Video Generation with Generative Adversarial Networks

Ian Pegg
ipegg@ucsd.edu

Matheus Gorski
mgorski@ucsd.edu

Abstract

Video generation is a rapidly growing field within deep learning. A highly accurate video generation model has potential applications ranging from robotics to film production. Such a model would contain a powerful semantic representation of objects, backgrounds, and scene dynamics. We build upon a generative model, iVGAN, by incorporating recently proposed improvements. We compare results for methods including spectral normalization, drift penalties, and modified gradient penalty losses (one-sided and zero-centered). We find that while spectral norm and drift penalties appear to offer benefits by controlling the scale of gradients, zero-centered gradient penalties yield the most realistic generated videos.

1 Introduction

Video generation is an emerging field of deep learning research with potential applications across robotics, video compression and media production. Video generation requires a complex semantic understanding of the scene under observation. To generate a video with convincing temporal dynamics, a model must contain a representation of dynamic elements like environment physics and human action. Such a representation has very important implications in the field of robotics for both control and estimation. Video generation could be revolutionary in the field of film production. A high-performing video generation model would have massive impacts across these fields.

This field has yet to see standout performances like those in image generation, so there is a large space available for theoretical and experimental exploration. The current state of the art models use convolutional neural networks and their variants in variational autoencoder or generative adversarial network (GAN) [1] architectures.

Inspired by the successes of the GAN framework on the task of image generation, many of the recent video generation models have been built on GAN. In particular, Wasserstein (WGAN) [2] has shown immense promise in image generation. WGAN is formulated around the idea of controlling GAN gradient penalties during training by using a loss function based on the earth-mover (EM) distance rather than the Jensen-Shannnon divergence proposed in the original GAN formulation. This helps with GAN stability, but does not fully address the issue of mode collapse, in which the generator can produce representations in only a small subset of the data space. The EM distance requires that a Lipschitz constraint be enforced for training to converge. WGAN proposed simple gradient clipping to enforce this constraint. Since then, several papers have suggested more principled methods to maintain the Lipschitz constraint. This paper explores some of these methods and empirically evaluates their utility using a combination of video distance metrics and human evaluation.

2 Related Work

GAN was first proposed by Goodfellow et. al. [1]. DCGAN [3] and WGAN [2] are two foundational improvements upon the vanilla GAN model. DCGAN, inspired by image classification successes,

incorporated deep convolutional (DC) neural networks into the GAN formulation. WGAN provided a long-awaited method to stably train generative adversarial networks.

Gulrajani et. al. proposed WGAN-GP [4], which addresses some of the issues from clipping the gradients in WGAN. WGAN-GP proposes to enforce the Lipschitz constraint by penalizing the norm of the gradient of the discriminator with respect to its input. This lead to high quality results and easy training with minimal hyper-parameter tuning. Petzka et. al. [5] propose a one-sided penalty, which builds upon WGAN-GP, only adding the gradient penalty term when the gradient norm is greater than one. Hung et. al. [6] propose to improve the generalizability (mode collapse) and stability of WGAN by using a zero-centered gradient penalty (0-GP). Miyato et. al. propose SN-GAN [7], and found that in the absence of batch normalization, spectral normalization improves upon gradient penalty regularization. Finally, Karras et. al [8] propose a drift penalty, which prevents destabilization as the discriminator loss drifts away from zero.

