CSCI 1430 Final Project Report: Everybody Dance Now!

Team Dancing Queen: Ryan Greenblatt, Neev Parikh, Jason Senthil, Nimish Garg.
Brown University
8th May 2019

Abstract

This project involved reimplementing the 2018 paper Everybody Dance Now [3] and replicating the results. We implement motion transfer from a source video to that of a target video by extracting motion information as pose and using a Generative Adversarial Network (GAN) to create the frames of the target subject in the pose of the source. To ensure video quality, Chan et. al. utilizes temporal smoothing, where the GAN is conditioned on the previous frame and is tasked with maintaining temporal coherence, as well as pose normalization, to ensure that different camera positions and difference in heights minimized impact on the final result. Due to the vagueness of implementation in their work (missing descriptions of variables and details), we test various methods of implementation and present our results.

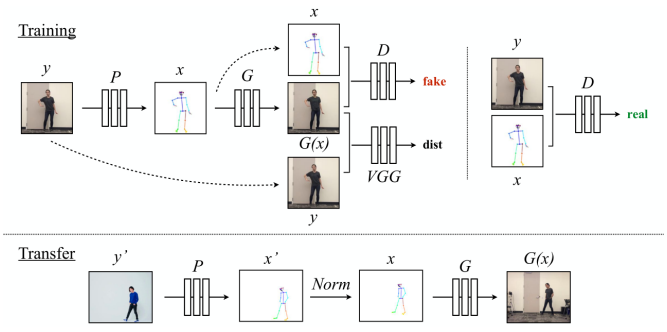
1. Introduction

Let us suppose we have two videos. Video A uses a stationary camera that is recording a good dancer dancing for some time. Video B also uses a stationary camera, but is recording a different person just showing his/her different poses to the camera over some period of time. The Everybody Dance Now paper proposes a method to transfer the dancing motion from the person in Video A to the person in Video B. This proposition has 3 main steps. The first pipeline needed is pose extraction. This is used on the individual frames of both videos. For Video A all we need is to extract and save the pose information (for each frame in the video) from the good dancer. For Video B, we also need extract pose information. The next step is to train a GAN. A GAN has two parts: a generator and a discriminator. The generator creates new data instances, while the discriminator checks the authenticity of whether it looks good enough to be part of the training set. With generators and discriminators going head to head, GANs are able to generate human like sophisticated images, and much more. In this case, the GAN takes in an image of Person B's pose and outputs a corresponding frame. This is trained on the Video B, since we want to know what Person B looks like given any pose, and a GAN can draw us a good representation of this after significant training. Finally, we send a dance pose frame from Person A to the GAN, and output Person B doing that pose. There is an intermediary step required, since Person A and Person B might have completely different pose proportions (longer limbs, different heights, etc), so pose normalization is used to convert Person's A pose into Person's B pose proportionally. Then the GAN is run on each of those frames and outputs frames for the video where Person B is dancing just like Person A.

2. Related Work

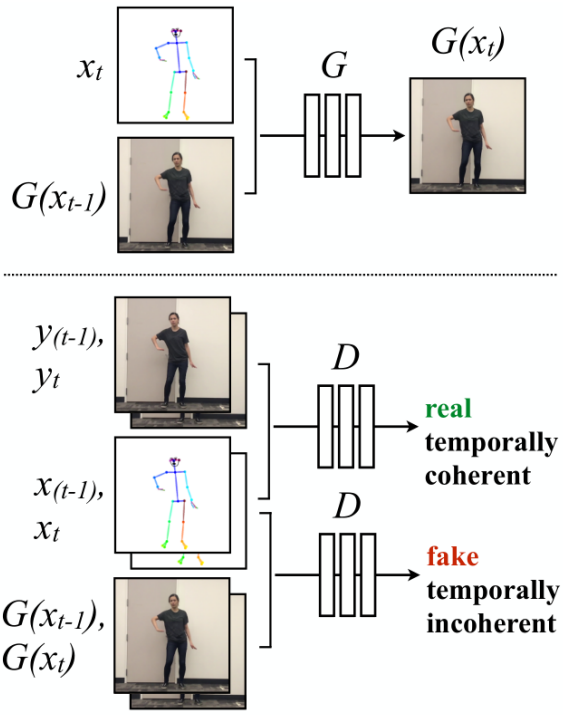
Motion transfer has seen active research in the past year or two, with work in pose estimation [4, 1, 2, 9, 12] and work in generating plausible images with GANs [6, 11, 10]. In this report, pix2pixHD [11] is used as a framework for the GAN architecture and Openpose [1] is used as the pose estimator, specifically the Pytorch implementation provided by [5]. The entire implementation is built on of the underlying PyTorch framework [8]. Other implementations of the same paper are incomplete and tend not to have pose normalization or temporal smoothing working; however, they were still helpful in interpreting the paper [7] [13].

