HAVS
Human Action Video Synthesis

Mohith L M, Krishna Vamsi G, Sirshapan M, Khoushik C, Sanjana A, Nikhil P

December 7, 2022

Abstract

As our world moves more towards Machine Learning and Artificial Intelligence applications, we ask whether Artificial Intelligence can be used to generate art. Contemporary models, such as Dall-E, and Make-a-Video, are making incredible progress in generative modeling. We explore such models using our minuscule knowledge in this field. We first try to generate an auto-encoder of our own to create videos according to the problem. We then explore and try to implement more advanced models. Finally, we use the video-GPT model to create class-conditioned and frame-conditioned videos.

1 Introduction

Deep generative models of multiple types have seen incredible progress in the last few years across multiple modalities, including natural images. There are several applications for video generation, such as robotics, augmented reality, data augmentation, and action imitation. Various variations include predicting video scenes, synthesizing videos, interpolating videos, and super-resolving videos. Video modeling has certainly advanced, but generative modeling of high-fidelity natural videos has yet to catch up with the development of images, audio, and text. As natural videos are complex, the input dimensions for modeling correlations are much larger in space and time.

As a result, deep generative models are well suited to modeling video. One reason for the relatively slow progress in the generative modeling of videos can also be attributed to the complexity of the problem. When combined with deep neural networks, these models can result in the generation of increasingly photorealistic images/videos. This work aims to generate human actions based on images of actors and targets.

2 Method

Our aim is to generate a video with 16 frames x1,x2,...,x16 showing how the action will be performed by giving an input image x0 of an actor and an action class ya. To address this, we are using our own model with the help of Auto Encoders and VideoGPT.

The action dynamics module creates latent action representations, while the video synthesis module creates action videos v by combining the appearance and produced motion properties.

Generative models: Generative models describe how a dataset is generated in a logistic way.

Auto Encoders: A bottleneck architecture known as an autoencoder transforms a high-dimensional input into a latent low-dimensional code (encoder) then reconstructs the input using the latent code (the decoder).

GAN: A new method of deep learning called Generative Adversarial Networks (GAN) was introduced by Goodfellow. From its name, GAN is a generative model that uses a deep neural network in an adversarial setting. Specifically, GANs learn the generative model of data distribution by using adversarial techniques. Artificial intelligence research has been focusing on GAN as one of the most successful generative models in recent years. Since it was originally proposed, GAN has received great attention because of its excellent performance. The GAN can not only be used as a generative model that performs exceptionally well, but it also creates a new perspective on deep learning, opening up new research areas and applications. GANs work by training generators and discriminators in an adversarial manner. The target goal depends on the project requirements, such as a stronger generator or a more sensitive discriminator.

VideoGPT: Over the last few years, deep generative models have made significant progress in a wide variety of fields, including the generation of natural images, audio waveforms based on language features, natural language text, and music. VideoGPT is a conceptually straightforward approach to scaling likelihood-based generative models to natural videos. By employing 3D convolutions and axial self-attention, VideoGPT learns downsampled discrete latent representations of a raw video. Using spatio-temporal position encodings, a simple GPTlike architecture is then used to autoregressively model the discrete latents. On the UTD-MHAD dataset, our architecture generates samples competitive with state-of-the-art GAN models despite simplicity in formulation and ease of training. VQ-VAE and GPT are adapted to create VideoGPT2, a simple video production architecture. A downsampled set of discrete latents is learned from the raw pixels of video frames using 3D convolutions and transposed convolutions in VideoGPT.

3 Experiments

3.1 Dataset

UTD MHAD: Microsoft Kinect and a wearable inertial sensor were used to gather the UTD-MHAD dataset inside. The dataset consists of 8 subjects' 27 actions (4 females and 4 males). Each participant performed each action four times. The dataset contains 861 data sequences after three corrupted sequences are removed. In three channels or threads, four data modalities—RGB films, depth videos, skeleton joint locations, and inertial sensor signals—were recorded. One channel was used to simultaneously record skeleton positions and depth movies, as well as RGB videos and inertial sensor inputs (3-axis acceleration and 3-axis rotation signals). Each sample was given a time stamp for the purpose of data synchronization.