Some of the earlier forays into video generation that to this date have produced the best results, involve decomposing the background and foreground. The assumption here is that the background will be static and only objects in the foreground have a temporal dimension. These are inherently limited to static backgrounds and stationary cameras. Vondrick [9] approached this by using a two-stream GAN architecture where a spatio-temporal mask is used to separate the foreground and background. MoCoGAN [10] approached this by semantically dividing the content and motion of the video. They assumed a latent space of images forming a path in the temporal dimension, thus extending video generation to a sequence of arbitrary length. Despite its success and flexibility, MoCoGAN by relying on video decomposition, is limited to static cameras.

iVGAN [11] approaches the video generation problem without assumptions limiting the space of videos to those with a static background. In this view, however, a more powerful model must be used to learn a temporal representation of low-frequency dynamics that were filtered out by the scene dynamics approaches. iVGAN uses 3D convolutions to gain complete video representations in the temporal and spatial dimensions. TGAN [12] asserts that there is a fundamental difference between the temporal and spatial dimensions, and so a 3D convolution is not appropriate. The TGAN model consists of a separate *image generator* and *temporal generator*.

There are several proposed models that can produce realistic videos using motion transfer and conditional video generation [13, 14, 15]. However, our goal is to evaluate loss functions, and with this goal in mind, we will start with the most general baseline, iVGAN, and improve upon it using the loss variants discussed above.

3 Generative Models for Video

3.1 Generative Adversarial Networks

First introduced by Goodfellow et al. [1] in 2014, a GAN consists of two separate networks that are trained simultaneously in an adversarial manner as illustrated in figure 1. In the context of video generation, this consists of designing a generator network G, which generates a video x from some low-dimensional Gaussian noise z, and a discriminator/critic network D, which is trained to distinguish generator outputs x from real video samples. Successfully training a video generation model, therefore, requires each network to optimize competing objectives; D must accurately differentiate generated and real videos, while G must generate videos that cannot be distinguished from the real ones.

If we denote the parameters of G with θ_g, we can describe this mapping as $G(z;\ \theta_g) : \mathcal{Z} \to \mathcal{X}$, a differentiable function that maps samples of random variable Z from probability space $\mathcal{Z}$ to elements x in the video space $\mathcal{X}$ with the distribution:

$$p_g \coloneqq G(Z;\ \theta_g)$$

If we denote the parameters of D with θ_d, we can define the differentiable mapping $D(x;\ \theta_g) : \mathcal{X} \to [0, 1]$, corresponding to the probability that video x came from the data and not from p_g.

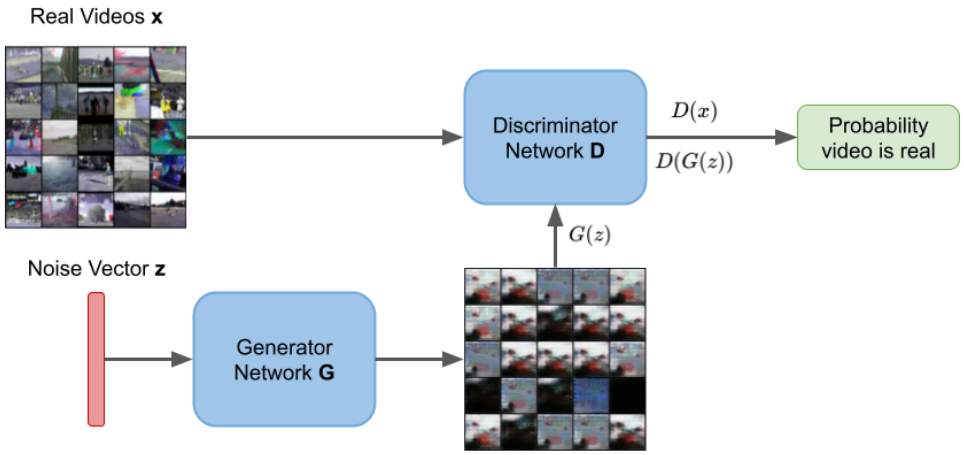

Figure 1: GAN training

D is trained to maximize the probability of correctly determining whether x came from p_{data} or p_g. Simultaneously G is trained to minimize the probability of D achieving this. The objective takes the following min-max form [1]:

$$\min_G \max_D V(D,G) = \min_G \max_D \mathbb{E}_{x\sim p_{data}(x)}[\ln D(x)] + \mathbb{E}_Z[\ln(1 - D(G(z)))] \tag{1}$$

This objective is solved computationally by alternating between the training of G and D, each optimizing, for a few epochs, their respective minimization and maximization components of the optimization problem.

