

VideoGPT: Video Generation using VQ-VAE and Transformers

Wilson Yan^{* 1} Yunzhi Zhang^{* 1} Pieter Abbeel¹ Aravind Srinivas¹

Abstract

We present VideoGPT: a conceptually simple architecture for scaling likelihood based generative modeling to natural videos. VideoGPT uses VQ-VAE that learns downsampled discrete latent representations of a raw video by employing 3D convolutions and axial self-attention. A simple GPT-like architecture is then used to autoregressively model the discrete latents using spatio-temporal position encodings. Despite the simplicity in formulation and ease of training, our architecture is able to generate samples competitive with state-of-the-art GAN models for video generation on the BAIR Robot dataset, and generate high fidelity natural images from UCF-101 and Tumbler GIF Dataset (TGIF). We hope our proposed architecture serves as a reproducible reference for a minimalistic implementation of transformer based video generation models. Samples and code are available at <https://wilson1yan.github.io/videogpt/index.html>.

1. Introduction

Deep generative models of multiple types (Kingma & Welling, 2013; Goodfellow et al., 2014; van den Oord et al., 2016b; Dinh et al., 2016) have seen incredible progress in the last few years on multiple modalities including natural images (van den Oord et al., 2016c; Zhang et al., 2019; Brock et al., 2018; Kingma & Dhariwal, 2018; Ho et al., 2019a; Karras et al., 2017; 2019; Van Den Oord et al., 2017; Razavi et al., 2019; Vahdat & Kautz, 2020; Ho et al., 2020; Chen et al., 2020; Ramesh et al., 2021), audio waveforms conditioned on language features (van den Oord et al., 2016a; Oord et al., 2017; Prenger et al., 2019; Bińkowski et al., 2019), natural language in the form of text (Radford et al., 2019; Brown et al., 2020), and music generation (Dhariwal et al., 2020). These results have been made possible thanks to fundamental advances in deep learning

^{*}Equal contribution ¹University of California, Berkeley. Correspondence to: Wilson Yan, Aravind Srinivas <wilson1.yan@berkeley.edu, aravind.srinivas@berkeley.edu>.

architectures (He et al., 2015; van den Oord et al., 2016b;c; Vaswani et al., 2017; Zhang et al., 2019; Menick & Kalchbrenner, 2018) as well as the availability of compute resources (Jouppi et al., 2017; Amodei & Hernandez, 2018) that are more powerful and plentiful than a few years ago.

While there have certainly been impressive efforts to model videos (Vondrick et al., 2016; Kalchbrenner et al., 2016; Tulyakov et al., 2018; Clark et al., 2019), high-fidelity natural videos is one notable modality that has not seen the same level of progress in generative modeling as compared to images, audio, and text. This is reasonable since the complexity of natural videos requires modeling correlations across both space and time with much higher input dimensions. Video modeling is therefore a natural next challenge for current deep generative models. The complexity of the problem also demands more compute resources which can also be deemed as one important reason for the *relatively* slow progress in generative modeling of videos.

Why is it useful to build generative models of videos? Conditional and unconditional video generation implicitly addresses the problem of video prediction and forecasting. Video prediction (Srivastava et al., 2015; Finn et al., 2016; Kalchbrenner et al., 2017; Sønderby et al., 2020) can be seen as learning a generative model of future frames conditioned on the past frames. Architectures developed for video generation can be useful in forecasting applications for weather prediction (Sønderby et al., 2020), autonomous driving (for e.g., such as predicting the future in more semantic and dense abstractions like segmentation masks (Luc et al., 2017)). Finally, building generative models of the world around us is considered as one way to measure our understanding of physical common sense and predictive intelligence (Lake et al., 2015).

Multiple classes of generative models have been shown to produce strikingly good samples such as autoregressive models (van den Oord et al., 2016b;c; Parmar et al., 2018; Menick & Kalchbrenner, 2018; Radford et al., 2019; Chen et al., 2020), generative adversarial networks (GANs) (Goodfellow et al., 2014; Radford et al., 2015), variational autoencoders (VAEs) (Kingma & Welling, 2013; Kingma et al., 2016; Vahdat & Kautz, 2020; Child, 2020), Flows (Dinh et al., 2014; 2016; Kingma & Dhariwal, 2018; Ho et al., 2019a), vector quantized VAE (VQ-VAE) (Van

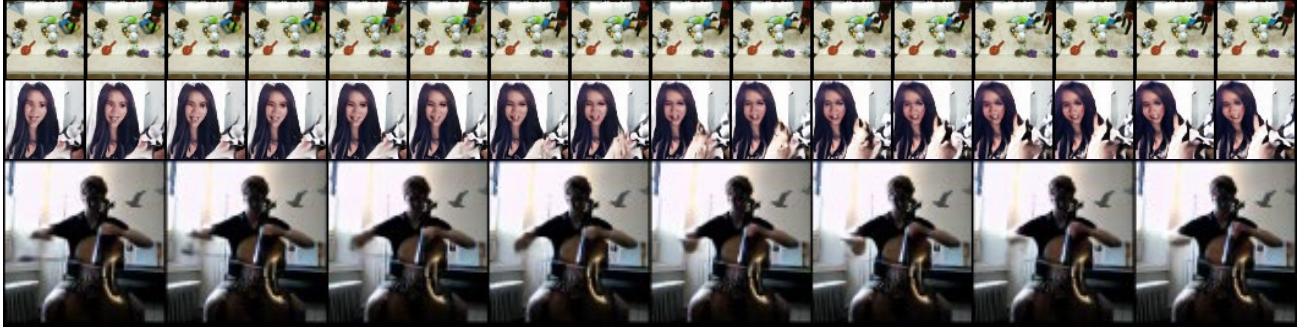


Figure 1. 64×64 and 128×128 video samples generated by VideoGPT

Den Oord et al., 2017; Razavi et al., 2019; Ramesh et al., 2021), and lately diffusion and score matching models (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020). These different generative model families have their tradeoffs across various dimensions: sampling speed, sample diversity, sample quality, optimization stability, compute requirements, ease of evaluation, and so forth. Excluding score-matching models, at a broad level, one can group these models into likelihood-based (PixelCNNs, iGPT, NVAE, VQ-VAE, Glow), and adversarial generative models (GANs). The natural question is: What is a good model class to pick for studying and scaling video generation?