Train - Test - Validation split of UTD MHAD dataset							
Action	Train sub	Test sub	Val sub	Action	Train sub	Test sub	Val sub
1	1,2,3	4,6,7,8	5	15	3,5,6	1,4,7,8	2
2	1,2,4	3,6,7,8	5	16	4,1,5	2,6,7,8	3
3	1,3,4	2,6,7,8	5	17	4,1,6	2,5,7,8	4
4	5,2,3	1,6,7,8	4	18	4,5,6	1,3,7,8	2
5	5,2,4	1,6,7,8	3	19	3,4,5,6	1,7,8	2
6	5,3,4	1,6,7,8	2	20	2,4,5,6	1,7,8	3
7	6,2,3	1,5,7,8	4	21	2,3,5,6	1,7,8	4
8	6,2,4	1,5,7,8	3	22	2,3,4,6	1,7,8	5
9	6,3,4	1,5,7,8	2	23	2,3,4,5	1,7,8	6
10	2,1,5	3,6,7,8	4	24	1,4,5,6	2,7,8	3
11	2,1,6	3,5,7,8	4	25	1,3,5,6	2,7,8	4
12	2,5,6	1,4,7,8	3	26	1,3,4,6	2,7,8	5
13	3,1,5	2,6,7,8	4	27	1,3,4,5	2,7,8	6
14	3,1,6	2,5,7,8	4				

3.2 AutoEncoder

In our CAP 5415 course, we were taught about the autoencoder as a generative model. Our initial approach tried to apply the autoencoder model for our task. Modeling an encoder and decoder of our own that would give good performance is challenging work, so we took reference from "github.com/cc-hpc-itwm/UpConv". This repo provides an architecture for the convolution-based generator model that they used for their GAN model. We modified the same architecture for the decoder of our autoencoder. Then we reversed the decoder portion to create an encoder architecture.

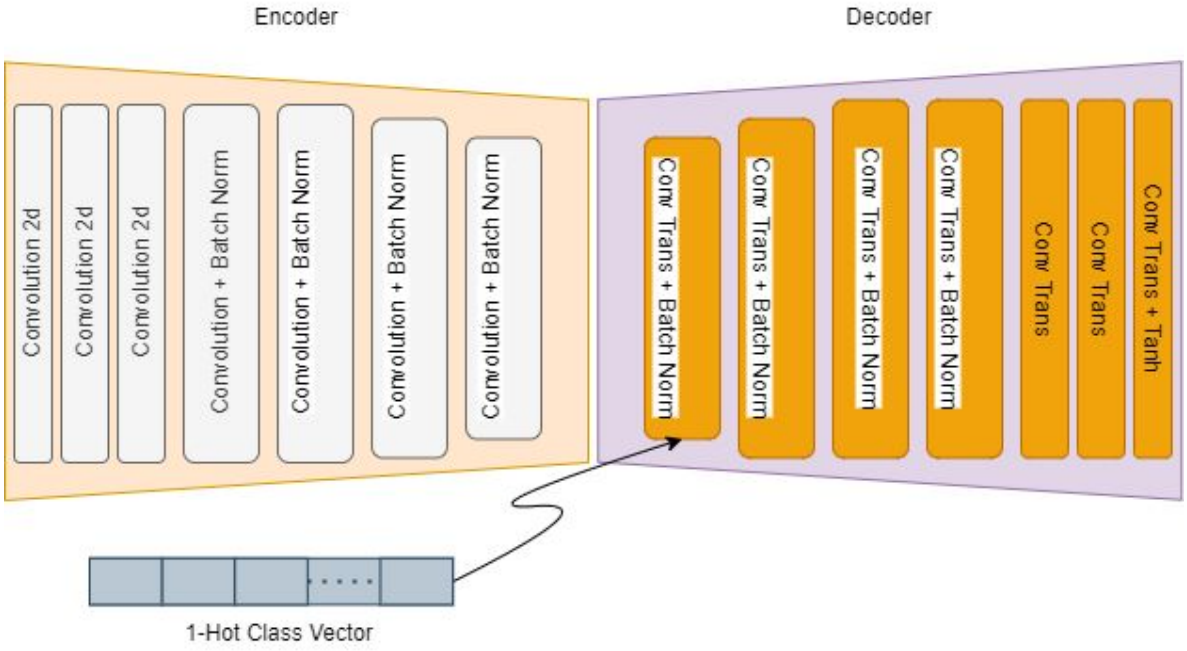

Figure I: Our Custom Auto Encoder Model

Architecture: We used encoder-decoder architecture for the auto-encoder model.

We used an input image of size 16x16 for our model. The encoder consists of 7 Convolution layers, to create a feature vector of size 64. Then we combine this feature vector with a class vector of size 27 (since there are 27 classes). We concat and pass this through a simple Multi-Layer Perceptron layer to generate a vector of size 256. Finally, we pass this vector of size 256 to the decoder to generate samples.

Training: Training was done using Google Colab. We trained the model for 30 epochs. After 25 epochs we saw no change in the loss, thus we stopped the training. The model was not able to produce any substantial result. The output was mostly noise. We tried a second experiment with only grayscale images/videos. The model was still unable to produce any proper results.