Many training problems arise in practice. Training D, especially early on, is significantly easier than training G, which can quickly lead to situations where D is so saturated that it cannot provide G with information it can act upon. For this and many other reasons, GAN training on complex data becomes extremely difficult.

3.2 Wasserstein GAN

The Wasserstein GAN (WGAN) [2] is a training framework that addresses many of the problems present in the classical GAN formulation (mode collapse, training instability, etc.). GAN training, fundamentally, is about finding a generator G that matches the data distribution p_{data} when fed with random samples from Z. The WGAN framework introduces a new metric (Earth Mover (EM) Distance, or Wasserstein-1) for evaluating the distance between two probability distributions that, when used as the GAN training objective, significantly reduces the training difficulties of the classical GAN. The EM Distance for distributions p_{data} and p_g is defined as

$$W(p_{data},\ p_g) = \inf_{\gamma \in \prod(p_{data}, p_g)} \mathbb{E}_{(x,y)\sim\gamma}\left[\|x - y\|\right] \tag{2}$$

where $\prod(p_{data},\ p_g)$ denotes the set of all joint distributions $\gamma(x,y)$ whose marginals are p_{data} and p_g.

Due to the intractability of the infimum above, the distance is converted to an equivalent dual form:

$$W(p_{data},\ p_g) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x\sim p_{data}}[f(x)] - \mathbb{E}_{x\sim p_g}[f(x)] \tag{3}$$

where $\|f\| \leq 1$ indicates a 1-Lipschitz function (*i.e.* $\|\nabla_x f(x)\| \leq 1 \quad \forall x \in \mathcal{X}$)

If the supremum in Equation 3 is then parameterized with the terms of the original generator-discriminator formulation by modelling p_g as

$$p_g = G(Z;\ \theta_g) \tag{4}$$

where $Z = \mathcal{N}(0, I)$, and we replace the Lipschitz-1 constraint with Lipschitz-K for some constant K, we can consider solving the following problem:

$$\max_{\theta_d \in \mathcal{W}} \mathbb{E}_{x \sim p_{data}}[D(x;\ \theta_d)] - \mathbb{E}_Z[D(G(z;\ \theta_g);\ \theta_d)] \tag{5}$$

where $\mathcal{W}$ is the set of all parameters for D that ensure $\|D\|_L \leq K$ for some K.

If the supremum in Equation 3 is attained for some $\theta_d^* \in \mathcal{W}$, then by duality, it is equivalent to the solution of the minimization in Equation 2, which gives us the precise value of the EM Distance $W(p_{data},\ p_g)$ (multiplied by constant K). We can then differentiate $W(p_{data},\ p_g)$ (again, scaled by K) and use it to train our generator G with gradient descent:

$$\nabla_{\theta_g} W(p_{data},\ p_g) = -\mathbb{E}_Z[\nabla_{\theta_g} D(G(z;\ \theta_g);\ \theta_d^*] \tag{6}$$

We now should theoretically **fully saturate** the training of discriminator D (maximization in Problem 5) before switching back to training generator G. Using this method, the balance between generator and discriminator becomes much less delicate.

The designers of WGAN propose that the K-Lipschitz constraint be enforced through naively clipping the weights of D. The authors note that this often leads to unstable learning behavior.

