Everybody GANs Now!

CS 6670 Final Report

Amrit Amar (aa792), Anirudh Maddula (aam252), Alisha Mithal (am2658), Leul Tesfaye (lst26)

Abstract

The paper, *Generating Videos with Scene Dynamics [1],* proposed a Generative Adversarial Networks (GANs) for videos that untangle the scene's foreground from the background, also known as VideoGAN. The VideoGAN generated videos that, while not photo-realistic, display motions that were fairly reasonable for the scenes it was trained on. This itself was ground-breaking as the authors believe that generative video models can impact many applications in video understanding and simulation. Our goal in this project is to see if it is possible to *generate human poses* using GANs. Specifically, we explore the understanding human dancing, by training our models on the *Let's Dance* [2] Dataset. We evaluate how well VideoGAN is able to output dance video and we propose a new architecture called VectorGAN that takes the vector representations of people dancing and generates videos.

1. Introduction

There is a lot of potential in being able to generate raw videos, especially videos about human pose. Such a model could be extended to add animations to static images by training it to generate the future frames, allowing for automatic animation of caricatures. Of course, the future is uncertain, so the model rarely generates the "correct" future, but the authors of VideoGAN think the prediction has some plausibility. Models that generate videos from static images given a target label can be used in a wide variety of applications. The original paper behind VideoGAN worked on detailed environmental scenes, such as waves in the ocean, trains, and lots of people walking around an area. We hope to extend this to something as specific as one single person's skeletal pose.

The animation industry, in particular, could use VideoGAN to rig a model doing a range of motions without needing to do it by hand. This can be done simply by giving the model a starting pose and the motion to do. The 'futures' generated by this model could also be used in simulating various things, where a start state and a series of instructions could be passed to the model to predict the future.

The main limitations are shown in the VideoGANs paper. First of all, the generated videos are not very realistic and somewhat distorted and are of fairly low resolution. Second, the evaluation of the model is extremely difficult. The original paper actually used Mechanical Turk to evaluate "which video is more realistic?", which is a good crowdsourcing-based approach, but is not really a concrete evaluation of how good the model is. Third, the paper shows that it is difficult to generalize scenes and that for accurate results, different scene categories need their own specifically trained models. For example, to generate video for beach scenes, the GAN is trained mostly on videos of beaches and shorelines. This is called a Conditional GAN, as opposed to a generalized GAN. There is a tight coupling between training data and output, in that GANs don't generalize well to other types of scenes. However, this will hopefully not affect our project as we are using only skeletal poses on a black background. Fourth, the further future extrapolations don't line up very well with the first frame, which may be attributed to bottlenecks in the encoder section of the model. While our model will have the same encoder, we will also be including additional information about the dance style. We hope this will help align the first frame and subsequent generated frames.

Pose detection (from the *Let's Dance* paper) also has its own limitations. Some styles of dance are hard to track because of all the limbs overlapping (ex. breakdancing) or having multiple people in the scene, so the pose estimator can't really identify the skeleton. Another issue is with motion blur, so frames without concrete structures can't detect limbs. In both cases, limbs will just disappear from frames in the pose estimator.

Overall, there's been no prior work that we found specifically in generating dancing motion, as the paper only covered general scene motions. We do not have any reference metrics that we can directly compare and evaluate. Evaluation of the generated videos is highly subjective and currently, the only option is to use MTurk to analyze which resulting videos are the most realistic.

For this paper, we choose to evaluate our generated videos subjectively, and explore a variety of architectures to produce the best possible video. In the end, we propose a brand new VectorGAN architecture that produces the most realistic videos of human pose. We do this by taking in a vector of size 34 and generate 32x34 frame image that encodes joint positions at each frame.

