Cascaded Video Generation for Videos In-the-Wild

Lluis Castrejon[1], **Nicolas Ballas** [2], **Aaron Courville** [1,3]

[1] Universite de Montreal - Mila Quebec AI Institute
[2] Facebook AI Research
[3] CIFAR AI Chair
lluis.enric.castrejon.subira@umontreal.ca, ballasn@fb.com, aaron.courville@umontreal.ca

Abstract

Videos can often be created by first outlining a global description of the scene and then adding local details. Inspired by this we propose a cascaded model for video generation which follows a coarse to fine approach. First our model generates a low resolution video, establishing the global scene structure, that is then refined by subsequent cascade levels operating at larger resolutions. We train each level sequentially on partial views of the videos. This reduces the computational complexity of our generative model, which scales to high-resolution videos beyond a few frames. We validate our approach on UCF101 and Kinetics-600, for which our model outperforms or matches the state-of-the-art. We further demonstrate the scaling capabilities of our model and train an unconditional three-level model on the BDD100K dataset which generates 256x256 pixels videos with 48 frames.

Introduction

Humans have the ability to simulate novel visual objects and their dynamics using their imagination. This ability is linked to the capacity to perform planning or counter-factual thinking. Replicating this ability in machines is a longstanding challenge that generative models try to address. Advances in generative modeling and increased computational resources have enabled the generation of realistic high-resolution images (Brock, Donahue, and Simonyan 2018) or coherent text in documents (Brown et al. 2020). Yet, models for video generation have been less successful, in part due to their high memory requirements that scale with the generation resolution and length.

When creating visual data, artists often first create a rough outline of the scene, to which then they add local details in one or multiple iterations (Locher 2010). The outline ensures global consistency of the scene and divides the creative process into multiple tractable steps. Inspired by this strategy we propose CVG, a cascaded video generation model which divides the generative process into a set of simpler problems. CVG first generates a low-resolution video that depicts a full video scene at a reduced framerate. This scene outline is then progressively upscaled and temporally interpolated to the desired final resolution by one or more upscaling levels as depicted in Figure 1. Every level in our model outputs a video that serves as the input to the next one, with each level specializing in a particular aspect of the generation.

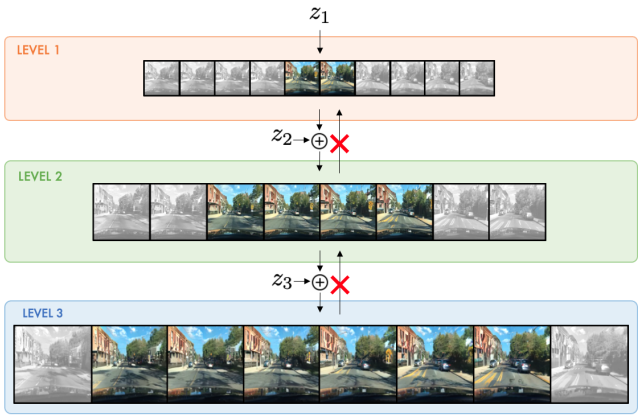

Figure 1: **Cascaded Video Generation** We propose to divide the generative process into multiple simpler problems. CVG first generates a low resolution video that depicts a full scene at a reduced framerate. This scene outline is then progressively upscaled and temporally interpolated. Levels are trained sequentially and do not backprogagate the gradient to the previous levels. Additionally, upscaling levels can be trained on temporal crops of previous level outputs during training (illustrated by the non-shaded images) to reduce their computational requirements. Our model outperforms or matches the state-of-the-art in video generation and enables the generation of longer high resolution videos due to better scaling properties than previous methods.

Levels in our model are trained greedily, i.e. in sequence and not end-to-end. This sequential training allows to consider only one level at a time and thus reduce the training memory requirements. We formulate each level as a adversarial game that we solve leveraging the GAN framework, and we theoretically demonstrate that our training setup has the same global solution as an end-to-end model.

To further reduce the computational needs of our model, upscaling levels can be applied only on temporal crops from previous outputs during training. Despite this temporally-local training, upscaling levels are capable of producing videos with temporal coherence at inference time, as they upscale the output of the first level, which is temporally complete albeit at a reduced framerate. This makes CVG more

scalable than previous methods, making it capable of generating high resolution videos with a larger number of frames.

Our contributions can be summarized as follows:

- We define a cascade model for video generation which divides the generation process into multiple tractable steps.
- We empirically validate our approach on UCF101, Kinetics-600 and BDD100K, large-scale datasets with complex videos in real-world scenarios. CVG matches or outperforms the state-of-art video generation models on these datasets.
- We demonstrate that our approach has better scaling properties than comparable non-cascaded approaches and train a three-level model to generate videos with 48 frames at a resolution of 256x256 pixels. To the best of our knowledge, we propose the first unconditional video generation model capable of generating such long videos at this resolution for the BDD100K dataset.

Cascaded Video Generation

Video scenes can be created by first outlining the global scene and then adding local details. Following this intuition we propose CVG, a cascade model in which each stage only treats a lower dimensional view of the data. Video generation models struggle to scale to high frame resolutions and long temporal ranges. The goal of our method is to break down the generation process into smaller steps which require less computational resources when considered independently.