3.3 WGAN-GP

Gulrajani et al. [4] introduce an important substitution to the weight-clipping approach posed in the original WGAN paper. To enforce the Lipschitz-K constraint on the discriminator D, a differentiable cost term is imposed on $\|\nabla_x D\|$. Since it is not feasible to determine the gradient magnitude of D for every possible $x \in \mathcal{X}$, an approximation of the video space distribution $p_{\hat{x}}$ is used. This consists of convex combinations of real-generated sample pairs, with combination coefficient α sampled from a uniform distribution for each combination:

$$p_{\hat{x}} = p_{data} + A\,(p_g - p_{data}) \tag{7}$$

where $A = U[0,\ 1]$

This gives us the new discriminator regularization term

$$\mathcal{L}_{GP} = \mathbb{E}_{\hat{x} \sim p_{\hat{x}}}\left[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2\right] \tag{8}$$

which is used to penalize the gradient magnitude's squared distance from 1. The authors of the paper mention the possibility of using a one-sided penalty, since the original Lipschitz constraint only requires that the gradient norm be *less than* 1, but they argue with that the optimal WGAN discriminator should have gradient norm of 1 at almost all points under p_{data} and p_g regardless [4].

3.4 Drift Penalty

Due to the lack of a saturating activation function on the discriminator network output in many WGAN GP formulations, the discriminator loss tends to drift significantly from zero. To prevent this critic loss drift from destabilizing training of the generator, Karras et al. [8] propose the inclusion of a small "drift penalty" term:

$$\mathcal{L}_{drift} = \mathbb{E}_{x \sim p_{data}}[D(x;\ \theta_d)^2] \tag{9}$$

The authors suggest that this greatly improves WGAN-GP training stability in its early stages by preventing an exponentially-increasing discriminator loss from providing unhelpful information when training the generator.

3.5 Zero-Centered Gradient Penalty

Thanh-Tung et al. suggest in [6] that the gradient penalty proposed by Gulrajani et al. (Equation 8) is inherently problematic for a variety of reasons. The original WGAN-GP's two-sided penalty centered at 1 forces the surface of the discriminator loss w.r.t. its inputs to never converge to a minimum. This, they argue, promotes gradient exploding in the discriminator, which leads to poor generalization and mode collapse in the generator network.

Thanh-Tung et al. instead suggest using a zero-centered gradient penalty. Theoretical analysis and empirical results in their paper suggest that this significantly improves training stability by addressing critic drift and by allowing the generator to actually converge. This term can be simply expressed as

$$\mathcal{L}_{GP0} = \mathbb{E}_{\hat{x} \sim p_{\hat{x}}} \left[\|\nabla_{\hat{x}} D(\hat{x})\|_2^2 \right] \tag{10}$$

In addition to their work supporting the benefits of using a zero-centered penalty, Thanh-Tung et al. also introduce a more sophisticated method of defining and sampling from $p_{\hat{x}}$. In the original WGAN-GP formulation, $\mathcal{L}_{GP}$ is calculated using convex combinations of real-generated sample pairs (Equation 7). Using points on lines between generated and real samples, Thanh-Tung et al. argue, does not actually yield elements of the target image space $\mathcal{X}$, since $\mathcal{X}$ is not a convex set (the average of two images is clearly very unlikely to be a "real" image).

Instead, the authors propose finding a path $\mathcal{C} : \mathbb{R} \rightarrow \mathcal{X}$ between real and generated samples $x \sim p_{data}$ and $\bar{x} \sim p_g$ by transforming the straight line segment connecting latent codes for x and $\bar{x}$ using generator G. Latent codes for real samples are generated by using an encoder E trained to map real images into a normal distribution, while latent codes for fake images are simply the same ones used to generate them. The "improved" estimate of $\mathcal{X}$ is therefore

$$p^*_{\hat{x}} = G\Big(E(p_{data}) + A(Z - E(p_{data}))\Big) \tag{11}$$

where $A = U[0,\ 1]$ and $Z = \mathcal{N}(0,\ I)$

Note that this method assumes that G is already trained well enough to generate samples that are already close to or in the target distribution $\mathcal{X}$.