First, we make a choice between likelihood-based and adversarial models. Likelihood-based models are convenient to train since the objective is well understood, easy to optimize across a range of batch sizes, and easy to evaluate. Given that videos already present a hard modeling challenge due to the nature of the data, we believe likelihood-based models present fewer difficulties in the optimization and evaluation, hence allowing us to focus on the architecture modeling¹. Next, among likelihood-based models, we pick autoregressive models simply because they have worked well on discrete data in particular, have shown greater success in terms of sample quality (Ramesh et al., 2021), and have well established training recipes and modeling architectures that take advantage of latest innovations in Transformer architectures (Vaswani et al., 2017; Child et al., 2019; Ho et al., 2019b).

Finally, among autoregressive models, we consider the following question: Is it better to perform autoregressive modeling in a downsampled latent space without spatio-temporal redundancies compared to modeling at the atomic level of all

¹It is not the focus of this paper to say likelihood models are better than GANs for video modeling. This is purely a design choice guided by our inclination to explore likelihood based generative models and non-empirically established beliefs with respect to stability of training.

pixels across space and time? Below, we present our reasons for choosing the former: Natural images and videos contain a lot of spatial and temporal redundancies and hence the reason we use image compression tools such as JPEG (Wallace, 1992) and video codecs such as MPEG (Le Gall, 1991) everyday. These redundancies can be removed by learning a denoised downsampled encoding of the high resolution inputs. For example, 4x downsampling across spatial and temporal dimensions results in 64x downsampled resolution so that the computation of powerful deep generative models is spent on these more fewer and useful bits. As shown in VQ-VAE (Van Den Oord et al., 2017), even a lossy decoder can transform the latents to generate sufficiently realistic samples. This framework has in recent times produce high quality text-to-image generation models such as DALL-E (Ramesh et al., 2021). Furthermore, modeling in the latent space downsampled across space and time instead of the pixel space improves sampling speed and compute requirements due to reduced dimensionality.

The above line of reasoning leads us to our proposed model: VideoGPT², a simple video generation architecture that is a minimal adaptation of VQ-VAE and GPT architectures for videos. VideoGPT employs 3D convolutions and transposed convolutions (Tran et al., 2015) along with axial attention (Clark et al., 2019; Ho et al., 2019b) for the autoencoder in VQ-VAE, learning a downsampled set of discrete latents from raw pixels of the video frames. These latents are then modeled using a strong autoregressive prior using a GPT-like (Radford et al., 2019; Child et al., 2019; Chen et al., 2020) architecture. The generated latents from the autoregressive prior are then decoded to videos of the original resolution using the decoder of the VQ-VAE.

²We note that Video Transformers (Weissenborn et al., 2019) also employ generative pre-training for videos using the Subscale Pixel Networks (SPN) (Menick & Kalchbrenner, 2018) architecture. Despite this, it is fair to use the GPT terminology for our model because our architecture more closely resembles the vanilla Transformer in a manner similar to iGPT (Chen et al., 2020).

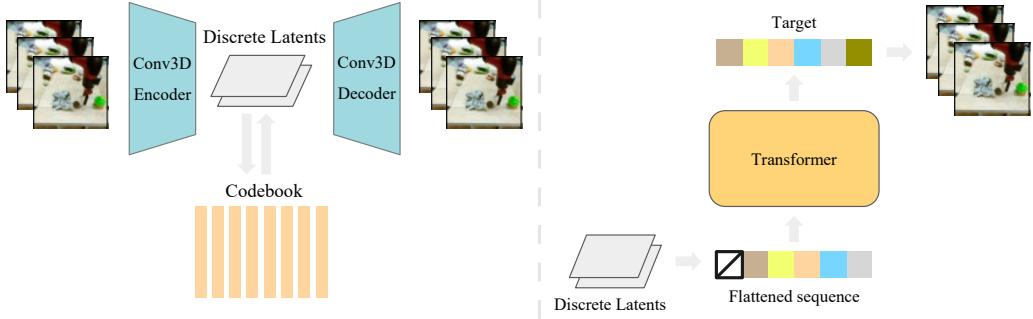


Figure 2. We break down the training pipeline into two sequential stages: training VQ-VAE (Left) and training an autoregressive transformer in the latent space (Right). The first stage is similar to the original VQ-VAE training procedure. During the second stage, VQ-VAE encodes video data to latent sequences as training data for the prior model. For inference, we first sample a latent sequence from the prior, and then use VQ-VAE to decode the latent sequence to a video sample.

Our results are highlighted below:

1. On the widely benchmarked BAIR Robot Pushing dataset (Ebert et al., 2017), VideoGPT can generate realistic samples that are competitive with existing methods such as TrIVD-GAN (Luc et al., 2020), achieving an FVD of 103 when benchmarked with real samples, and an FVD* (Razavi et al., 2019) of 94 when benchmarked with reconstructions.
2. In addition, VideoGPT is able to generate realistic samples from complex natural video datasets, such as UCF-101 and the Tumblr GIF dataset
3. We present careful ablation studies for the several architecture design choices in VideoGPT including the benefit of axial attention blocks, the size of the VQ-VAE latent space, number of codebooks, and the capacity (model size) of the autoregressive prior.
4. VideoGPT can easily be adapted for action conditional video generation. We present qualitative results on the BAIR Robot Pushing dataset and Vizdoom simulator (Kempka et al., 2016).

