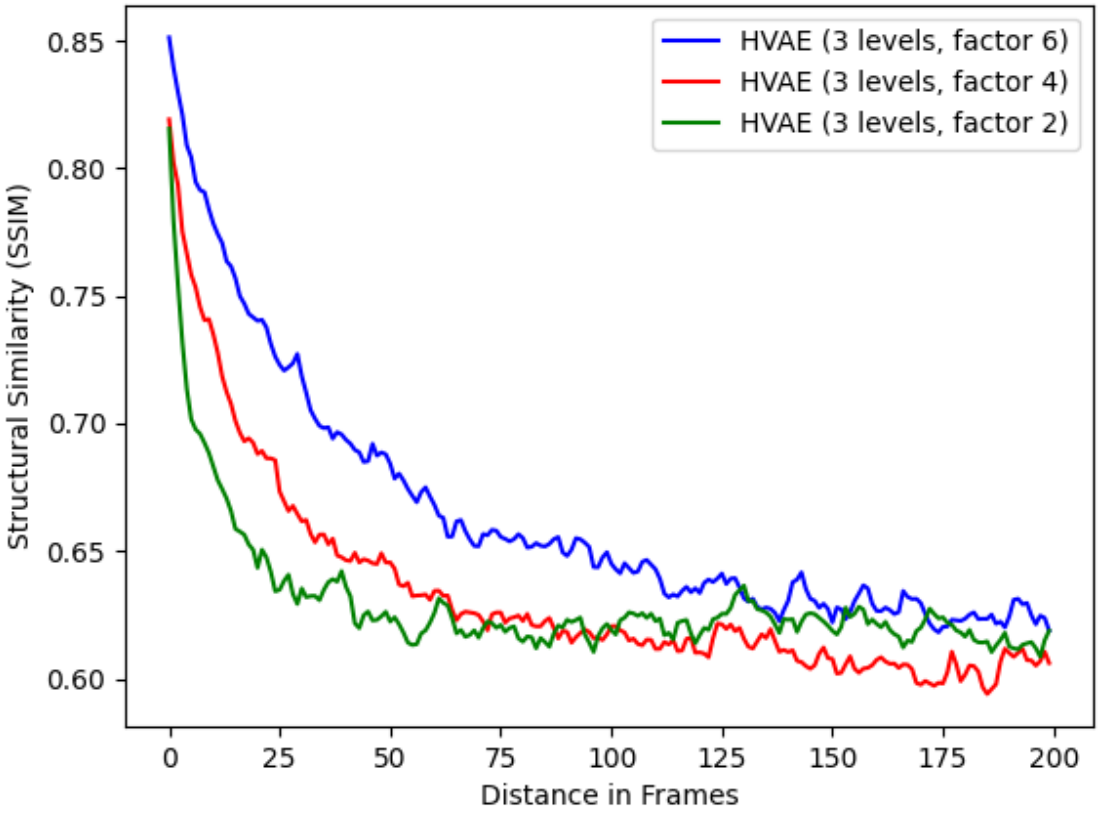

Figure 4.6: 3 versions of HVAE with temporal abstraction factors of 2, 4, and 6.

4.7 Discussion

In this work, we introduce the Hierarchical Variational Autoencoder (HVAE) that uses a temporally abstract hierarchical structure of latent variables for long-term video prediction. We demonstrate the empirical performance of the model on two diverse video prediction datasets with up to 1000 frames, and show that HVAE outperforms state-of-the-art video prediction models from the literature.

4.7.1 Challenges

We point out some of the challenges of our work here that could be promising directions for future work:

- The typical video prediction metrics are not ideal at capturing the quality of video predictions, especially for long horizons. To this end, we have experimented with evaluating the best out of 30 samples for each evaluation video without observing any significant differences in the results. Using datasets where attributes of the scene are available would allow evaluation of long-horizon predictions by how well those attributes can be extracted from the underlying representations using a separately trained network.

- We used relatively small convolutional neural networks for the encoder and decoder. A larger architecture could increase the quality of generated images, where the model is able to better predict both high frequency details and long-term dependencies in the data.

REFERENCES

[1] J. J. Gibson, *The Ecological Approach to Visual Perception*. Houghton Mifflin, 1979.

[2] J. E. Cutting, *Perception with an eye for motion*. MIT Press, 1986.

[3] R. P. N. Rao and D. H. Ballard, "Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects," *Nature Neuroscience*, vol. 2, no. 1, 1999.