3.6 One-Sided Gradient Penalty

A more obvious solution to the problems induced by the original two-sided gradient penalty would be the use of a one-sided constraint on the discriminator loss gradient's norm. Although the use of a one-sided gradient penalty in place of the original two-sided term is hinted at by Gulrajani et al. in the original WGAN-GP paper [4], Petzka et al. [5] perform an in-depth exploration of the variant and share results that suggest it leads to significantly improved training behavior. Named "WGAN-LP" for "Lipschitz Penalty", in contrast to the original "WGAN-GP", the one-sided penalty can be expressed as

$$\mathcal{L}_{LP} = \mathbb{E}_{\hat{x} \sim p_{\hat{x}}} \left[\max\{0,\ \|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1\}^2 \right] \tag{12}$$

The authors suggest that this modified gradient penalty is significantly less sensitive to choice of hyperparameter λ than the original method.

3.7 Spectral Normalization

An alternate method of enforcing the Lipschitz constraint of the WGAN formulation is through Spectral Normalization[7]. Proposed by Miyato et al., this method is a refinement of the Spectral Norm Regularization technique proposed by Yoshida et al. [16].

In loose terms, this technique is based on the fact that the Lipschitz constant of a composite of multiple functions (with bounded gradient magnitudes) is bounded by the product of each of their individual Lipschitz constants. In the context of Linear Algebra, this is the same as saying that the spectral norm of a series of matrix multiplications is bounded by the product of each of the individual matrices' spectral norms.

Since the discriminator network is a composite function of multiple functions (convolutions, non-linear activations, etc.) the Lipschitz constraint can therefore be enforced simply by normalizing the parameters of each of the network's layers so that they have Lipschitz norms of 1. Since convolutional layers can be expressed as matrix multiplications, there are many fast algorithms designed specifically for computing these (*i.e.* the largest singular value of a matrix).

Miyato et al. suggest using the "power iteration" method for finding layer spectral norms for significant computational improvement over the Singular Value Decomposition method.

4 Methods

Using the iVGAN video prediction framework proposed by Kratzwald et al. [11] as a baseline, we address some of its weaknesses and build upon its strengths by incorporating promising techniques and heuristics.

4.1 Datasets

Despite the increasing amount of research being done on the use of Generative Adversarial Networks (GANs) for video applications, there is still no clear consensus on which architectures perform best across datasets. Indeed, there are still no standard benchmark data sets for video synthesis and prediction like there are for image classification problems (e.g. MNIST [17], ImageNet).

We will be experimenting with and comparing results on both the UCF-101 [18] and TinyVideo [9] (made public with the paper by Vondrick et al.) video datasets.

UCF-101, an extension of the UCF50 dataset, consists of over 13,000 labeled videos depicting human actions from 101 different 'action classes'. These include categories such as 'skydiving', 'table tennis', 'playing flute', and 'skiing'. All videos have a fixed frame-rate and resolution of 25 FPS and 320×240. Clips have a variety of different lengths, but are clipped to 32 frames for our purposes.

The TinyVideo dataset consists of both labeled and unlabeled 64×64 32-frame videos. We use the 'beach' and 'golf' subsets of the overall dataset, which each consist of around a million videos depicting beach and golf scenes (See Figure 2). Note that TinyVideo golf and beach each contains a small but significant number of corrupted and off-topic videos.

4.2 Fréchet Video Distance

For comparing generated images, the Fréchet inception distance (FID) has become the defacto standard for evaluation. We will use its video analog, the Fréchet video distance (FVD) [19]. However, FVD is *not* a standard metric for video generation, which generally relies upon human evaluators to score generated videos. Indeed, we question the generalizability of the original FVD, trained on Kinetics 400 [20], because it was trained on human action videos only. So, due to the limitations of resources and metrics, we will present our FVD scores, but rely mostly upon the authors' subjective evaluation of generated video quality. We note that related works have relied upon Amazon Mechanical Turk to evaluate video quality.