2. Approach

a. Let's Dance Dataset

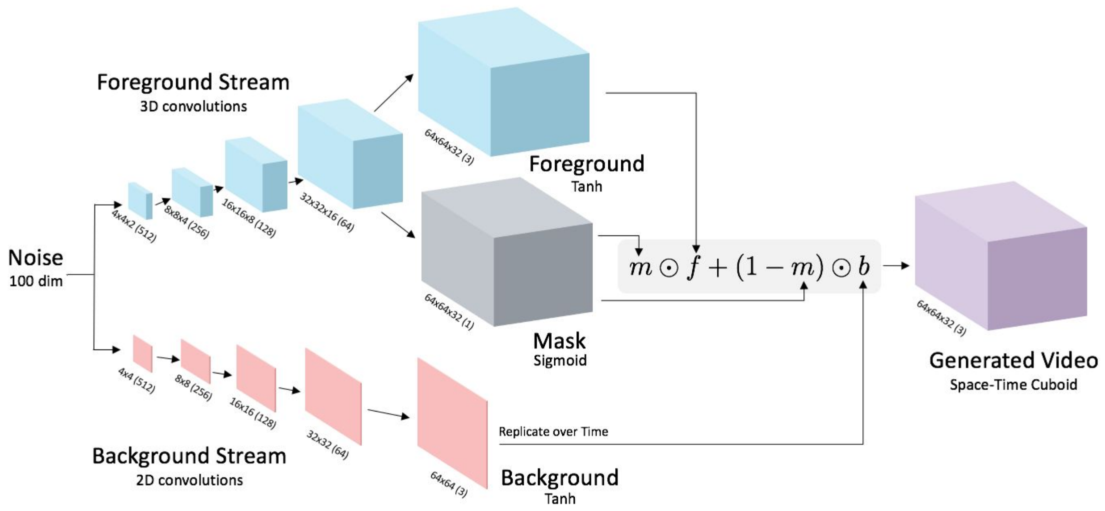
We will use the *Let's Dance (Skeletal Pose)* data set. The data set contains several pre-labeled clips of people dancing along with associated skeletal pose of the people in videos. The skeletal pose system used by the paper uses bounding boxes to identify humans in each frame, and then constructs the pose from them. The generated skeletal pose images have no background, thus allowing us to bypass categorizing scene backgrounds, letting us simply train on motions rather than the image setting itself. This dataset also comes with the vector data of all the joints of skeletons (i.e legs, knees, arms, head, etc...) for every frame.

Let's Dance Dataset comes with 10 dance types including hip-hop, salsa, and others. We train specifically on Ballet dancing. We feel the other dance types would impede the model's sense of action, considering there are very few frames to execute a clear action in the first place. For example, hip-hop has a lot of limb occlusions due to quick, tight forms and spinning.

b. VideoGAN

We trained the VideoGAN based on the labeled data set of skeletal videos, and VectorGAN on the skeletal vector data. A Generative Adversarial Network (GAN) uses a Generator to make images/videos that try and trick a Discriminator which tries to tell if videos are real or fake. The Generator eventually learns to create content that is indistinguishable from the actual training data. We will use the same GAN architecture from *Scene Dynamics* [1], which is a modified version of a Deep Convolutional Generative Adversarial Network (DCGAN) [3] and modify it to suit vector data as well.

VideoGAN expands on DCGAN by modifying the generator to separate the foreground and background of the video, allowing us to focus on a moving foreground over a stabilized background image. It also allows for multiple approaches for generating new video, including training on different categories of images and having the starting image generated from noise vs a realistic frame (shown below) [1]. There is also an extension to this which places an encoder in front of the generator which takes in frames. Thus, the generator can start from a predefined skeletal image instead of trying to converge from noise. We used both versions of the model in this project.

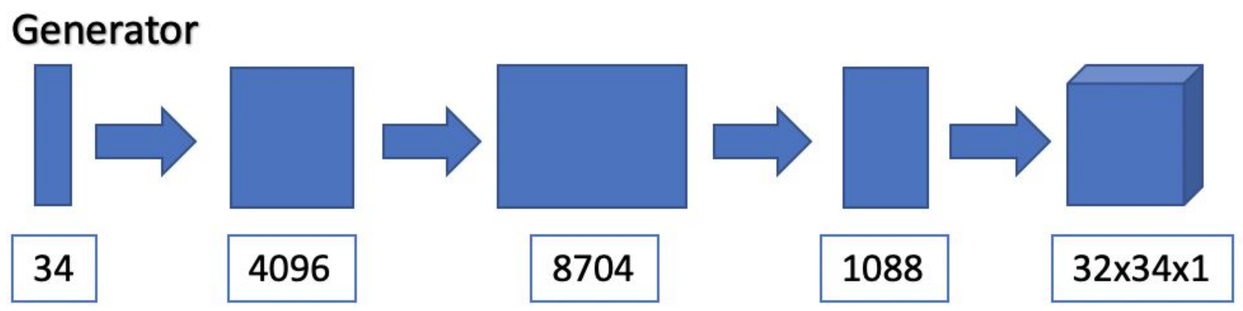


The original code for VideoGAN was written using Lua and Torch7, which is no longer supported. We updated the code use Keras/Tensorflow 2.0 to enable GPU support.