Problem Setting We consider a dataset of videos $(\mathbf{x}_1, ..., \mathbf{x}_n)$ where each video $\mathbf{x}_i = (\mathbf{x}_{i;0}, ..., \mathbf{x}_{i;T})$ is a sequence of T frames $\mathbf{x}_{i;t} \in \mathbb{R}^{H \times W \times 3}$. Let f_s denote a spatial bilinear downsampling operator and f_t a temporal subsampling operator. For each video $\mathbf{x}_i$, we can obtain lower resolution views of our video by repeated application of f_s and f_t, i.e. $\mathbf{x}_i^l = f_s(f_t(\mathbf{x}_i^{l+1}), \forall l \in [1..L]$ with $\mathbf{x}_i^L = \mathbf{x}_i$.

Each $(\mathbf{x}_i^1, ..., \mathbf{x}_i^L)$ comes from a joint data distribution p_d. The task of video generation consists in learning a generative distribution p_g such that $p_g = p_d$.

Hierarchical Generative Model We define a generative model that approximates the joint data distribution according to the following factorization:

$$p_g(\mathbf{x}^1, ..., \mathbf{x}^L) = p_{g_L}(\mathbf{x}^L|\mathbf{x}^{L-1})...p_{g_2}(\mathbf{x}^2|\mathbf{x}^1)p_{g_1}(\mathbf{x}^1). \quad (1)$$

Each p_{g_i} defines a level in our model. This formulation allows us to decompose the generative process in to a set of smaller problems. The first level p_{g_1} produces low resolution and temporally subsampled videos from a latent variable. For subsampling factors K_T and K_S (for time and space respectively), the initial level generates videos $\mathbf{x}_i^1 = (\mathbf{x}_{i;0}^1, \mathbf{x}_{i;K_T}^1, \mathbf{x}_{i;2K_T}^1, ..., \mathbf{x}_{i;T}^1)$, which is a sequence of $\frac{T}{K_T}$ frames $\mathbf{x}_{i;t}^1 \in \mathbb{R}^{\frac{H}{K_S} \times \frac{W}{K_S} \times 3}$. The output of the first level is spatially upscaled and temporally interpolated by one or more subsequent upscaling levels.

Training We train our model greedily one level at a time and in order, i.e. we train the first level to generate global but downscaled videos, and then we train upscaling stages on previous level outputs one after each other. We do not train the levels in an end-to-end fashion, which allows us to break down the computation into tractable steps by only training one level at a time. We formulate a GAN objective for each stage of our model. We consider the distribution p_{g_1} in eq. 1 and solve a min-max game with the following value function:

$$\mathbb{E}_{\mathbf{x}^1 \sim p_d}[\log(D_1(\mathbf{x}^1))] + \mathbb{E}_{\mathbf{z}_1 \sim p_{z_1}}[\log(1 - D_1(G_1(\mathbf{z}_1)))], \quad (2)$$

where G_1 and D_1 are the generator/discriminator associated with the first stage and p_{z_1} is a noise distribution. This is the standard GAN objective (Goodfellow et al. 2014). For upscaling levels corresponding to $p_{g_l}, l > 1$, we consider the following value function:

$$\mathbb{E}_{\mathbf{x}^{l-1}, ..., \mathbf{x}^1 \sim p_d} \mathbb{E}_{\mathbf{x}^l \sim p_d(.|\mathbf{x}^{l-1}, ..., \mathbf{x}^1)}[\log(D_l(\mathbf{x}^l, \mathbf{x}^{l-1}))] + \mathbb{E}_{\hat{\mathbf{x}}^{l-1} \sim p_{g_{l-1}}} \mathbb{E}_{\mathbf{z}_l \sim p_{z_l}}[\log(1 - D_l(G_l(\mathbf{z}_l, \hat{\mathbf{x}}^{l-1}), \hat{\mathbf{x}}^{l-1}))], \quad (3)$$

where G_l, D_l are the generator and discriminator of the current level and $p_{g_{l-1}}$ is the generative distribution of the level $l-1$.[1] The min-max game associated with this value function has a global minimum when the two joint distributions are equal, $p_d(\mathbf{x}, ..., \mathbf{x^l}) = p_{g_l}(\mathbf{x^l}|\mathbf{x}^{l-1})..p_{g_1}(\mathbf{x}^1)$ (Dumoulin et al. 2016; Donahue, Krähenbühl, and Darrell 2016). For more details about the training objectives for each level and a complete proof that the sequential training admits the same global minimum than an end-to-end training refer to the Appendix. We also see from eq. 3 that the discriminator for upscaling stages operates on pairs $(\mathbf{x}^l, \mathbf{x}^{l-1})$ videos to determine whether they are real or fake. This ensures that the upscaling stages are grounded on their inputs, i.e that $\mathbf{x}^l$ "matches" its corresponding $\mathbf{x}^{l-1}$.

Partial View Training Computational requirements for upscaling levels can be high when generating large outputs, even if each level is trained sequentially. As we increase the length and resolution of a generation, models need to store large activation tensors during training that use large amounts of GPU memory. To reduce the computational requirements, we can apply the upscaling levels only on temporal crops of their inputs during training. This strategy reduces training costs since we upscale smaller tensors, at the expense of having less available context to interpolate frames. At inference time, CVG generates videos by applying the upscaling levels on their inputs without any temporal cropping. Upscaling levels are convolutional, i.e. they learn functions that are applied in a sliding window manner over their inputs. Since we do not temporally crop the inputs at inference time, the upscaling function is applied to all possible input windows to generate a full video, as the first level generates a downscaled but temporally complete video.

