Cindy Zhang, XinciZ

Michell Li, MLi5

Karan Rathour, KRathour

**Introduction to AI**

**“CMU Dances Now”**

**INTRODUCTION**

This project explores the 2D human-to-human motion transfer process without the need for expensive 3D motion capture data. In other words, given a video of a person dancing (the “Source”), our project intends to transfer the motion to an amateur dancer (the “Target”) through pose detection and image-to-image translation.

Currently, the majority of video data exists in the absence of 3D information. Making the technology of 2D motion transfer mainstream lends itself to enhancing new industries, and also calls to question many ethical questions about surveillance. Human to human motion transfer can impact the animation and film industry, due to its relatively inexpensive reproduction capabilities.

Furthermore, pose detection has implications that range from aiding autonomous vehicles responding to policemen directing traffic, to elevating AR/VR development. Detecting poses enables a computational perspective on human body language; an example of which is advertising billboards assessing if bystanders react positively or negatively when seeing the ad in real time or playing a video game with a digital replica of each player.

Nonetheless, human motion transfer is still under research and this project demonstrated that the process is not stylistically robust, hinting that movements could be unique to an individual’s identity. Further research now has the ability to quantitatively uncover that nuance of physical movement at a relatively lower cost and in real time.

**PREVIOUS WORK**

This project was first completed by Berkeley AI Lab in August 2018:

[Berkeley Everybody Dance Now](https://carolineec.github.io/everybody_dance_now/)

Following the guides below, this project reproduced the result under Pytorch environment.

[Part Affinity](https://github.com/NiteshBharadwaj/part-affinity)

[Multi-Person Pose Estimation](https://github.com/ZheC/Realtime_Multi-Person_Pose_Estimation#training)

[Everybody Dance Now Reproduction](https://github.com/CUHKSZ-TQL/EverybodyDanceNow_reproduce_pytorch)

**MODEL REVIEW**

***Data Sources***

The project calls for 2 types of data:

1. Video data of the Target and the Source:

* Target: A video of a teammate moving in front of a relatively monotonous background. 125 frames were extracted from the video as the base target data.
* Source: Mars, Bruno. “Bruno Mars - That's What I Like (Official Video).” (YouTube, 1 Mar. 2017, www.youtube.com/watch?v=PMivT7MJ41M). The dataset is ideal for this project for its simple background. Using 24 fps, 1000 frames were extracted from the video as the base source data.

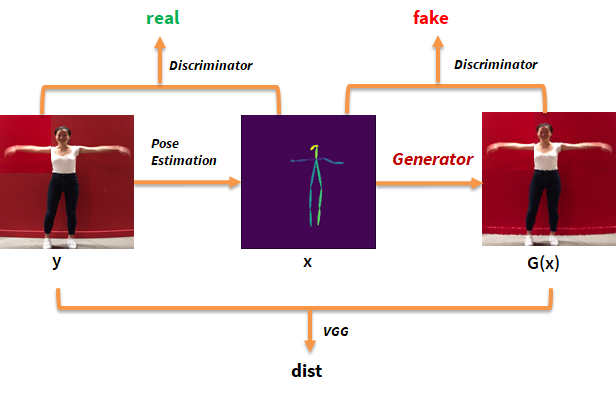
Both the target and source datasets were resized to be 512x512 pixels. Further data engineering was performed after the training of pose estimation model in order to generate more focused frames of the poses. In addition, headshots of each frame were identified by the pose estimation model for potential face enhancements, although they were not in the scope of this project at this time.

In both videos, subjects may have different limb proportions or stand closer or farther to the camera than one another. Therefore, when transferring motion between two subjects, it may be necessary to transform the pose key points of the source person so that they appear in accordance with the target person’s body shape and proportion. We find this transformation by analyzing the heights and ankle positions for poses of each subject and use a linear mapping between the closest and farthest ankle positions in both videos. After gathering these statistics we calculate the scale and translation for each frame based on its corresponding pose detection.