2. Background

2.1. VQ-VAE

The Vector Quantized Variational Autoencoder (VQ-VAE) (Van Den Oord et al., 2017) is a model that learns to compress high dimensional data points into a discretized latent space and reconstruct them. The encoder $E(x) \rightarrow h$ first encodes x into a series of latent vectors h which is then discretized by performing a nearest neighbors lookup in a codebook of embeddings $C = \{e_i\}_{i=1}^K$ of size K . The decoder $D(e) \rightarrow \hat{x}$ then learns to reconstruct x from the quantized encodings. The VQ-VAE is trained using the

following objective:

$$\mathcal{L} = \underbrace{\|x - D(e)\|_2^2}_{\mathcal{L}_{\text{recon}}} + \underbrace{\|sg[E(x)] - e\|_2^2}_{\mathcal{L}_{\text{codebook}}} + \underbrace{\beta \|sg[e] - E(x)\|_2^2}_{\mathcal{L}_{\text{commit}}}$$

where sg refers to a stop-gradient. The objective consists of a reconstruction loss $\mathcal{L}_{\text{recon}}$, a codebook loss $\mathcal{L}_{\text{codebook}}$, and a commitment loss $\mathcal{L}_{\text{commit}}$. The reconstruction loss encourages the VQ-VAE to learn good representations to accurately reconstruct data samples. The codebook loss brings codebook embeddings closer to their corresponding encoder outputs, and the commitment loss is weighted by a hyperparameter β and prevents the encoder outputs from fluctuating between different code vectors.

An alternative replacement for the codebook loss described in (Van Den Oord et al., 2017) is to use an EMA update which empirically shows faster training and convergence speed. In this paper, we use the EMA update when training the VQ-VAE.

2.2. GPT

GPT and Image-GPT (Chen et al., 2020) are a class of autoregressive transformers that have shown tremendous success in modelling discrete data such as natural language and high dimensional images. These models factorize the data distribution $p(x)$ according to $p(x) = \prod_{i=1}^d p(x_i|x_{<i})$ through masked self-attention mechanisms and are optimized through maximum likelihood. The architectures employ multi-head self-attention blocks followed by pointwise MLP feedforward blocks following the standard design from (Vaswani et al., 2017).

3. VideoGPT

Our primary contribution is VideoGPT, a new method to model complex video data in a computationally efficient

manner. An overview of our method is shown in Fig 2.

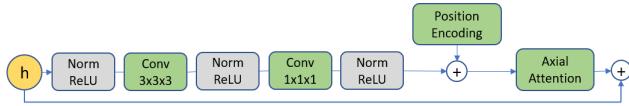


Figure 3. Architecture of the attention residual block in the VQ-VAE as a replacement for standard residual blocks.

Learning Latent Codes In order to learn a set of discrete latent codes, we first train a VQ-VAE on the video data. The encoder architecture consists of a series of 3D convolutions that downsample over space-time, followed by attention residual blocks. Each attention residual block is designed as shown in Fig 3, where we use LayerNorm (Ba et al., 2016), and axial attention layers following (Ho et al., 2019b).

The architecture for the decoder is the reverse of the encoder, with attention residual blocks followed by a series of 3D transposed convolution that upsample over space-time. The position encodings are learned spatio-temporal embeddings that are shared between all axial attention layers in the encoder and decoder.

Learning a Prior The second stage of our method is to learn a prior over the VQ-VAE latent codes from the first stage. We follow the Image-GPT architecture for prior network, except that we add dropout layers after the feed-forward and attention block layers for regularization.

Although the VQ-VAE is trained unconditionally, we can generate conditional samples by training a conditional prior. We use two types of conditioning:

- **Cross Attention:** For video frame conditioning, we first feed the conditioned frames into a 3D ResNet, and then perform cross-attention on the ResNet output representation during prior network training.
- **Conditional Norms:** Similar to conditioning methods used in GANs, we parameterize the gain and bias in the transformer Layer Normalization (Ba et al., 2016) layers as affine functions of the conditional vector. This conditioning method is used for action and class-conditioning models.

4. Experiments

In the following section, we evaluate our method and design experiments to answer the following questions:

- Can we generate high-fidelity samples from complex video datasets?
- How do different architecture design choices for VQ-VAE and prior network affect performance?

4.1. Training Details

All image data is scaled to $[-0.5, 0.5]$ before training. For VQ-VAE training, we use random restarts for embeddings, and codebook initialization by copying encoder latents as described in (Dhariwal et al., 2020). In addition, we found VQ-VAE training to be more stable (less codebook collapse) when using Normalized MSE for the reconstruction loss, where MSE loss is divided by the variance of the dataset. For all datasets except UCF-101, we train on 64×64 videos of sequence length 16. For the transformer, we train Sparse Transformers (Child et al., 2019) with local and strided attention across space-time. Exact architecture details and hyperparameters can be found in Appendix A. We achieve all results with a maximum of 8 Quadro RTX 6000 GPUs (24 GB memory).

4.2. Moving MNIST

For Moving MNIST, VQ-VAE downsamples input videos by a factor of 4 over space-time (64x total reduction), and contains two residual layers with no axial-attention. We use a codebook of 512 codes, each 64-dim embeddings. To learn the single-frame conditional prior, we train a conditional transformer with 384 hidden features, 4 heads, 8 layers, and a ResNet-18 single frame encoder. Fig 6 shows several different generated trajectories conditioned on a single frame.

Table 1. FVD on BAIR

Method ³	FVD (\downarrow)
SV2P	262.5
LVT	125.8
SAVP	116.4
DVD-GAN-FP	109.8
VideoGPT (ours)	103.3
TrIVD-GAN-FP	103.3
Video Transformer	94 \pm 2

4.3. BAIR Robot Pushing

For BAIR, VQ-VAE downsamples the inputs by a factor of 2x over each of height, width and time dimensions. The embedding in the latent space is a 256-dimensional vector, which is discretized through a codebook with 1024 codes. We use 4 axial-attention residual blocks for the VQ-VAE encoder and a prior network with a hidden size of 512 and 16 layers.