3. Method

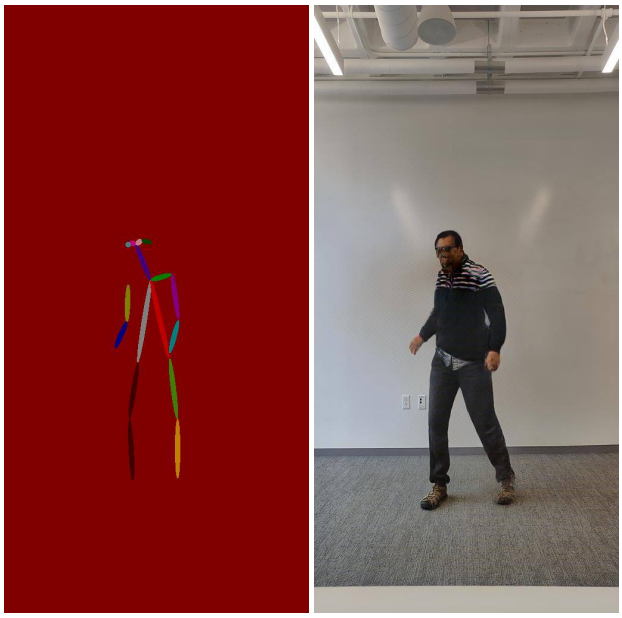


The method closely follows the method in Chan et. al.

(shown above, figure from [3]). However, certain aspects of their method are vague and ill defined, which forced us to test out different ideas at various stages. The training pipeline involves extracting the poses of the frames from the target using the pretrained pose estimator [5] and passing that as a label mask with each body segment labeled $l \in \{1, 2, 3 \dots k\}$ where k is the number of keypoints in the pose and 0 for the black background. To implement temporal smoothing (see below, figure from [3]), in addition to the label mask, the previous generated frame from the GAN was passed in, concatenated in the channel dimension. The idea behind this, as described by Chan et. al., is to condition the GAN based on the previous frame, so as to introduce smoothness into the resulting video. In the zeroth iteration, the GAN has no previous output. Different methods were tested and using an average frame worked best. If the GAN is presented with a black frame as it's previous frame during training, it leads to artificating during inference. However, even with the average frame, artificating still occurred, and we decided to split the dataset into chunks to increase the frequency at which the GAN observed the average frame. A parameter was introduced which replaced the previous generated frame with the average frame during any training step with probability p. We tuned this parameter in between epochs, supervising it to nudge the model towards not persisting artifacts. Training requires a lot of compute, and we went with GCP nodes with 4 Tesla V100s, training in a distributed manner.



Once the GAN is trained, we are almost ready to generate the video. We go through the dancer's video frame by frame, extracting pose information from each frame. This is then fed into a tuned pose normalizer. Then we send that pose data directly to the GAN along with the average frame as per temporal smoothing, which will draw the other person in the pose based on the trained mapping.



(a) Pose inputted to GAN (b) Final Output from Generator

Figure 1: GAN results

4. Results

4.1. Discussion

This method works surprisingly well, but not perfectly. GAN artifacts (darkish gray blobs not part of the image) can be seen at times, and could never fully be fixed (although the result is pretty good on downsampled images). The general algorithm design itself raises interesting questions in solving other complex problems (computer vision related or not). This works by essentially abstracting the relevant information needed (in this case a pose) and working from there. This is similar to how the Microsoft Kinect detected pose by using a infrared camera to abstract information like shape while removing relevant information like color and texture on the image. Also this raises questions about the computation power needed to do further complicated vision tasks like motion transfer, since cloud computation was required to train the GAN model. The face is not as accurate as we wanted, so future implementations could use a GAN specific for the face (as suggested and implemented in the original paper).

5. Conclusion

The power of deep learning for computer vision applications is incredible. We have shown a practical use of CNNs and GANs in developing motion transfer between two subjects in two separate stable videos, and all we need are just two video files to run this process from begin to end. Abstracting parts like pose is essential for training a GAN, and this concept of abstracting useful parts may be useful in other applications or problems where GANs may be relevant.

References

[1] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh. OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. In *arXiv preprint arXiv:1812.08008*, 2018. 1

[2] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017. 1

[3] C. Chan, S. Ginosar, T. Zhou, and A. A. Efros. Everybody dance now. 2018. 1, 2

[4] R. A. Güler, N. Neverova, and I. Kokkinos. Densepose: Dense human pose estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7297–7306, 2018. 1

[5] Hzzone. Pytorch-openpose implementation. https://github.com/Hzzone/pytorch-openpose, 2018. 1, 2

[6] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arxiv*, 2016. 1

[7] Nyoki. Pytorch-openpose implementation. https://github.com/nyoki-mtl/pytorch-EverybodyDanceNow/, 2018. 1

[8] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017. 1

[9] T. Simon, H. Joo, I. Matthews, and Y. Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *CVPR*, 2017. 1

[10] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, G. Liu, A. Tao, J. Kautz, and B. Catanzaro. Video-to-video synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 1

[11] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 1

[12] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *CVPR*, 2016. 1

[13] Yanx27. Pytorch-openpose implementation. https://github.com/CUHKSZ-TQL/EverybodyDanceNow_reproduce_pytorch, 2018. 1

Appendix

Team contributions

Please describe in one paragraph per team member what each of you contributed to the project.

Jason Senthil He was responsible for finding and implementing the pose estimation pipeline for the project. He used a Pytorch version of OpenPose found online and modified it to work well with the project. He then assisted in the pose normalization algorithm that was needed when transferring pose between two people with very different body measurements.

Neev Parikh He was primarily responsible for implementing the pose normalization algorithm for the project. He also assisted in coding the GAN required for the pose to image pipeline, and setup the GCP instance for the cloud compute required for the project.

Ryan Greenblatt He was responsible for coding the GAN (pix2pix) and tuning the parameters such that the project would see the best results (and much more). He did the Docker for the GCP.

Nimish Garg He developed an algorithm to remove frames which had invalid poses. Produced a testing pipeline for the GAN. Led the presentation efforts through development of the presentation.