2. Training data for the Post Estimation model: COCOdatasets.org (2017 Training, Validation, and Keypoint annotations). COCO datasets include 40,000 training and 5,000 test images, in addition to 135 body part annotations. The body part annotations aid us in identifying limbs and joints and ultimately creating a skeleton representation of humans in an image.

***Network Architecture Overview - Training***

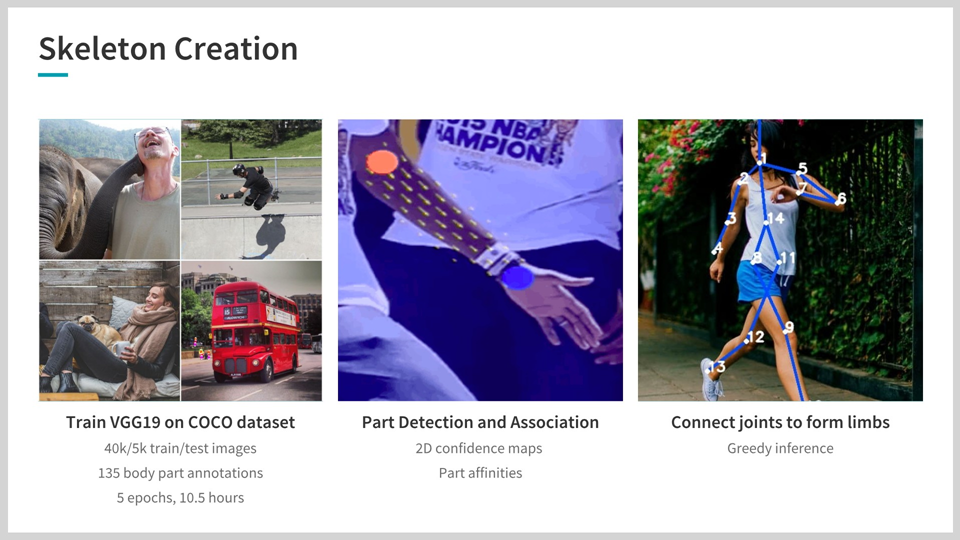
As demonstrated in Figure 1, the training process involves two major steps: 1) Pose estimation; 2) Pix2PixHD image translation.



*Figure 1. Architecture Overview of the Training Process*

1. **Pose Estimation**

During training, the ground truth confidence maps S is generated from given annotated 2D body part key points from COCO datasets. A confidence map constitutes the belief that a particular body part exists at a particular pixel location. Each image is analyzed by the first 10 layers of VGG-19 to predict a set of 2D confidence maps S of body part locations as well as a set of 2D vector fields L of part affinities, which encode the degree of association between body parts. The confidence maps and part affinity fields are parsed using greedy inference, which outputs the skeleton for each person in an image.