Quantitatively, Table 1³ shows FVD results on BAIR, com-

³SV2P (Babaeizadeh et al., 2017), SAVP (Lee et al., 2018), DVD-GAN-FP (Clark et al., 2019), Video Transformer (Weissenborn et al., 2019), Latent Video Transformer (LVT) (Rakhimov et al., 2020), and TrIVD-GAN (Luc et al., 2020) are our baselines

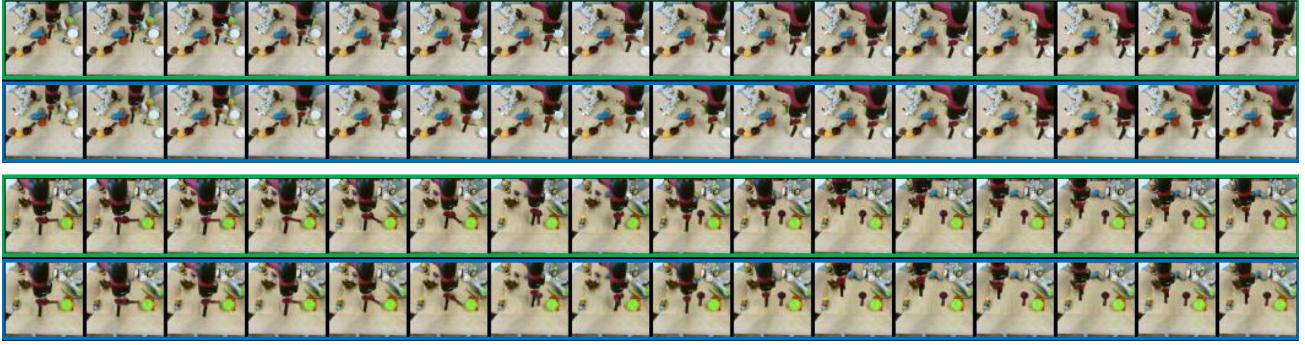


Figure 4. VQ-VAE reconstructions for BAIR Robot Pushing. The original videos are contained in green boxes and reconstructions in blue.



Figure 5. BAIR Robot Pushing samples from a single-frame conditioned VideoGPT model. Frames highlighting in red are conditioning frames. Although all videos follow the same starting frame, the samples eventually diverge to varied trajectories.

paring our method with prior work. Although our method does not achieve state of the art, it is able to produce very realistic samples competitive with the best performing GANs.

Qualitatively, Fig 4 shows VQ-VAE reconstructions on BAIR. Fig 5 shows samples primed with a single frames. We can see that our method is able to generate realistically looking samples. In addition, we see that VideoGPT is able to sample different trajectories from the same initial frame, showing that it is not simply copying the dataset.

4.4. ViZDoom

For ViZDoom, we use the same VQ-VAE and transformer architectures as for the BAIR dataset, with the exception that the transformer is trained without single-frame conditioning. We collect the training data by training a policy in each ViZDoom environment, and collecting rollouts of the final trained policies. The total dataset size consists of 1000 episodes of length 100 trajectories, split into an 8:1:1 train / validation / test ratio. We experiment on the Health Gathering Supreme and Battle2 ViZDoom environments, training both unconditional and action-conditioned priors.

VideoGPT is able to capture complex 3D camera movements and environment interactions. In addition, action-conditioned samples are visually consistent with the input action sequence and show a diverse range of backgrounds and scenarios under different random generations for the same set of actions. Samples can be found in Appendix B.⁴

4.5. UCF-101

UCF-101 (Soomro et al., 2012) is an action classification dataset with 13,320 videos from 101 different classes. We train unconditional VideoGPT models on 16 frame 64×64 and 128×128 videos, where the original videos have their shorter side scaled to 128 pixels, and then center cropped.

Table 2 shows results comparing Inception Score⁵ (IS) (Sal-

⁴VGAN (Vondrick et al., 2016), TGAN (Saito et al., 2017), MoCoGAN (Tulyakov et al., 2018), Progressive VGAN (Acharya et al., 2018), TGAN-F (Kahembwe & Ramamoorthy, 2020), TGANv2 (Saito & Saito, 2018), DVD-GAN (Clark et al., 2019) are our baselines for IS on UCF-101.

⁵Inception Score is calculated using the code at <https://github.com/pfnet-research/tgan2>

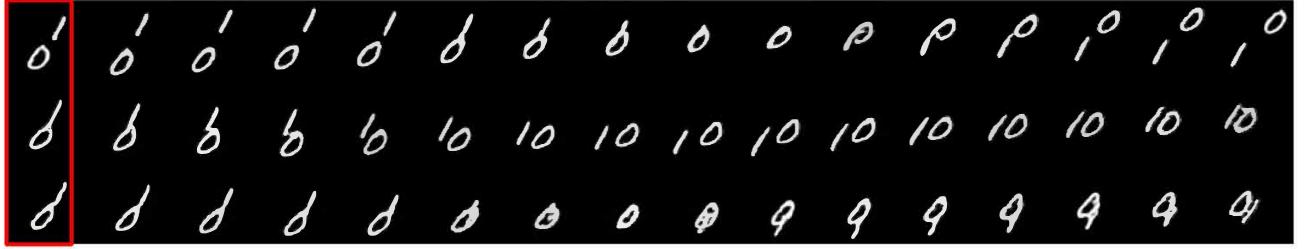


Figure 6. Moving MNIST samples conditioned on a single given frame (red).

Table 2. IS on UCF-101

Method ⁴	IS (\uparrow)
VGAN	8.31 ± 0.09
TGAN	11.85 ± 0.07
MoCoGAN	12.42 ± 0.03
Progressive VGAN	14.56 ± 0.05
TGAN-F	22.91 ± 0.19
VideoGPT (ours)	24.69 ± 0.30
TGANv2	28.87 ± 0.67
DVD-GAN	32.97 ± 1.7

Table 3. Ablation on attention in VQ-VAE. R-FVD is with reconstructed examples

VQ-VAE Architecture	NMSE (\downarrow)	R-FVD (\downarrow)
No Attention	0.0041	15.3
With Attention	0.0033	14.9

We compare VQ-VAE with and without axial attention blocks as shown in Table 3. Empirically, incorporating axial attention into the VQ-VAE architecture improves reconstruction (NMSE) performance, and has better reconstruction FVD. Note that in order to take into account the added parameter count from attention layers, we increase the number of convolutional residual blocks in the "No Attention" version for better comparability. Fig 4 shows samples of videos reconstructed by VQ-VAE with axial attention module.

Larger prior network capacity increases performance.