Model Parametrization

In this section we describe the parametrization of the different levels of CVG. We keep the discussion at a high level, briefly mentioning the main components of our model. Precise details on the architecture are provided in the Appendix.

[1] We simplify the notation and denote by $p_{g_{l-1}}(\mathbf{x^{l-1}})$ the joint distribution $p_{g_{l-1}}(\mathbf{x}^{l-1}|\mathbf{x}^{l-1})...p_{g_2}(\mathbf{x}^2|\mathbf{x}^1)p_{g_1}(\mathbf{x}^1)$.

First Level The first level generator stacks units composed by a ConvGRU layer (Ballas et al. 2015), modeling the temporal information, and 2D-ResNet blocks that upsample the spatial resolution. Similar to MoCoGAN (Tulyakov et al. 2018) and DVD-GAN (Clark, Donahue, and Simonyan 2019), we use a dual discriminator with both a spatial discriminator that randomly samples k full-resolution frames and discriminates them individually, and a temporal discriminator that processes spatially downsampled but full-length videos.

Upsampling Levels The upsampling levels are composed by a conditional generator and three discriminators (spatial, temporal and matching). The conditional generator produces an upscaled version $\hat{\mathbf{x}}^{\mathbf{l}}$ of a lower resolution video $\hat{\mathbf{x}}^{\mathbf{l-1}}$. To discriminate samples from real videos, upscaling stages use a spatial and temporal discriminator, as in the first level. Additionally, we introduce a matching discriminator. The goal of the matching discriminator is to ensure that the output is a valid upsampling of the input. Without this discriminator, the upsampling generator could learn to ignore the low resolution input video. The conditional generator is trained jointly with the spatial, temporal and matching discriminators.

Conditional Generator The conditional generator takes as input a lower resolution video $\hat{\mathbf{x}}^{l-1}$, a noise vector $\mathbf{z}$ and optionally a class label y, and generates $\hat{\mathbf{x}}^{l}$. We increase the duration of the low resolution video $\hat{\mathbf{x}}^{l}$ to the same temporal duration as the output by repeating its frames before feeding it to the generator as a condition.

Our conditional generator stacks units composed by one 3D-ResNet block and two 2D-ResNet blocks. Spatial upsampling is performed gradually by progressively increasing the resolution of the generator blocks. To condition the generator we add residual connections (He et al. 2016; Srivastava, Greff, and Schmidhuber 2015) from the low-resolution video to the output of each generator unit. We sum nearest-neighbor interpolations of the lower resolution input to each unit output. We do not use skip connections for units whose outputs have higher spatial resolution than the lower dimensional video input, i.e. we do not upscale the low resolution video.

Matching Discriminator The matching discriminator uses an architecture like that of the temporal discriminator. It discriminates real or generated input-output pairs. The output is downsampled to the same size as the input, and both tensors are concatenated on the channel dimension. A precise description of all discriminator architectures can be found in the Appendix.

Related Work

The modern video generation literature (Ranzato et al. 2014; Srivastava, Mansimov, and Salakhudinov 2015) was first started as a result of adapting techniques for language modeling to video. Since then, many papers have proposed different approaches to represent and generate videos (Luc et al. 2017, 2018; Villegas et al. 2017a,b; Xue et al. 2016), including different kinds of tasks, conditionings and models. We review the most common types of generative video models below.

Autoregressive models (Larochelle and Murray 2011; Dinh, Sohl-Dickstein, and Bengio 2016; Kalchbrenner et al. 2017; Reed et al. 2017; Weissenborn, Täckström, and Uszkoreit 2020; Yan et al. 2021) model the conditional probability of each pixel value given the previous ones. They do not use latent variables and their training can be easily parallelized. Inference in autoregressive models often requires a full forward pass for each output pixel, which does not scale well to long high resolution videos whose output dimensionality can be in the million pixels.

Normalizing flows (Rezende and Mohamed 2015; Kingma and Dhariwal 2018; Kumar et al. 2019) learn bijective functions that transform latent variables into data samples. By defining bijective functions, normalizing flows are able to directly maximize the data likelihood, which eases their training. However they require the latent variable to have the same dimensionality as its output, which becomes an obstacle when generating videos due to their large dimensionality.

Variational AutoEncoders (VAEs) (Kingma and Welling 2013; Rezende, Mohamed, and Wierstra 2014; Babaeizadeh et al. 2017) also transform latent variables into data samples. While more scalable, VAEs often produce blurry results when compared to other generative models. Models based on VRNNs (Chung et al. 2015; Denton and Fergus 2018; Lee et al. 2018; Castrejon, Ballas, and Courville 2019) use one latent variable per video frame and often produce better results.

Generative Adversarial Networks (GANs) are another type of latent variable models which optimize a min-max game between a generator G and a discriminator D trained to tell real and generated data apart (Goodfellow et al. 2014). Empirically, GANs usually produce better samples than competing approaches but might suffer from mode collapse. GAN models for video were first proposed in (Vondrick, Pirsiavash, and Torralba 2016b,a; Mathieu, Couprie, and LeCun 2015). In recent work, SAVP (Lee et al. 2018) proposed to use the VAE-GAN (Larsen et al. 2015) framework for video. TGANv2 (Saito and Saito 2018) improves upon TGAN (Saito, Matsumoto, and Saito 2017) and proposes a video GAN trained on data windows, similar to our approach. However, unlike TGANv2, our model is composed of multiple stages which are not trained jointly. MoCoGAN (Tulyakov et al. 2018) first introduced a dual discriminator architecture for video, with DVD-GAN (Clark, Donahue, and Simonyan 2019) scaling up this approach to high resolution videos in the wild. DVD-GAN outperforms MoCoGAN and TGANv2, and is arguably the current state-of-the-art in adversarial video generation.