(a) Golf

(b) Beach

Figure 2: Sample Frames from GIFs in TinyVideo Dataset

4.3 iVGAN Baseline

iVGAN, introduced by Kratzwald et al. [11], is a multi-functional GAN framework built on WGAN described in Section 3.2. The problem formulation, as in WGAN, involves minimizing the EM distance between the data and generated data distributions:

$$\min_{\theta_g} W(p_{data},\ p_g) = \min_{\theta_g} \left[\max_{\theta_d \in \mathcal{W}} \mathbb{E}_{x \sim p_{data}}[D(x;\ \theta_d)] - \mathbb{E}_Z[D(G(z;\ \theta_g);\ \theta_d)] \right] \tag{13}$$

Through incorporation of the WGAN improvements introduced in [4], iVGAN implements the standard WGAN-GP training formulation:

$$\min_{\theta_g} \max_{\theta_d} \mathbb{E}_{x \sim p_{data}}[D(x;\ \theta_d)] - \mathbb{E}_Z[D(G(z;\ \theta_g);\ \theta_d)] + \lambda \mathcal{L}_{GP} \tag{14}$$

or more concisely as

$$\min_{\theta_g} \max_{\theta_d} V(D, G) + \lambda \mathcal{L}_{GP} \tag{15}$$

where λ is a training hyperparameter used to balance the gradient penalty term defined in Equation 8. For all experiments, we set $\lambda = 10$ following [11].

The input of generator network G consists of a linear upsampling layer, which transforms the low-dimensional code vector into a high dimension, and a reshaping layer, which shapes the upsampled code noise vector into a tensor with very small spatial dimension but many channels (4×4 2-frame video with 512 channels). This is then passed through a series of 3D Transposed Convolutional blocks, ReLU activations, and batch normalization layers, transforming the initial tensor into a 64×64 32-frame video with 3 channels (RGB).

The discriminator network D is almost identical to the inverse of the generator network except for one important difference: since the gradient penalty $\mathcal{L}_{GP}$ is calculated with respect to *individual* samples $\hat{x} \sim p_{\hat{x}}$, batch normalization layers must be replaced with layer normalization. The final layer of D has no activation function (just a fully-connected linear layer). While the theoretical purpose of the discriminator is to classify images as fake or real, in practice its main purpose is to provide the generator with strong gradient information for its updates. A saturating activation such as the logit function, according to Kratzwald et al., would detract from this.

The two networks are optimized alternately using SGD with the Adam algorithm (first and second moment decay rates $\beta_1 = 0.5$ and $\beta_2 = 0.999$, initial learning rate $\eta_0 = 0.0001$). As recommended

(a) Generated golf

(b) Generated beach

Figure 3: Screen grabs of samples generated with baseline GP.

in [11], we perform 1 training iteration on the generator network G for every 5 iterations on the discriminator network D.

4.4 Variants

In addition to experiments using the baseline model and training specifications, we implement the problem variants discussed in Section 3.

Spectral Normalization (SN) is applied both with and without the original GP loss ($\mathcal{L}_{GP}$). When applied with GP, layer normalization layers are removed from the discriminator network after initial experiments suggested that training quickly collapses otherwise. SN layers are applied after every layer in the discriminator, including the final linear layer.

The drift penalty term defined in Section 3.4 is incorporated into the WGAN-GP training objective as

$$\min_{\theta_g} \max_{\theta_d} V(D, G) + \mathcal{L}_{GP} + \epsilon_{drift}\mathcal{L}_{drift} \tag{16}$$

where $\epsilon_{drift} = 0.0001$, as recommended in [8].

The two variants of GP loss described in Sections 3.5 ($\mathcal{L}_{GP0}$) and 3.6 ($\mathcal{L}_{LP}$) are implemented by substituting the respective variants of $\mathcal{L}_{GP}$ into Objective 15 and using the same penalty weight of $\lambda = 10$.

To maintain homogeneity of the results, we will use the hyper-parameters suggested by the authors of iVGAN.