[4] D. Mumford, "On the computational architecture of the neocortex," *Biological Cybernetics*, vol. 66, no. 3, 1992.

[5] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h.264/avc video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, 2003.

[6] H. Choi and I. V. Bajić, "Deep frame prediction for video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 7, pp. 1843–1855, 2020.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems* (Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, eds.), vol. 27, Curran Associates, Inc., 2014.

[8] M. Babaeizadeh, C. Finn, D. Erhan, R. H. Campbell, and S. Levine, "Stochastic variational video prediction," *CoRR*, vol. abs/1710.11252, 2017.

[9] E. Denton and R. Fergus, "Stochastic video generation with a learned prior," *CoRR*, vol. abs/1802.07687, 2018.

[10] J. Walker, A. Gupta, and M. Hebert, "Dense optical flow prediction from a static image," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2443–2451, 2015.

[11] C. Finn, I. Goodfellow, and S. Levine, "Unsupervised learning for physical interaction through video prediction," in *Advances in neural information processing systems*, pp. 64–72, 2016.

[12] R. Shu, J. Brofos, F. Zhang, H. H. Bui, M. Ghavamzadeh, and M. Kochenderfer, "Stochastic video prediction with conditional density estimation," in *ECCV Workshop on Action and Anticipation for Visual Learning*, vol. 2, pp. 64–72, 2016.

[13] N. Wichers, R. Villegas, D. Erhan, and H. Lee, "Hierarchical long-term video prediction without supervision," in *International Conference on Machine Learning (ICML)*, 2018.

[14] J. Franceschi, E. Delasalles, M. Chen, S. Lamprier, and P. Gallinari, "Stochastic latent residual video prediction," *CoRR*, vol. abs/2002.09219, 2020.

[15] A. X. Lee, R. Zhang, F. Ebert, P. Abbeel, C. Finn, and S. Levine, "Stochastic adversarial video prediction," *CoRR*, vol. abs/1804.01523, 2018.

[16] F. Ebert, S. Dasari, A. X. Lee, S. Levine, and C. Finn, "Robustness via retrying: Closed-loop robotic manipulation with self-supervised learning," in *Conference on Robot Learning (CoRL)*, 2018.

[17] R. E. Kalman, "A new approach to linear filtering and prediction problems.," *Journal of basic Engineering*, no. 82(1), pp. 35–45, 1960.

[18] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther, "Ladder variational autoencoders," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, p. 3745–3753, Curran Associates Inc., 2016.

[19] S. Zhao, J. Song, and S. Ermon, "Learning hierarchical features from generative models," *CoRR*, vol. abs/1702.08396, 2017.

[20] A. Vahdat and J. Kautz, "NVAE: A deep hierarchical variational autoencoder," in *Neural Information Processing Systems (NeurIPS)*, 2020.

[21] R. Child, "Very deep vaes generalize autoregressive models and can outperform them on images," *CoRR*, vol. abs/2011.10650, 2020.

[22] L. Castrejon, N. Ballas, and A. Courville, "Improved conditional vrnns for video prediction," in *The IEEE International Conference on Computer Vision (ICCV)*, 2019.

[23] T. Kim, S. Ahn, and Y. Bengio, "Variational temporal abstraction," *CoRR*, vol. abs/1910.00775, 2019.

[24] P. Agrawal, J. Carreira, and J. Malik, "Learning to see by moving," in *ICCV*, 2015.

[25] E. Denton and V. Birodkar, "Unsupervised learning of disentangled representations from video," *CoRR*, vol. abs/1705.10915, 2017.

[26] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels.," in *International Conference on Machine Learning*, pp. 2555–2565, 2019b.

[27] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, p. II–1278–II–1286, JMLR.org, 2014.

[28] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2014.

[29] J. Koutník, K. Greff, F. J. Gomez, and J. Schmidhuber, "A clockwork RNN," *CoRR*, vol. abs/1402.3511, 2014.

[30] J. Chung, S. Ahn, and Y. Bengio, "Hierarchical multiscale recurrent neural networks," *CoRR*, vol. abs/1609.01704, 2016.

[31] W. H. Guss, B. Houghton, N. Topin, P. Wang, C. Codel, M. Veloso, and R. Salakhutdinov, "MineRL: A large-scale dataset of Minecraft demonstrations," *Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019.

[32] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *4th International Conference on Learning Representations, ICLR*, 2016.

[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, 2014.

[34] H. R. S. Z. Wang, A. C. Bovik and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.