Our model is also related to work that proposes hierarchical or progressive training approaches for generative models (Karras et al. 2017; Denton et al. 2015; Xiong et al. 2018; Zhao et al. 2020). Our model is different from those approaches in that our stages are trained greedily in separate steps without backpropagation from one stage to the other, which reduces its computational requirements.

Experiments

In this section we empirically validate our proposed approach. First, we show that our approach outperforms or matches the state-of-the-art on the Kinetics-600 and UCF101 datasets. Then, we analyze the scaling properties of our model in

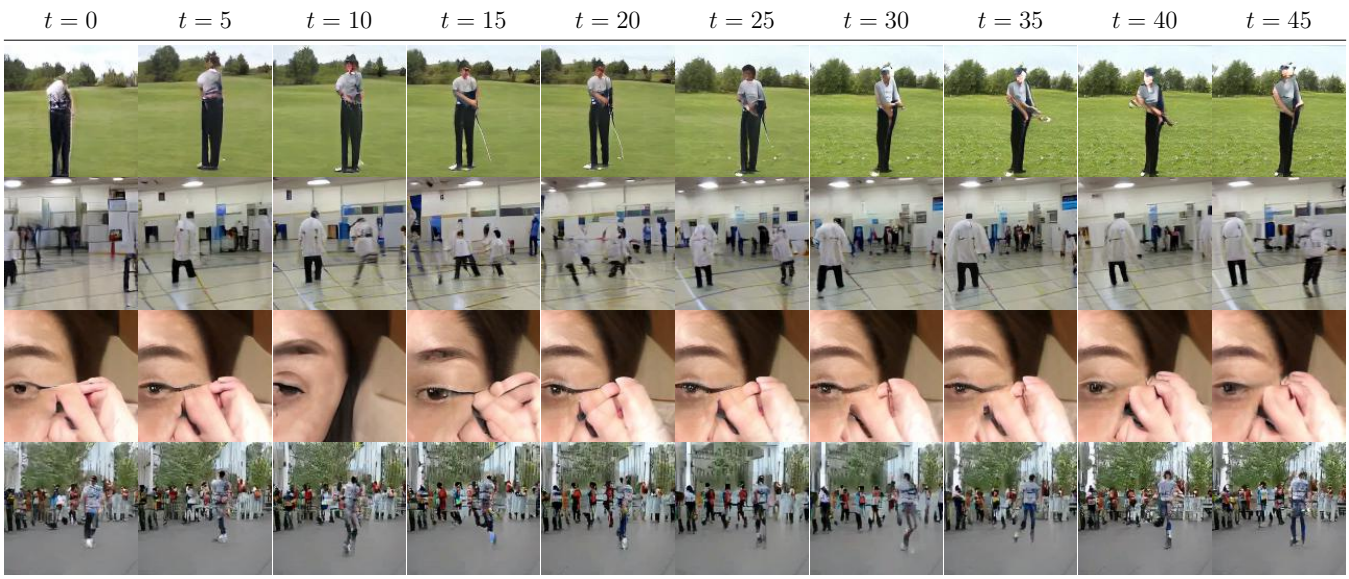

Figure 2: **Randomly selected CVG 48/128x128 frame samples for Kinetics-600:** These samples were generated by unrolling CVG 12/128x128 to generate 48 frame sequences, 4 times its training horizon. Each row shows frames from the same sample at different timesteps. The generations are temporally consistent and the frame quality does not degrade over time.

		Evaluated on 12 frames			Evaluated on 48 frames		
Model	Trained on	IS (↑)	FID (↓)	FVD (↓)	IS (↑)	FID (↓)	FVD (↓)
DVD-GAN	12/128x128	77.45	**1.16**	-	N/A	N/A	N/A
DVD-GAN	48/128x128	N/A	N/A	N/A	**81.41**	28.44	-
2-Level CVG	12/128x128	**104.00**	2.09	591.90	77.36	**14.00**	517.21

Table 1: **Results on Kinetics-600 128x128** We compare our two-level CVG model against the reported metrics for two DVD-GAN models (Clark, Donahue, and Simonyan 2019), one trained to generate 12 frames and one trained to generate 48 frames. Our model is trained to only generate 12 frames and is able to match the performance of the 12-frame DVD-GAN model. Additionally, the same CVG model is able to generate 48 frames when applied convolutionally over the full first level output. In that setup our model also matches the quality of a 128x128 DVD-GAN model trained to generate 48 frames, while having computational requirements close to the 12/128x128 version.

Section . Finally, we ablate the main components of our model in Section .

For the rest of the section we denote video dimensions by their output resolution DxD and number of frames F in the format F/DxD.

Experimental Setting

Datasets We consider the Kinetics-600 (Kay et al. 2017; Carreira et al. 2018) and the UCF101 (Soomro, Zamir, and Shah 2012) dataset for class conditional video generation. Kinetics-600 is a large scale dataset of Youtube videos depicting 600 action classes. The videos are captured in the wild and exhibit lots of variability. The amount of videos available from Kinetics-600 is constantly changing as some videos become unavailable from streaming platforms. We use a version of the dataset collected on June 2018 with around 350K videos.