Discussion: We tried this ambitious approach even though we knew the approach was far too simple to perform such a complicated task. But we also understood, the complexity of the task.

3.3 Video-GPT

Simple models such as autoencoders, variational-autoencoders, would not be able to perform the complicated task that we aim. We took a look at other state-of-the art yet computationally effective generative models. Out of multiple models that we tried to implement, such as C-GAN, Moco-GAN etc, we were successfully able to use the Video-GPT model for our task.

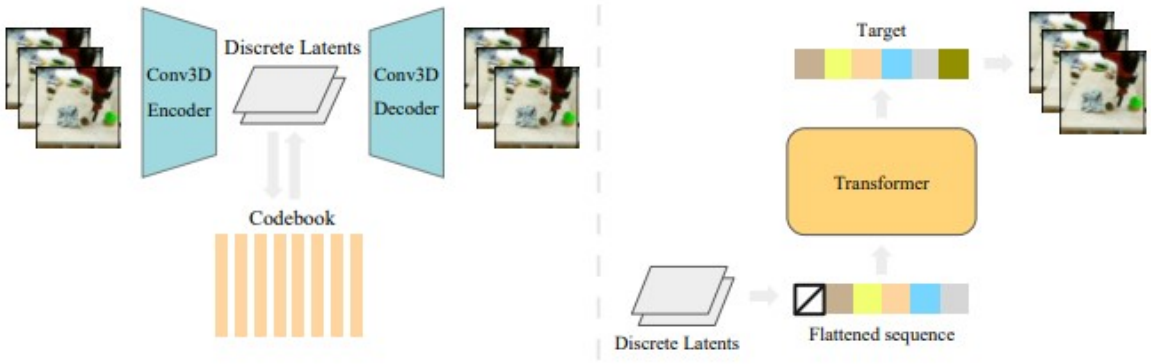

Figure II: Video-GPT architecture

Video-GPT is a conceptually simple architecture for scaling likelihood based generative modeling to natural videos. VideoGPT uses VQVAE that learns downsampled discrete latent representations of a raw video by employing 3D convolutions and axial self-attention. Video-GPT was tested on BAIR Robot Pushing and UCF-101 datasets. We tried to train and test the model on UTD-MHAD dataset.

Training: We needed to transform the UTD-MHAD dataset according to the format required by the Video-GPT code. We used the BAIR pretrained weights with 1 frame conditioned and class conditioned settings. We also used mixed precision settings, batch size of 16 and 1 gpu. We trained for 10660 iterations, after which the loss was not decreasing, so we stopped the training. Finally, we created our own script to visualize the results for UTD-MHAD dataset.

Result: We generate 16 frames, conditioning on a single frame and the class of the action

Figure III: Sample of frames generated by Video-GPT

Discussion: We could generate some realistic videos of UTD-MHAD dataset using the video-gpt model. We used the fvd score to evaluate the model. We got a fvd score of 394.11. This may be because of low amount of data the we used to train, or training for more epochs would have helped the score.

4 Conclusion

Although we were able to manage to run our model successfully, we were not able to train and test different models multiple times as there were limitations on time and resources. In the beginning we used the autoencoder model but could not get the

required results as the autoencoder model is far too simple to work with the dataset. In Order to overcome this issue we searched for various models to find a solution for our problem statement such as Conditional GANs and MOCO GANs. These other models were either too difficult to implement or required a lot of resources to run or both. But after further exploration we were finally able to run our model using VideoGPT.

Despite the fact that VideoGPT requires a decent amount of resources to run a model, we ran the model for 10660 iterations which provided us with an FVD score of 394.11. We think that this is a satisfactory result for our model based on the constraints and lack of resources. As the accuracy for a model is directly proportional to the total number of iterations the model has run through, we strongly believe that running the model for far more iterations could increase the FVD score and improve the accuracy of the model. It is to be noted that the code for the model was almost completely written by us except the training phase of the VideoGPT model as the architecture is too complex for our understanding. For future work we propose to use different diffusion models that can create much more realistic output.

5 Contribution

UTD-MHAD was chosen as the dataset for training the model considering it's size and ease of availability. This data is passed to the dataloader using the following transformations:

- *Resize*: Converts the data size to (16 x 16)
- *Tensor*: Converts the data into a tensor type
- *Normalise*: Values (0.13, 0.30) were used to normalise the images.

The dataloader outputs the following according to the given batch size:

- First frame of the whole video
- The whole video after applying the transformations mentioned above
- 1-hot encoding of that action class