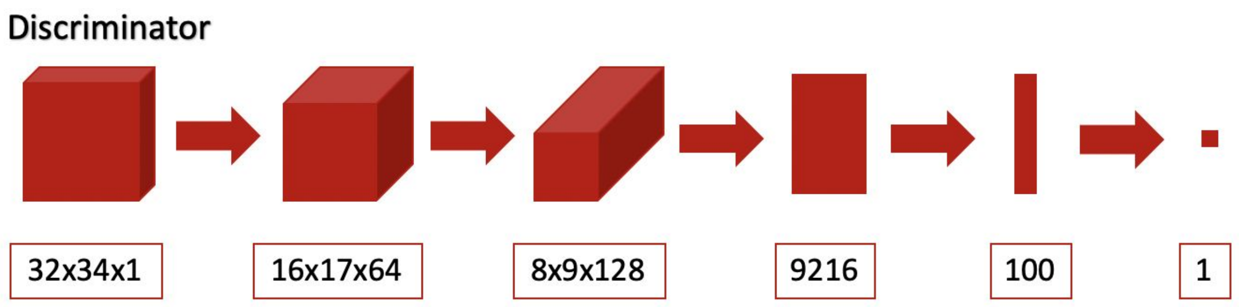
c. VectorGAN

The VectorGAN model does not need to worry about foreground or background streams, as we have direct access to vector data that configures each skeletal pose. In addition, VideoGAN has a lot of issues maintaining the limbs of the skeleton, whereas now we can just draw limbs clearly between keypoints. We propose a GAN that encodes this vector data into “images” to teach the model where and how to move joints around.

To enable this we built a generator that takes in a starting pose or random noise vector of 34 length and outputs an image. We went through several iterations of this generator model but we noticed that any type of convolution or batch normalization layer in our generator model ended up generating random poses. The best result we got was when we simply had fully connected layers and applied tanh activation to the last layer. The tanh normalization allows our model to simply focus on capturing the biological motion rather than having to learn how to scale biological motion relative to the training image size. When we first tried to do it without adding the tanh layer our generator loss constantly grew with each epoch and the generated videos looked like random noise. To fix this we added sigmoid activation layer but we noticed that our model seemed to only be stuck in the lower right quadrant of our image. This was fixed by using a tanh layer instead of sigmoid.



The output of our generator is a 32x34 image. We then treat each row as pose at a frame n. And draw the pose on a blank canvas and save it as a gif file. Initially we had the problem that our discriminator was failing to converge, in other words the discriminator loss grew with each epoch. To solve this we added 100 fully connected layer before connecting to the final layer with 1 as the output. This solved our divergence problem and ended up making our model significantly better.



During our initial model design phase we tested several loss metrics but in the end stuck with binary cross entropy loss and adam optimizer with learning rate 1e-4.

All of the following models are trained to be what is called ConditionalGANs. This means that each network is trained on an individual dance category in hopes of better results on that dance type. In this paper, we trained the models on the *Ballet* category to generate ballet movements.

Evaluation is very difficult for this problem. The original VideoGAN paper used Amazon Mechanical Turk (MTurk) to run psychoanalysis exams to see which generated videos seemed more realistic. Due to time constraints, we will rely on subjective visual evaluation. We will attempt to at least show the qualitative differences between videos generated from VideoGAN and VectorGAN. We show that there are significant improvements over a baseline generative model.