UCF101 collects around 13K videos with around 27 hours of video from 101 human action categories. Its videos have camera motion and cluttered backgrounds, and it is a common benchmark in the video generation community.

Additionally, we use the BDD100K dataset (Yu et al. 2018) for unconditional video generation. BDD100K contains 100k videos recorded from inside cars representing more than 1000 hours of driving under different conditions. We use the training set split of 70K videos to train our models.

Evaluation metrics Defining proper evaluation metrics for video generation is an open research problem. We use metrics inspired by the image generation literature and adapted to video. On Kinetics, we report three metrics: i) Inception Score (IS) given by an Inflated 3D Convnet (Carreira and Zisserman 2017) network trained on Kinetics-400, ii) Frechet Inception Distance on logits from the same I3D network, also known as Frechet Video Distance (FVD) (Unterthiner et al. 2018), and iii) Frechet Inception Distance on the last layer activations of an I3D network trained on Kinetics-600 (FID). On BDD100K we report FVD and FID as described before, but we omit IS scores as they are not applicable since there are no labelled classes. On UCF101 we report IS scores following the common setup in the literature.

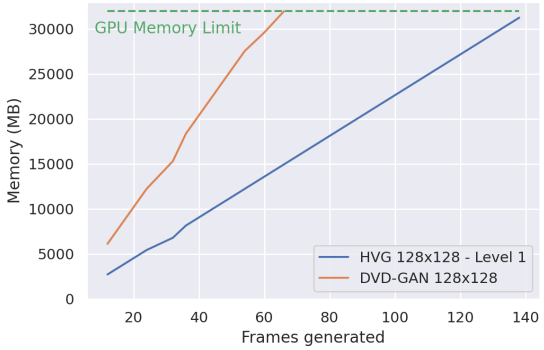

Figure 3: **Scaling the computational costs** This plot shows the required GPU memory for a two-level CVG. We observe that the costs scales linearly with the output length for the first level, while the cost for the second level is fixed because it operates on a fixed length partial view of its input. Our model scales better than a comparable non-hierarchical model.

Implementation details All CVG models are trained with a batch size of 512 examples and we fix a computational budget of up to 64 nVidia V100 GPUs. Levels for Kinetics-600 are trained for 300k iterations, while levels for BDD100K and UCF101 are trained for 100k iterations. We use early stopping if the metrics for a model stop improving. We use the PyTorch framework and distribute training across multiple machines using data-parallelism. We use global-synchronous batch norm to synchronize the batch norm statistics across workers.

We employ the Adam (Kingma and Ba 2014) optimizer to train all levels with a fixed learning rate of 1×10^{-4} for G and 5×10^{-4} for D. We use orthogonal matrices to initialize all the weights in our model as well as spectral norm in the generator and the discriminator. Further implementation details can be found in the Appendix.

Baselines As baselines we consider DVD-GAN (Clark, Donahue, and Simonyan 2019), TGANv2 (Saito, Matsumoto, and Saito 2017; Saito and Saito 2018), VideoGPT (Yan et al. 2021) and MoCoGAN (Tulyakov et al. 2018). DVD-GAN is the current state-of-the-art model for class-conditional video generation, being the first approach that can generate realistic samples on Kinetics-600.

Kinetics-600

We first evaluate the performance of CVG on the Kinetics-600 dataset with complex natural videos.

We train a two-level CVG on Kinetics-600 that generates either 12/128x128 or 48/128x128 videos, to compare to the previous state-of-the-art. The first level of CVG generates 24/32x32 videos, with a temporal subsampling of 4 frames. The second level upsamples the first level output using a factor of 2 for the temporal resolution and a factor 4 for the spatial resolution, producing 48/128x128 videos with a temporal subsampling of 2 frames. Since these generations are large, we employ partial views on first level outputs to train the second level. It takes as input windows of 6/32x32 frames and is trained to generate 12/128x128 video snippets, which are 4x lower dimensional than the final output. As a result, this level has approximately the same training cost than a model generating videos of size 12/128x128. At inference time we run the second level convolutionally over all the 24 first level frames to generate 128x128 videos with 48 frames. We also use the same model to generate 12/128x128 videos by using random 6 frame windows from the first level output (i.e. we use the same training and inference setup). We compare to the metrics reported in the original DVD-GAN paper for both 12/128x128 and 48/128x128 videos with temporal subsampling 2, as the current state-of-the-art in this dataset. Note that DVD-GAN outperforms by a large margin other approaches on Kinetics-600 such as (Weissenborn, Täckström, and Uszkoreit 2020).

We report the scores obtained by our model in Table 1. For 12/128x128 videos, our model achieves higher IS and comparable FID to DVD-GAN, validating that both models perform comparably when using a similar amount of computational resources. Additionally, CVG outperforms a 48/128x128 DVD-GAN model in FID score and reaches a similar IS score, despite only being trained on reduced views of the data. Qualitatively, the generations of both models are similar - they do not degrade noticeable in appearance through time although both have some temporal inconsistencies. For CVG temporal inconsistencies are usually the result of a bad generation from the first level. For DVD-GAN we hypothesize that most inconsistencies are due to the reduced temporal field of view of its discriminator, which is of less than 10 frames while the model has to generate 48 frames.

UCF101

We additionally confirm the results obtained by our approach on the smaller UCF101 dataset. We train a two-level CVG to generate 16/128x128 videos, as commonly done in the literature. The first level generates 8/64x64 videos with a temporal subsampling of 2 frames. The second level upscales the output of the first level to 16/128x128 videos, with no temporal subsampling. Since these are relatively small videos, we do not use temporal windows to train the second upscaling level.