Table 4. Ablations comparing the number of transformer layers

Transformer Layers	bits/dim	FVD (\downarrow)
2	2.84	120.4 ± 6.0
4	2.52	110.0 ± 2.4
8	2.39	103.3 ± 2.2
16	2.05	103.6 ± 2.0

Computational efficiency is a primary advantage of our method, where we can first use the VQ-VAE to downsample by space-time before learning an autoregressive prior. Lower resolution latents allow us to train a larger and more expressive prior network to learn complex data distributions under memory constraints. We run an ablation on the prior network size which shows that a larger transformer network produces better results. Table 4 shows the results of training transformers of varied number of layers on BAIR. We can see that for BAIR, our method benefits from training larger models, where the bits per dim shows substantial improvement in increasing layers, and FVD and sample quality show increments in performance up until around 8 layers.

mans et al., 2016) calculations against various baselines. Unconditionally generated samples are shown in Figure 7. Similarly observed in (Clark et al., 2019), we notice that that VideoGPT easily overfits UCF-101 with a train loss of 3.40 and test loss of 3.12, suggesting that UCF-101 may be too small a dataset of the relative complexity of the data itself, and more exploration would be needed on larger datasets.

4.6. Tumblr GIF (TGIF)

TGIF (Li et al., 2016) is a dataset of 103,068 selected GIFs from Tumblr, totalling around 100,000 hours of video. Figure 8 shows samples from a trained unconditional VideoGPT model. We see that the video sample generations are able to capture complex interactions, such as camera movement, scene changes, and human and object dynamics. Unlike UCF-101, VideoGPT did not overfit on TGIF with a train loss of 2.87 and test loss 2.86.

4.7. Ablations

In this section, we perform ablations on various architectural design choices for VideoGPT.

Axial-attention in VQ-VAE increases reconstruction and generation quality.

Figure 7. 128×128 UCF-101 unconditional samplesFigure 8. 64×64 TGIF unconditional samples

A balanced temporal-spatial downsampling in VQ-VAE latent space increase performance.

Table 5. Ablations comparing different VideoGPT latent sizes on BAIR. R-FVD is the FVD of VQ-VAE reconstructions, and FVD* is the FVD between samples generated by VideoGPT and samples encoded-decoded from VQ-VAE. For each partition below, the total number of latents is the same with varying amounts of spatio-temporal downsampling

Latent Size	R-FVD (\downarrow)	FVD (\downarrow)	FVD* (\downarrow)
$4 \times 16 \times 16$	82.1	135.4 ± 3.7	81.8 ± 2.3
$16 \times 8 \times 8$	108.1	166.9 ± 3.1	81.6 ± 2.2
$8 \times 16 \times 16$	49.9	124.7 ± 2.7	90.2 ± 2.4
$1 \times 64 \times 64$	41.6	126.7 ± 3.1	98.1 ± 3.6
$4 \times 32 \times 32$	28.3	104.6 ± 2.7	90.6 ± 2.7
$16 \times 16 \times 16$	32.8	113.4 ± 2.5	94.9 ± 1.7
$2 \times 64 \times 64$	22.4	124.3 ± 1.4	104.4 ± 2.5
$8 \times 32 \times 32$	14.9	103.6 ± 2.0	94.6 ± 1.5
$4 \times 64 \times 64$	15.7	109.4 ± 2.1	102.3 ± 2.8
$16 \times 32 \times 32$	10.1	118.4 ± 3.2	113.8 ± 3.3

A larger downsampling ratio results in a smaller latent code size, which allows us to train larger and more expressive prior models. However, limiting the expressivity of the discrete latent codes may introduce a bottleneck that results in poor VQ-VAE reconstruction and sample quality. Thus, VideoGPT presents an inherent trade-off between the size of the latents, and the allowed capacity of prior network.

Table 5 shows FVD results from training VideoGPT on varying latent sizes for BAIR. We can see that larger latents sizes have better reconstruction quality (lower R-FVD), however, the largest latents $16 \times 32 \times 32$ does not perform the best sample-quality-wise due to limited compute constraints on prior model size. On the other hand, the smallest set of latents $4 \times 16 \times 16$ and $16 \times 8 \times 8$ have poor reconstruction quality and poor samples. There is a sweet-spot in the trade-off at around $8 \times 32 \times 32$ where we observe the best sample quality.

In addition to looking at the total number of latents, we also investigate the appropriate downsampling for each latent resolution. Each partition in Table 5 shows latent sizes with the same number of total latents, each with different spatio-temporal downsampling allocations. Unsurprisingly, we find that a balance of downsampling ratio ($2 \times 2 \times 2$, corresponding to latent size $8 \times 32 \times 32$) between space and time is the best, as opposed to downsampling over only space or only time.

Further increasing the number of latent codes does not affect performance.

Table 6. Ablations comparing the number of codebook codes

# of Codes	R-FVD (\downarrow)	FVD (\downarrow)	bits/dim
256	18.2	103.8 ± 3.7	1.55
1024	14.9	103.6 ± 2.0	2.05
4096	11.3	103.9 ± 5.1	2.60

In Table 6, we show experimental results for running VideoGPT with different number of codes in the codebooks. For all three runs, the VQ-VAE latent code vector has size $8 \times 32 \times 32$. In the case of BAIR, we find that reconstruction quality improves with increasing the number of codes due to better expressivity in the discrete bottleneck. However, they ultimately do not affect sample quality. This may be due to the fact that in the case of BAIR, using 256 codes surpasses a base threshold for generation quality.

Using one VQ-VAE codebook instead of multiple improves performance.