3. Experiments

We propose the following different experiments:

Experiment Architecture	**Expected Output**
VideoGAN Architecture	32 Frames of Video
Modified VideoGAN Architecture	128 Frames of Video
Encoder-VideoGAN Architecture	32 Frames of Video
VectorGAN Architecture with Random Noise	32 Frames of Video
VectorGAN Architecture with Starting Vector Image	32 Frames of Video
VectorGAN Architecture Composited Video (using previous experiment)	128 Frames of Video

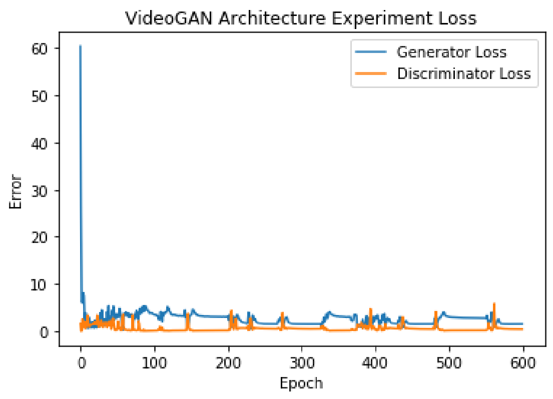
Under each experiment, we train a model built with the specified architecture and evaluate the output and the loss. We used Google Cloud Platform (GCP) to train the models for long periods of time.

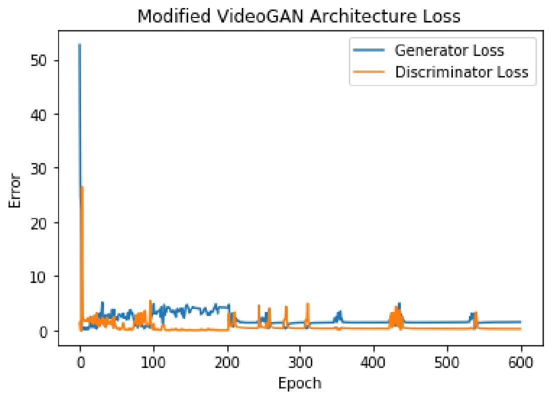
We first tested the VideoGAN architecture from [1]. We input 32 frame videos. Overall, from the output video, this model does well in making a figure of a human doing a dance move. However, it is very noisy. We expect this from a model that takes in raw pixels. This also happened in the second experiment where, even though the generated dance video was longer, it was still blurry. However, VideoGAN recognized people and captured some of their movements.

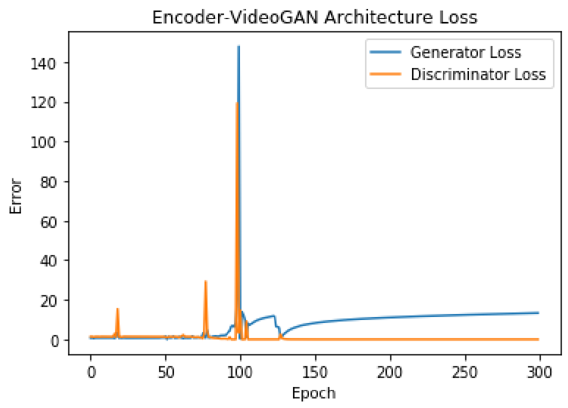
We then tested VideoGAN with the encoder extension. However, we had complications with encoding the video and was not able to produce meaningful results here, so we do not display them.

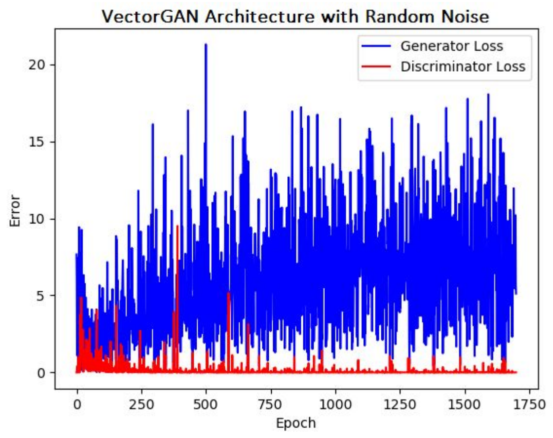
We then tested our own model, VectorGAN. VectorGAN had to be trained on more epoch than before experiments to get something smooth but this method worked pretty well. The generated vector positions through time resemble a lot of the motion that ballet dancers do. It is interesting to note that the model figured out how the different limbs of humans connect to each other after about 100 epochs but needed much more to generate coherent movement. There are still a lot of places where the dancer’s limbs jerk, but we attribute this to the shortcomings of the dataset we used; not every single frame had a coherent human structure and some frames had nothing. We then tested the VectorGAN with an encoder added on to the beginning of the architecture that had the starting vectors of the input video. We then used this in the *composited video experiment*, where we generated 128 frame videos by feeding the last frame of a generated 32 frame video into the model 3 times. Our loss and results shown with images is shown below:

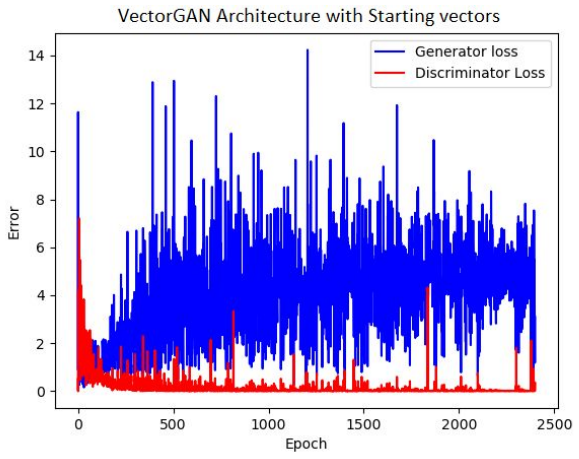
Experiment Loss Graphs

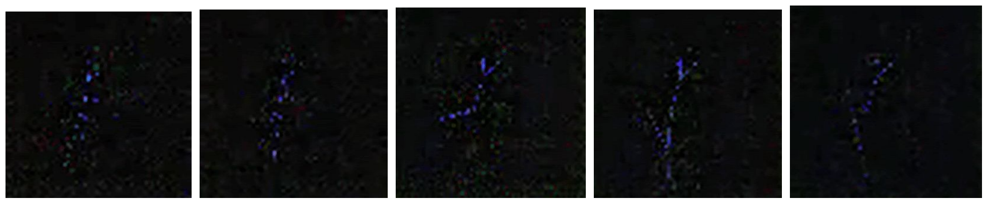
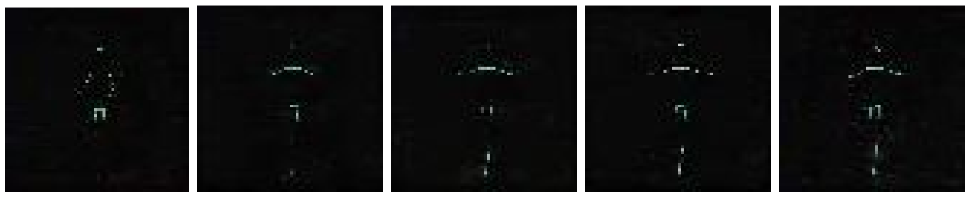
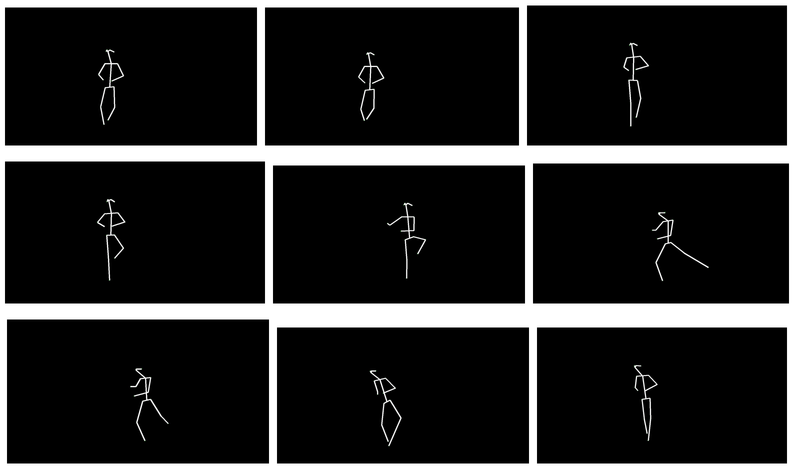
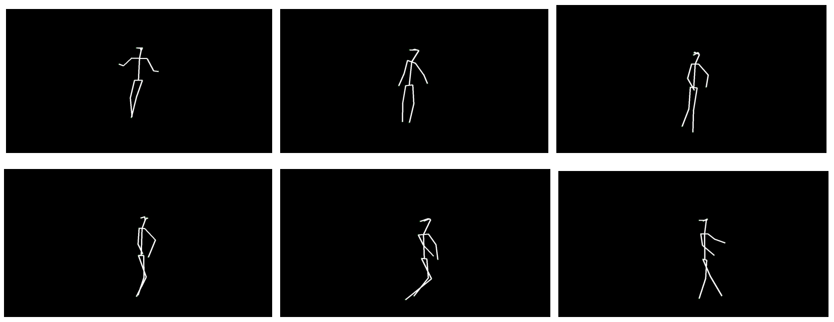