We report the scores obtained by our model in Table 2. Our model obtains state-of-the-art results, outperforming previous approaches by a large margin. Qualitatively, CVG generates coherent samples with high fidelity details, with small temporal inconsistencies. Samples from our model for the UCF101 dataset can be found in the appendix.

Scaling up CVG for BDD100K

To show that our model scales well with the video dimensionality, we train a three-stage model on the BDD dataset to generate 48/256x256 videos. A training iteration for a batch with a single example of size 48/256x256 for a similarly sized model trained end-to-end requires more than 32GB of GPU memory. Such a model is therefore not trainable on most current GPUs, unless we use engineering techniques like gradient checkpointing or model parallelism which add a significant overhead to the training time. Instead, we can train a CVG with a batch size of 256 using 64 GPUs, fitting 4 examples per GPU.

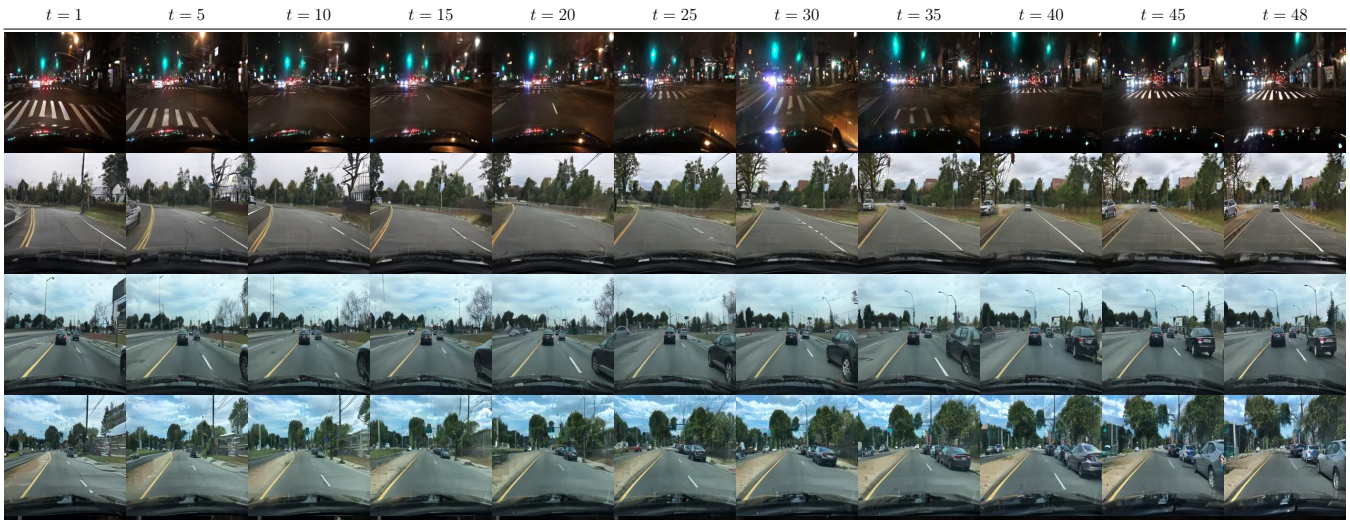

Figure 4: **Random 48/256x256 BDD100K samples:** We show samples from our three-stage BDD100K model. Each row shows a different sample over time. Despite the two stages of local upsampling, the frame quality does not degrade noticeably through time.

Model	IS(↑)
MoCoGAN (Tulyakov et al. 2018)	12.42
VideoGPT (Yan et al. 2021)	24.69
TGANv2 (Saito and Saito 2018)	28.87
DVD-GAN (Clark, Donahue, and Simonyan 2019)	32.97
CVG (ours)	**53.72**

Table 2: **UCF101 16/128x128 comparison:** We compare CVG to previous video generation approaches on UCF101. Our method obtains significantly higher Inception Score.

We train the first level to output 12 frames at 64x64 resolution with a temporal subsampling of 8 frames. The second level upsamples windows of 12 frames at 128x128 resolution with temporal subsampling of 4 frames (since we are doubling the framerate of the first level). The third level is trained to upscale 12 frame windows at 256x256 resolution for a final temporal subsampling of 2 frames. Figure 4 shows some samples from this model. We observe a variety of scenes with different times (day/night) and different number of cars and objects. The videos appear crisp and do not degrade through time. However, we also observe some small temporal inconsistencies and wrong scenes for some generations. More samples from this model as well as its IS and FVD scores can be found in the Appendix.

To further illustrate the scaling capabilities of our model, we show the memory requirements for a two-level CVG trained to generate 128x128 videos as a function of the number of output frames in Figure 3. The first level generates half the total output frames at 64x64, while the second level is trained to upscale windows of 6 frames into 12 128x128 frames regardless of the first level output length. Compared to a non-hierarchical model trained to generate 128x128 videos directly, our first level scales better thanks to the lower resolution and reduced number of frames. Additionally, the second level has a fixed memory cost of 10290MB and is trained independently of the first stage. Given the same GPU memory budget, our model can generate sequences of up to 140 frames, more than twice the number of frames compared to a non-hierarchical model. These computational costs are further increased when generating longer higher resolution videos with multiple upsampling stages. An additional comparison in running time can be found in the Appendix. costs Our model, despite the training of multiple stages, requires less training time due to the reduced costs for each stage.