Table 7. Ablations comparing the number of codebooks

Latent Size	R-FVD (\downarrow)	FVD (\downarrow)	bits/dim
$8 \times 32 \times 32 \times 1$	14.9	103.6 ± 2.0	2.05
$16 \times 16 \times 16 \times 2$	17.2	106.3 ± 1.7	2.41
$8 \times 16 \times 16 \times 4$	17.7	131.4 ± 2.9	2.68
$4 \times 16 \times 16 \times 8$	23.1	135.7 ± 3.3	2.97

In our main results, we use one codebook for VQ-VAE. In Table 7, we compare VideoGPT with different number of codebooks. Specifically, multiple codebooks is implemented by multiplying VQ-VAE’s encode output channel dimension by C times, where C is the number of codebooks. The encoder output is then sliced along channel dimension, and each slice is quantized through a separate codebook. As a result, the size of the discrete latents are of dimension $T \times H \times W \times C$, as opposed to $T \times H \times W$ when using a single codebook. Generally, multiple codebooks may be more favorable over increasing the downsampling resolution as multiple codebooks allows a combinatorially better scaling in bottleneck complexity. In our experiments, we increase the number of codebooks, and reduce spatio-temporal resolutions on latent sizes to keep the size of the latent space constant. We see that increasing the number of codebooks worsens sample quality performance, and the best results are attained at the highest resolution with one codebook. Nevertheless, incorporating multiple codebooks might shows its advantage when trained with a larger dataset or a different VQ-VAE architecture design.

5. Related Work

Video Prediction The problem of video prediction (Srivastava et al., 2015) is quite related to video generation in that the latter is one way to solve the former. Plenty of methods have been proposed for video prediction on the BAIR Robot dataset (Finn et al., 2016; Ebert et al., 2017; Babaeizadeh et al., 2017; Denton et al., 2017; Denton & Fergus, 2018; Lee et al., 2018) where the future frames are predicted given the past frame(s) and (or) action(s) of a robot arm moving

across multiple objects thereby benchmarking the ability of video models to capture object-robot interaction, object permanence, robot arm motion, etc. Translating videos to videos is another paradigm to think about video prediction with a prominent example being vid2vid (Wang et al., 2018). The vid2vid framework uses automatically generated supervision from more abstract information such as semantic segmentation (Luc et al., 2017) masks, keypoints, poses, edge detectors, etc to further condition the GAN based video translation setup.

Video Generation Most modern generative modeling architectures allow for easy adaptation of unconditional video generation to conditional versions through conditional batch-norm (Brock et al., 2018), concatenation (Salimans et al., 2017; van den Oord et al., 2016c), etc. Video Pixel Networks (Kalchbrenner et al., 2017) propose a convolutional LSTM based encoding of the past frames to be able to generate the next frame pixel by pixel autoregressively with a PixelCNN (van den Oord et al., 2016c) decoder. The architecture serves both as a video generative as well as predictive model, optimized through log-likelihood loss at the pixel level. Subscale Video Transformers (Weissenborn et al., 2019) extend the idea of Subscale Pixel Networks (Menick & Kalchbrenner, 2018) for video generation at the pixel level using the subscale ordering across space and time. However, the sampling time and compute requirements are large for these models. In the past, video specific architectures have been proposed for GAN based video generation with primitive results by (Vondrick et al., 2016). Recently, DVD-GAN proposed by (Clark et al., 2019) adopts a BigGAN like architecture for videos with disentangled (axial) non-local (Wang et al., 2017) blocks across space and time. They present a wide range of results, unconditional, past frame(s) conditional, and class conditional video generation.

Other examples of prior work with video generation of GANs include (Saito et al., 2017), (Tulyakov et al., 2018), (Acharya et al., 2018), (Yushchenko et al., 2019). In addition, (Saito & Saito, 2018) and (Kahembwe & Ramamoorthy, 2020) propose more scalable and efficient GAN models for training on less compute. Our approach builds on top of VQ-VAE (Van Den Oord et al., 2017) by adapting it for video generation. A clean architecture with VQ-VAE for video generation has not been presented yet and we hope VideoGPT is useful from that standpoint. While VQ-VAE-2 (Razavi et al., 2019) proposes using multi-scale hierarchical latents and SNAIL blocks (Chen et al., 2017) (and this setup has been applied to videos in (Walker et al., 2021)), the pipeline is inherently complicated and hard to reproduce. For simplicity, ease of reproduction and presenting the first VQ-VAE based video generation model with minimal complexity, we stick with a single scale of discrete latents and transformers for the autoregressive priors, a design choice also adopted in DALL-E (Ramesh et al., 2021).

6. Conclusion

We have presented VideoGPT, a new video generation architecture adapting VQ-VAE and Transformer models typically used for image generation to the domain of videos with minimal modifications. We have shown that VideoGPT is able to synthesize videos that are competitive with state-of-the-art GAN based video generation models. We have also presented ablations on key design choices used in VideoGPT which we hope is useful for future design of architectures in video generation.

Acknowledgement

The work was in part supported by NSF NRI Grant #2024675 and by Berkeley Deep Drive.

References

Acharya, D., Huang, Z., Paudel, D. P., and Van Gool, L. Towards high resolution video generation with progressive growing of sliced wasserstein gans. *arXiv preprint arXiv:1810.02419*, 2018.

Amodei, D. and Hernandez, D. Ai and compute. *Heruntergeladen von https://blog.openai.com/ai-and-compute*, 2018.

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Babaeizadeh, M., Finn, C., Erhan, D., Campbell, R. H., and Levine, S. Stochastic variational video prediction. *arXiv preprint arXiv:1710.11252*, 2017.

Bińkowski, M., Donahue, J., Dieleman, S., Clark, A., Elsen, E., Casagrande, N., Cobo, L. C., and Simonyan, K. High fidelity speech synthesis with adversarial networks. *arXiv preprint arXiv:1909.11646*, 2019.

Brock, A., Donahue, J., and Simonyan, K. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

Chen, M., Radford, A., Child, R., Wu, J., Jun, H., Dhariwal, P., Luan, D., and Sutskever, I. Generative pretraining from pixels. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Chen, X., Mishra, N., Rohaninejad, M., and Abbeel, P. Pixelsnail: An improved autoregressive generative model. *arXiv preprint arXiv:1712.09763*, 2017.

Child, R. Very deep vaes generalize autoregressive models and can outperform them on images. *arXiv preprint arXiv:2011.10650*, 2020.

Child, R., Gray, S., Radford, A., and Sutskever, I. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Clark, A., Donahue, J., and Simonyan, K. Adversarial video generation on complex datasets, 2019.