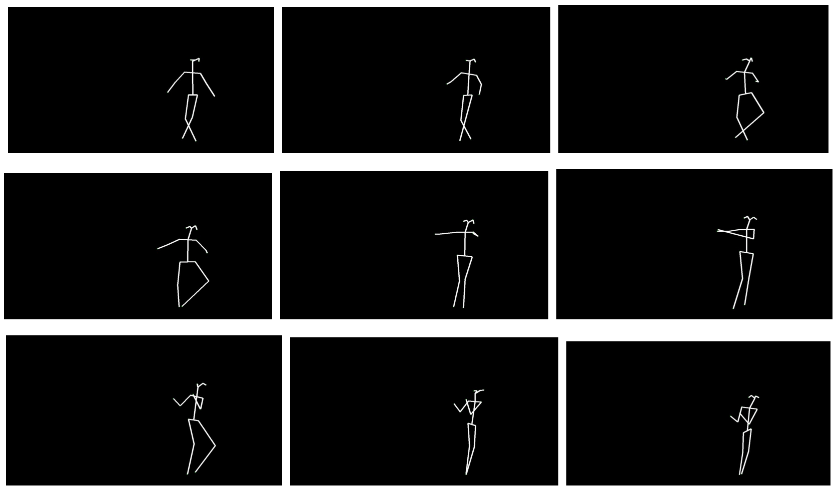








Experiment	Image Results
VideoGAN Architecture Experiment	 *We can see a person lifting their left leg up doing a classic ballet move.*
Modified VideoGAN Architecture	 *A person raises their arms to their sides*
VectorGAN Architecture with Random Noise	 *A person doing the classic ballet 'kick' to the side*
VectorGAN Architecture with Starting vectors	 *A person putting their hands together while walking*

VectorGAN Architecture Composited Video	 *A person jumping up and then coming down putting their hands together*

4. Conclusion and Future Work

VideoGAN and VectorGAN provide tangible results in generating videos of human pose. However, the frames generated VideoGAN were of very low resolution as computing images of stick figures is still very difficult. VectorGAN had much better results as we could animate together distinct key points to see the figure move more clearly. Our conclusion is that generating human pose is possible with VectorGAN, but we don't have nearly enough training data. The original VideoGAN paper used over 35 million clips while we used only 84 videos for training our models. However, the results we got are promising.

It is possible to extend this project to make an improved visualization by taking the skeleton pose output and rigging that with a body and creating an animation sequence. This would not only help the viewer recognize what's happening in the clip but it would also help us gauge the performance of our model. Since our model was simply trained with (x, y) skeleton positions of the performer, it's current output is for visualization purposes only. However given (x, y, z) joints positions of a performer, this model can be quickly re-adjusted to output an image with two channels that encodes in x,y,z position.

There is also potential in generalizing the model on all types of dancing instead of training Conditional GANs on one dance (in this paper, *Ballet*). This would involve using the original videos, not the pose estimations. We would attempt to specify a dance type such that the model knows what types of motion to execute. Furthermore, it would be interesting if the model could infer what type of dance to execute based on the characteristics and context of the input image(s), and generate respective video.

5. References

[1] Vondrick, Carl, et al. *Generating Videos with Scene Dynamics*, 2016.
http://www.cs.columbia.edu/~vondrick/tinyvideo/paper.pdf

[2] Castro, Daniel, et al. *Let's Dance: Learning From Online Dance Videos*, 23 Jan. 2018.
https://www.cc.gatech.edu/cpl/projects/dance/.

[3] Radford, Alec, et al. *Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks*, 7 Jan. 2016.

[4] Cao, Zhe, et al. *OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields,* 30 May. 2019.