Model Ablations

Matching Discriminator Ablation To assess the importance of the matching discriminator, we compare two-level CVG models with and without the matching discriminator (we refer to the latter as No MD). We use the same setup as in Section , where we generate 48 frames with a two-level model trained on Kinetics-600. We train the second level on 3-frame windows of the first level to generate 6 frames.

We expect the No MD model to fail at generating coherent full-length videos when applied over the full first level output since its outputs are not necessarily grounded on its inputs. On 6 frame generations (i.e. the. training setup), CVG and CVG No MD obtain similar scores as reported in Table 3. While the No MD model ignores its previous level inputs, it still learns to generate plausible 6 frame videos at 128x128. However, when we use the models to generate full-length 48 frame videos, we observe that CVG No MD generates valid local snippets but is inconsistent through time.

Fig. 5 shows an example of a full length No MD generation in which this effect is clearly observable. In contrast, our model with a matching discriminator stays grounded and consistent through time. This is reflected in the reported metrics in Table 3, where the No MD model obtains significantly worse scores. This ablation highlights the importance of using a matching discriminator, as it ensures that upsampling

Figure 5: **Matching discriminator samples** We show a random sample from our two-level model on Kinetics-600 with the matching discriminator and without the matching discriminator (No MD). For each sample we show the output of the first level and the corresponding second level output. While the (no MD) model generates plausible local snippets at level 2, it does not remain temporally coherent. Our model with the matching discriminator is temporally consistent because it is grounded in the low resolution input.

		6 Frames			50 Frames		
Dataset	Model	IS (↑)	FID (↓)	FVD (↓)	IS (↑)	FID (↓)	FVD (↓)
Kinetics-600	CVG (No MD)	**50.31**	1.62	594.99	37.81	42.29	1037.79
	CVG	48.44	**1.06**	**565.95**	**49.44**	**31.87**	**790.97**
BDD100K	CVG (No MD)	N/A	1.36	211.69	N/A	26.52	575.51
	CVG	N/A	**1.07**	**144.96**	N/A	**18.73**	**326.78**

Table 3: **Matching discriminator comparison** We report metrics on Kinetics and BDD for our model with and without the matching discriminator. Both models perform similarly for 6 frames, corresponding to the training video length. However, the model without the matching discriminator produces incoherent generations when applied over the full first level input because it can ignore it.

	Kinetics-600 48 Frames		
Window	IS (↑)	FID (↓)	FVD (↓)
3-frame	58.21	31.59	714.74
6-frame	**77.36**	**14.00**	**517.21**

Table 4: **Temporal Window Ablation** We analyze the impact of the temporal window for upsampling levels by comparing models trained with different window sizes. These two-level models are trained with 3-frame or 6-frame windows for the upsampling level. While the 6-frame model has higher computational requirements, it outperforms the 3-frame model, confirming that there is a trade-off between computational savings and final performance when selecting the size of the temporal window for upsampling stages.

level outputs are grounded to its inputs.

Temporal Window Ablation One of the modelling choices in CVG is the temporal window length used in the upsampling levels. Shorter inputs provide less context to upsample frames, while longer inputs require more compute. To assess the impact of the window length, we compare two-level models trained on Kinetics-600 128x128: one trained on first level windows of 6 frames (same setup as in Section) and one trained on windows of only 3 frames. The 6-frame level requires approximately 2x GPU memory than the 3-frame level during training, but we expect it to perform better due to the larger context available for upscaling. We compare their performance to generate 48 frames in Table 4.

We conclude that the window size defines a trade-off between computational resources - shorter temporal windows require less computation - and sample quality - longer windows have more context to fill in details when upsampling.

We refer the reader to the Appendix for an additional ablation on the choice of recurrent layers, an additional experiment comparing the power spectrum of our generations and the original Kinetics-600 data, an analysis of the influence of motion in our results, and additional samples.

Conclusions

We propose CVG, a hierarchical video generator that divides the generative process into multiple simpler steps. Our model is competitive with state-of-the-art approaches in terms of sample quality, while enabling higher resolutions generations for longer temporal horizons than possible before. Higher capacity models that produce larger outputs are key aspects for improving video generation, and CVG is a step in that direction that has better scaling properties than previous approaches.

References

Ayzel, G.; Scheffer, T.; and Heistermann, M. 2020. RainNet v1. 0: a convolutional neural network for radar-based precipitation nowcasting. *Geoscientific Model Development*, 13(6): 2631–2644.

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

Ballas, N.; Yao, L.; Pal, C.; and Courville, A. 2015. Delving deeper into convolutional networks for learning video representations. *arXiv preprint arXiv:1511.06432*.

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

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

Carreira, J.; Noland, E.; Banki-Horvath, A.; Hillier, C.; and Zisserman, A. 2018. A short note about kinetics-600. *arXiv preprint arXiv:1808.01340*.

Carreira, J.; and Zisserman, A. 2017. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6299–6308.

Castrejon, L.; Ballas, N.; and Courville, A. 2019. Improved conditional vrnns for video prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, 7608–7617.

Chung, J.; Kastner, K.; Dinh, L.; Goel, K.; Courville, A. C.; and Bengio, Y. 2015. A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*, 2980–2988.

Clark, A.; Donahue, J.; and Simonyan, K. 2019. Efficient video generation on complex datasets. *arXiv preprint arXiv:1907.06571*.