Denton, E. and Fergus, R. Stochastic video generation with a learned prior. *arXiv preprint arXiv:1802.07687*, 2018.

Denton, E. L. et al. Unsupervised learning of disentangled representations from video. In *Advances in neural information processing systems*, pp. 4414–4423, 2017.

Dhariwal, P., Jun, H., Payne, C., Kim, J. W., Radford, A., and Sutskever, I. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.

Dinh, L., Krueger, D., and Bengio, Y. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using Real NVP. *arXiv preprint arXiv:1605.08803*, 2016.

Ebert, F., Finn, C., Lee, A. X., and Levine, S. Self-supervised visual planning with temporal skip connections. *arXiv preprint arXiv:1710.05268*, 2017.

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

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

Ho, J., Chen, X., Srinivas, A., Duan, Y., and Abbeel, P. Flow++: Improving flow-based generative models with variational dequantization and architecture design. *arXiv preprint arXiv:1902.00275*, 2019a.

Ho, J., Kalchbrenner, N., Weissenborn, D., and Salimans, T. Axial attention in multidimensional transformers. *arXiv preprint arXiv:1912.12180*, 2019b.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020.

Jouppi, N. P., Young, C., Patil, N., Patterson, D., Agrawal, G., Bajwa, R., Bates, S., Bhatia, S., Boden, N., Borchers, A., et al. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th Annual International Symposium on Computer Architecture*, pp. 1–12, 2017.

Kahembwe, E. and Ramamoorthy, S. Lower dimensional kernels for video discriminators. *Neural Networks*, 132: 506–520, 2020.

Kalchbrenner, N., Oord, A. v. d., Simonyan, K., Danihelka, I., Vinyals, O., Graves, A., and Kavukcuoglu, K. Video pixel networks. *arXiv preprint arXiv:1610.00527*, 2016.

Kalchbrenner, N., Oord, A., Simonyan, K., Danihelka, I., Vinyals, O., Graves, A., and Kavukcuoglu, K. Video pixel networks. In *International Conference on Machine Learning*, pp. 1771–1779. PMLR, 2017.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

Kempka, M., Wydmuch, M., Runc, G., Toczek, J., and Jaśkowski, W. Vizdoom: A doom-based ai research platform for visual reinforcement learning. In *2016 IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 1–8. IEEE, 2016.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint arXiv:1807.03039*, 2018.

Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. *Proceedings of the 2nd International Conference on Learning Representations*, 2013.

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. Improving variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, 2016.

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Le Gall, D. Mpeg: A video compression standard for multimedia applications. *Communications of the ACM*, 34(4): 46–58, 1991.

Lee, A. X., Zhang, R., Ebert, F., Abbeel, P., Finn, C., and Levine, S. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018.

Li, Y., Song, Y., Cao, L., Tetreault, J., Goldberg, L., Jaimes, A., and Luo, J. Tgif: A new dataset and benchmark on animated gif description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4641–4650, 2016.

Luc, P., Neverova, N., Couprise, C., Verbeek, J., and LeCun, Y. Predicting deeper into the future of semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

Luc, P., Clark, A., Dieleman, S., Casas, D. d. L., Doron, Y., Cassirer, A., and Simonyan, K. Transformation-based adversarial video prediction on large-scale data. *arXiv preprint arXiv:2003.04035*, 2020.

Menick, J. and Kalchbrenner, N. Generating high fidelity images with subscale pixel networks and multidimensional upscaling. *arXiv preprint arXiv:1812.01608*, 2018.

Oord, A. v. d., Li, Y., Babuschkin, I., Simonyan, K., Vinyals, O., Kavukcuoglu, K., Driessche, G. v. d., Lockhart, E., Cobo, L. C., Stimberg, F., et al. Parallel wavenet: Fast high-fidelity speech synthesis. *arXiv preprint arXiv:1711.10433*, 2017.

Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, Ł., Shazeer, N., and Ku, A. Image transformer. *arXiv preprint arXiv:1802.05751*, 2018.

Prenger, R., Valle, R., and Catanzaro, B. Waveglow: A flow-based generative network for speech synthesis. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3617–3621. IEEE, 2019.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.

Rakhimov, R., Volkhonskiy, D., Artemov, A., Zorin, D., and Burnaev, E. Latent video transformer. *arXiv preprint arXiv:2006.10704*, 2020.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. Zero-shot text-to-image generation. *arXiv preprint arXiv:2102.12092*, 2021.

Razavi, A., van den Oord, A., and Vinyals, O. Generating diverse high-fidelity images with vq-vae-2. In *Advances in Neural Information Processing Systems*, pp. 14866–14876, 2019.

Saito, M. and Saito, S. Tganv2: Efficient training of large models for video generation with multiple subsampling layers. *arXiv preprint arXiv:1811.09245*, 2018.

Saito, M., Matsumoto, E., and Saito, S. Temporal generative adversarial nets with singular value clipping. In *Proceedings of the IEEE international conference on computer vision*, pp. 2830–2839, 2017.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. *arXiv preprint arXiv:1606.03498*, 2016.

Salimans, T., Karpathy, A., Chen, X., and Kingma, D. P. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*, 2017.

Sohl-Dickstein, J., Weiss, E. A., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. *arXiv preprint arXiv:1503.03585*, 2015.

Sønderby, C. K., Espeholt, L., Heek, J., Dehghani, M., Oliver, A., Salimans, T., Agrawal, S., Hickey, J., and Kalchbrenner, N. Metnet: A neural weather model for precipitation forecasting. *arXiv preprint arXiv:2003.12140*, 2020.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, pp. 11918–11930, 2019.

Soomro, K., Zamir, A. R., and Shah, M. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

Srivastava, N., Mansimov, E., and Salakhudinov, R. Unsupervised learning of video representations using lstms. In *International conference on machine learning*, pp. 843–852, 2015.

Tran, D., Bourdev, L., Fergus, R., Torresani, L., and Paluri, M. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 4489–4497, 2015.

Tulyakov, S., Liu, M.-Y., Yang, X., and Kautz, J. Mocogan: Decomposing motion and content for video generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1526–1535, 2018.