5 Results

After some experimentation with UCF-101, we found that meaningful results could not be obtained with such a small dataset. The results shown here were generated using the TinyVideo golf and beach subsets. Figure 3 shows screen grabs of samples generated using the baseline GP model. For video samples, refer to the GitHub repository[1].

The FVD results for each of the investigated loss variants are shown in table 1. An individual FVD measurement uses only 16 samples, and so has very high variance. The reported numbers

[1] https://github.com/talcron/frame-prediction-pytorch/tree/media

	Golf		Beach	
Method	FVD	steps	FVD	steps
GP	2020	383,278	2076	114,783
SN-GP	2943	53,444	2838	115,105
SN	2345	375,576	2361	403,724
0-GP	**1938**	421,432	**1936**	61,901
1-sided	2097	147,262		
drift penalty	2212	78,678		

Table 1: FVD evaluated on the final trained models. The numbers show the average of FVD evaluated 100 times on 16 randomly selected data and generated samples. The number of training steps for each model is provided for context.

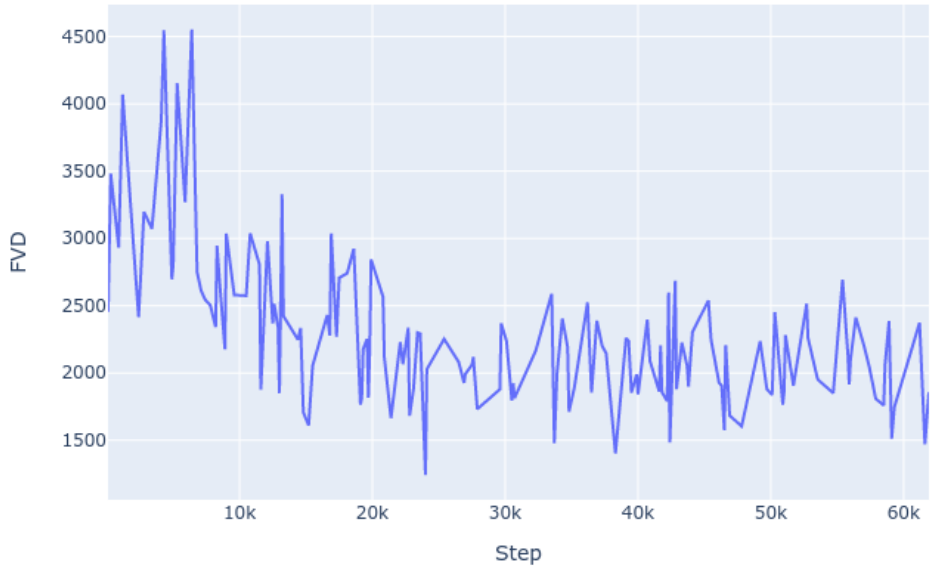

Figure 4: FVD during training using 0-GP on TinyVideo beach. These are single, 16-sample FVD measurements evaluated every 100 training steps.

are an average of 100 FVD measurements. Despite our reservations about the utility of FVD in this context, the FVD results correspond quite well with our qualitative evaluations of the generated data. Another indicator of quality is that FVD decreases with training, as can be seen in figure 4. These are single 16-sample FVD measurements at 100-step intervals.

We note the number of steps in table 1 for context. Not all models were trained for the same number of steps due to time constraints. One step represents one iteration of either generator *or* discriminator optimization. For all models other than SN, 43,000 steps represents about 24 hours of training on four GTX 1080 Ti GPUs. SN is about three times faster than this. Also due to time constraints, the 1-sided penalty and the drift penalty were not evaluated on the TinyVideo beach subset.

These experiments identified two standout performers, spectral norm combined with gradient penalty (SN-GP) and 0-GP. The former standing out for its poor performance, not achieving any-

thing resembling a real video after 100,000 iterations, and the latter standing out for its superior performance, even over fewer iterations.