Denton, E.; and Fergus, R. 2018. Stochastic Video Generation with a Learned Prior. In *International Conference on Machine Learning*, 1182–1191.

Denton, E. L.; Chintala, S.; Fergus, R.; et al. 2015. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, 1486–1494.

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

Donahue, J.; Krähenbühl, P.; and Darrell, T. 2016. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*.

Dumoulin, V.; Belghazi, I.; Poole, B.; Mastropietro, O.; Lamb, A.; Arjovsky, M.; and Courville, A. 2016. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*.

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

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Kalchbrenner, N.; van den Oord, A.; Simonyan, K.; Danihelka, I.; Vinyals, O.; Graves, A.; and Kavukcuoglu, K. 2017. Video pixel networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 1771–1779. JMLR. org.

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

Kay, W.; Carreira, J.; Simonyan, K.; Zhang, B.; Hillier, C.; Vijayanarasimhan, S.; Viola, F.; Green, T.; Back, T.; Natsev, P.; et al. 2017. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, D. P.; and Dhariwal, P. 2018. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, 10215–10224.

Kingma, D. P.; and Welling, M. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Kumar, M.; Babaeizadeh, M.; Erhan, D.; Finn, C.; Levine, S.; Dinh, L.; and Kingma, D. 2019. Videoflow: A flow-based generative model for video. *arXiv preprint arXiv:1903.01434*, 2(5).

Larochelle, H.; and Murray, I. 2011. The neural autoregressive distribution estimator. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 29–37.

Larsen, A. B. L.; Sønderby, S. K.; Larochelle, H.; and Winther, O. 2015. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*.

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

Locher, P. J. 2010. How does a visual artist create an artwork. *The Cambridge handbook of creativity*, 131–144.

Luc, P.; Couprie, C.; Lecun, Y.; and Verbeek, J. 2018. Predicting future instance segmentation by forecasting convolutional features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 584–599.

Luc, P.; Neverova, N.; Couprie, C.; Verbeek, J.; and LeCun, Y. 2017. Predicting deeper into the future of semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, 648–657.

Mathieu, M.; Couprie, C.; and LeCun, Y. 2015. Deep multi-scale video prediction beyond mean square error. *arXiv preprint arXiv:1511.05440*.

Ranzato, M.; Szlam, A.; Bruna, J.; Mathieu, M.; Collobert, R.; and Chopra, S. 2014. Video (language) modeling: a baseline for generative models of natural videos. *arXiv preprint arXiv:1412.6604*.

Reed, S.; van den Oord, A.; Kalchbrenner, N.; Colmenarejo, S. G.; Wang, Z.; Chen, Y.; Belov, D.; and de Freitas, N. 2017. Parallel multiscale autoregressive density estimation. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2912–2921. JMLR. org.

Rezende, D. J.; and Mohamed, S. 2015. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*.

Rezende, D. J.; Mohamed, S.; and Wierstra, D. 2014. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*.

Saito, M.; Matsumoto, E.; and Saito, S. 2017. Temporal Generative Adversarial Nets with Singular Value Clipping. In *ICCV*.

Saito, M.; and Saito, S. 2018. TGANv2: Efficient training of large models for video generation with multiple subsampling layers. *arXiv preprint arXiv:1811.09245*.

Soomro, K.; Zamir, A. R.; and Shah, M. 2012. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*.

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

Srivastava, R. K.; Greff, K.; and Schmidhuber, J. 2015. Highway networks. *arXiv preprint arXiv:1505.00387*.

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

Unterthiner, T.; van Steenkiste, S.; Kurach, K.; Marinier, R.; Michalski, M.; and Gelly, S. 2018. Towards Accurate Generative Models of Video: A New Metric & Challenges. *arXiv preprint arXiv:1812.01717*.

Villegas, R.; Yang, J.; Hong, S.; Lin, X.; and Lee, H. 2017a. Decomposing motion and content for natural video sequence prediction. *arXiv preprint arXiv:1706.08033*.

Villegas, R.; Yang, J.; Zou, Y.; Sohn, S.; Lin, X.; and Lee, H. 2017b. Learning to generate long-term future via hierarchical prediction. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 3560–3569. JMLR. org.

Vondrick, C.; Pirsiavash, H.; and Torralba, A. 2016a. Anticipating visual representations from unlabeled video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 98–106.

Vondrick, C.; Pirsiavash, H.; and Torralba, A. 2016b. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, 613–621.

Weissenborn, D.; Täckström, O.; and Uszkoreit, J. 2020. Scaling autoregressive video models. *International Conference on Learning Representations*.

Xiong, W.; Luo, W.; Ma, L.; Liu, W.; and Luo, J. 2018. Learning to generate time-lapse videos using multi-stage dynamic generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2364–2373.

Xue, T.; Wu, J.; Bouman, K.; and Freeman, B. 2016. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *Advances in neural information processing systems*, 91–99.

Yan, W.; Zhang, Y.; Abbeel, P.; and Srinivas, A. 2021. VideoGPT: Video Generation using VQ-VAE and Transformers. *arXiv preprint arXiv:2104.10157*.

Yu, F.; Xian, W.; Chen, Y.; Liu, F.; Liao, M.; Madhavan, V.; and Darrell, T. 2018. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*.

Zhao, L.; Peng, X.; Tian, Y.; Kapadia, M.; and Metaxas, D. N. 2020. Towards Image-to-Video Translation: A Structure-Aware Approach via Multi-stage Generative Adversarial Networks. *International Journal of Computer Vision*.