Vahdat, A. and Kautz, J. Nvae: A deep hierarchical variational autoencoder. *arXiv preprint arXiv:2007.03898*, 2020.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016a.

van den Oord, A., Kalchbrenner, N., and Kavukcuoglu, K. Pixel recurrent neural networks. *International Conference on Machine Learning (ICML)*, 2016b.

van den Oord, A., Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A., and Kavukcuoglu, K. Conditional image generation with pixelcnn decoders. *arXiv preprint arXiv:1606.05328*, 2016c.

Van Den Oord, A., Vinyals, O., et al. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pp. 6306–6315, 2017.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

Vondrick, C., Pirsiavash, H., and Torralba, A. Generating videos with scene dynamics. In *Advances in neural information processing systems*, pp. 613–621, 2016.

Walker, J., Razavi, A., and Oord, A. v. d. Predicting video with vqvae. *arXiv preprint arXiv:2103.01950*, 2021.

Wallace, G. K. The jpeg still picture compression standard. *IEEE transactions on consumer electronics*, 38(1):xviii–xxxiv, 1992.

Wang, T.-C., Liu, M.-Y., Zhu, J.-Y., Liu, G., Tao, A., Kautz, J., and Catanzaro, B. Video-to-video synthesis. *arXiv preprint arXiv:1808.06601*, 2018.

Wang, X., Girshick, R., Gupta, A., and He, K. Non-local neural networks. *arXiv preprint arXiv:1711.07971*, 2017.

Weissenborn, D., Täckström, O., and Uszkoreit, J. Scaling autoregressive video models. *arXiv preprint arXiv:1906.02634*, 2019.

Yushchenko, V., Araslanov, N., and Roth, S. Markov decision process for video generation. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 0–0, 2019.

Zhang, H., Goodfellow, I., Metaxas, D., and Odena, A. Self-attention generative adversarial networks. In *International Conference on Machine Learning*, pp. 7354–7363. PMLR, 2019.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and

A. Architecture Details and Hyperparameters

A.1. VQ-VAE Encoder and Decoder

Table 8. Hyperparameters of VQ-VAE encoder and decoder models for each dataset

	Moving MNIST	BAIR / RoboNet / ViZDoom	UCF-101 / TGIF
Input size	$16 \times 64 \times 64$	$16 \times 64 \times 64$	$16 \times 64 \times 64$
Latent size	$4 \times 16 \times 16$	$8 \times 32 \times 32$	$4 \times 32 \times 32$
β (commitment loss coefficient)	0.25	0.25	0.25
Batch size	32	32	32
Learning rate	7×10^{-4}	7×10^{-4}	7×10^{-4}
Hidden units	240	240	240
Residual units	128	128	128
Residual layers	2	4	4
Uses attention	No	Yes	Yes
Codebook size	512	1024	1024
Codebook dimension	64	256	256
Encoder filter size	3	3	3
Upsampling conv filter size	4	4	4
Training steps	20k	100K	100K

A.2. Prior Networks

Table 9. Hyperparameters of prior networks for each dataset

	Moving MNIST	BAIR / RoboNet	ViZDoom	UCF-101 / TGIF
Input size	$4 \times 16 \times 16$	$8 \times 32 \times 32$	$8 \times 32 \times 32$	$4 \times 32 \times 32$
Conditional sizes	$1 \times 64 \times 64$	$3 \times 64 \times 64$, 64	60 (HGS), 315 (Battle2)	n/a
Batch size	32	32	32	32
Learning rate	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}
Vocabulary size	512	1024	1024	1024
Attention heads	4	4	4	8
Attention layers	8	16	16	20
Embedding size	192	512	512	1024
Feedforward hidden size	384	2048	2048	4096
Resnet depth	18	34	n/a	n/a
Resnet units	512	512	n/a	n/a
Dropout	0.1	0.2	0.2	0.2
Training steps	80k	150K	150K	200K / 600K

B. ViZDoom Samples

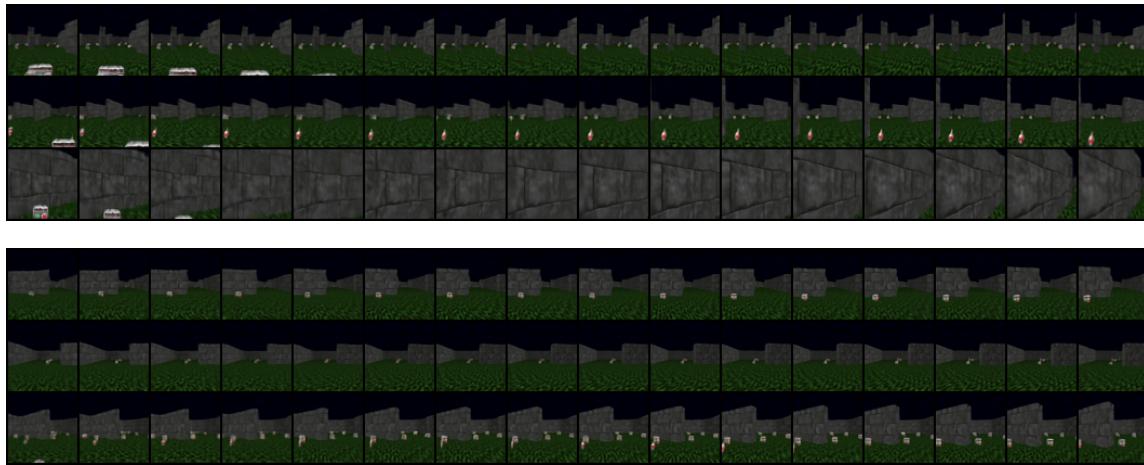


Figure 9. Samples for ViZDoom health gathering supreme environment. (Top) shows unconditionally generated samples. (Bottom) shows samples conditioned on the same action sequence (turn right and go straight).



Figure 10. Samples for ViZDoom battle2 environment. (Top) shows unconditionally generated samples. (Bottom) shows three samples conditioned on the same action sequence (moving forward and right).