6 Conclusion

We found that, while spectral norm does deliver on its promise of computational efficiency, it cannot produce results matching those obtained using the zero-centered gradient penalty, even when trained for longer. The zero-centered gradient achieves what it sets out to do, which is controlling gradients in a way that prevents mode collapse and maintains generalizability.

Despite achieving improvements upon our iVGAN baseline, it is clear that radical advancements are still necessary for general video generation to achieve results comparable to the more narrowly focused scene dynamics approaches like MoCoGAN. We suggest that future authors employ a 0-GP baseline, and explore the use of deeper, more powerful models that have demonstrated high performance in the image generation space.

References

[1] Ian Goodfellow et al. “Generative adversarial nets”. In: *Advances in neural information processing systems* 27 (2014), pp. 2672–2680.

[2] Martín Arjovsky, Soumith Chintala, and L. Bottou. “Wasserstein GAN”. In: *ArXiv* abs/1701.07875 (2017).

[3] Alec Radford, Luke Metz, and Soumith Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *arXiv preprint arXiv:1511.06434* (2015).

[4] Ishaan Gulrajani et al. “Improved Training of Wasserstein GANs”. In: *CoRR* abs/1704.00028 (2017). arXiv: 1704.00028. URL: http://arxiv.org/abs/1704.00028.

[5] H. Petzka, Asja Fischer, and D. Lukovnikov. “On the regularization of Wasserstein GANs”. In: *ArXiv* abs/1709.08894 (2018).

[6] Hoang Thanh-Tung, Truyen Tran, and Svetha Venkatesh. “Improving Generalization and Stability of Generative Adversarial Networks”. In: *CoRR* abs/1902.03984 (2019). arXiv: 1902.03984. URL: http://arxiv.org/abs/1902.03984.

[7] Takeru Miyato et al. “Spectral Normalization for Generative Adversarial Networks”. In: *CoRR* abs/1802.05957 (2018). arXiv: 1802.05957. URL: http://arxiv.org/abs/1802.05957.

[8] Tero Karras et al. “Progressive Growing of GANs for Improved Quality, Stability, and Variation”. In: *CoRR* abs/1710.10196 (2017). arXiv: 1710.10196. URL: http://arxiv.org/abs/1710.10196.

[9] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. “Generating Videos with Scene Dynamics”. In: *CoRR* abs/1609.02612 (2016). arXiv: 1609.02612. URL: http://arxiv.org/abs/1609.02612.

[10] Sergey Tulyakov et al. “Mocogan: Decomposing motion and content for video generation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 1526–1535.

[11] Bernhard Kratzwald et al. “Improving Video Generation for Multi-functional Applications”. In: *CoRR* abs/1711.11453 (2017). arXiv: 1711.11453. URL: http://arxiv.org/abs/1711.11453.

[12] Masaki Saito, Eiichi Matsumoto, and Shunta Saito. “Temporal generative adversarial nets with singular value clipping”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2830–2839.

[13] Yaohui Wang et al. “ImaGINator: Conditional spatio-temporal GAN for video generation”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2020, pp. 1160–1169.

[14] Aayush Bansal et al. “Recycle-gan: Unsupervised video retargeting”. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 119–135.

[15] Caroline Chan et al. “Everybody dance now”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 5933–5942.

[16] Yuichi Yoshida and Takeru Miyato. *Spectral Norm Regularization for Improving the Generalizability of Deep Learning*. 2017. arXiv: 1705.10941 [stat.ML].

[17] Y. Lecun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324. DOI: 10.1109/5.726791.

[18] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. “UCF101: A dataset of 101 human actions classes from videos in the wild”. In: *arXiv preprint arXiv:1212.0402* (2012).

[19] Thomas Unterthiner et al. “Towards Accurate Generative Models of Video: A New Metric & Challenges”. In: *CoRR* abs/1812.01717 (2018).

[20] Joao Carreira et al. “A short note about kinetics-600”. In: *arXiv preprint arXiv:1808.01340* (2018).