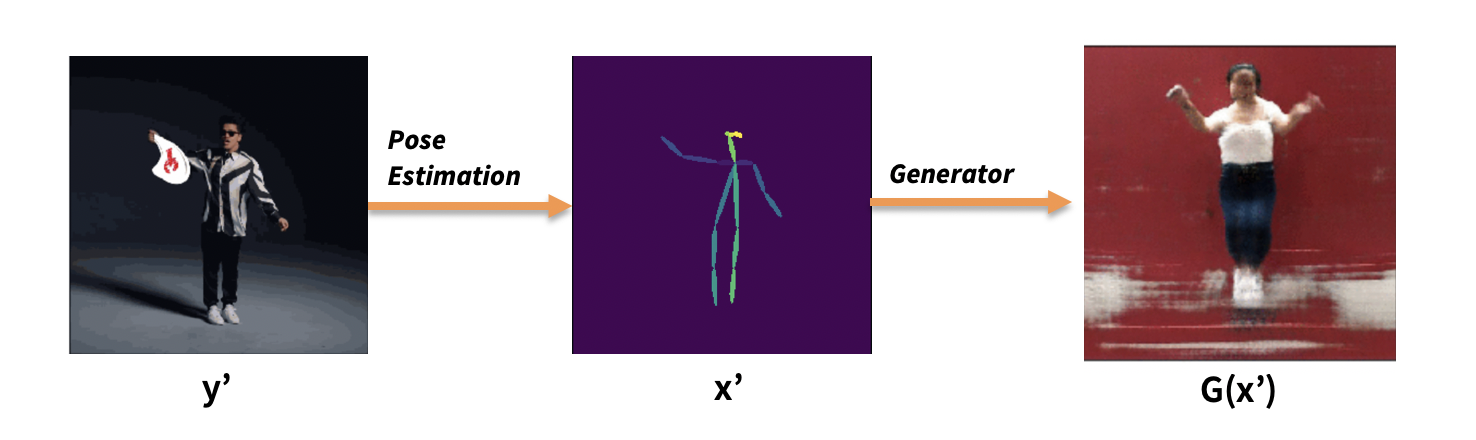
1. **pix2pixHD[[1]](#footnote-1)**

pix2pix is an adversarial training of image-to-image translation process. In the original conditional GAN setup, the generator network G is engaged in a minimax game against multi-scale discriminators D = (D1, D2, D3). The generator’s task is to synthesize realistic images in order to fool the discriminator which must discern between “real" (ground truth data) images from the “fake” images produced by the generator.

These two networks are trained simultaneously and drive each other to improve, as the generator must learn to synthesize more realistic images to deceive the discriminator which in turn learns differences between generator outputs and ground truth data.

***Network Architecture Overview - Transfer***

As demonstrated in Figure 2, the transfer process involves abstracting the Target into normalized poses using Pose Estimation and apply the generator G to create a synthesized Source sharing the same moves as the Target.



*Figure 2. Architecture Overview of the Transferring Process*

**RESULTS**

The project relies on two methods of performance evaluation. First, we examined the loss data collected as we increased the number of epochs of training the generator G and the discriminator D. As Figure 3 reviews, the losses slightly decreased over time, and were the lowest at our last epoch 200. Note that there is fluctuation of the loss values and the decreases were not significant.



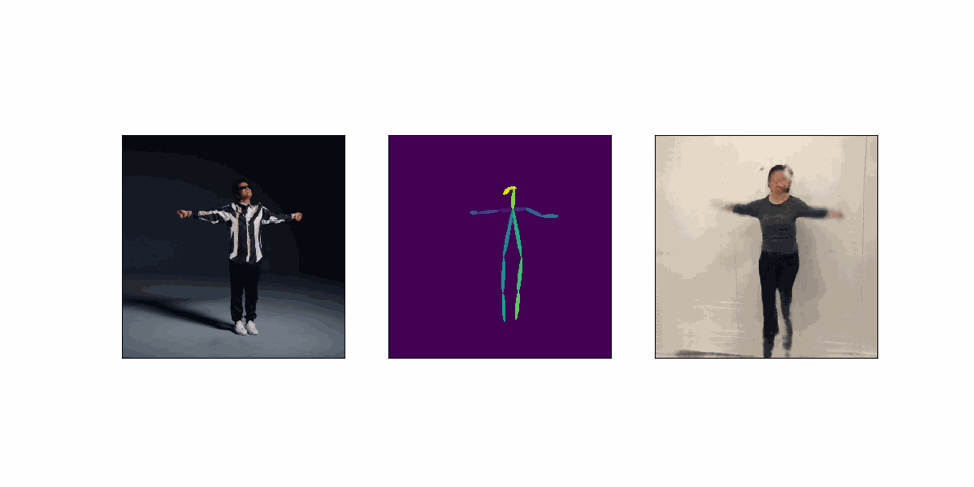
*Figure 3. Losses of the Generator G and the Discriminator D, by Training Epochs*

The second source of performance evaluation is a visual inspection of the images synthesized by the generator G. It is obvious from the photos in Figure 4 that the generator’s ability to transform a labeled pose into a picture of the Target has significantly improved over time.

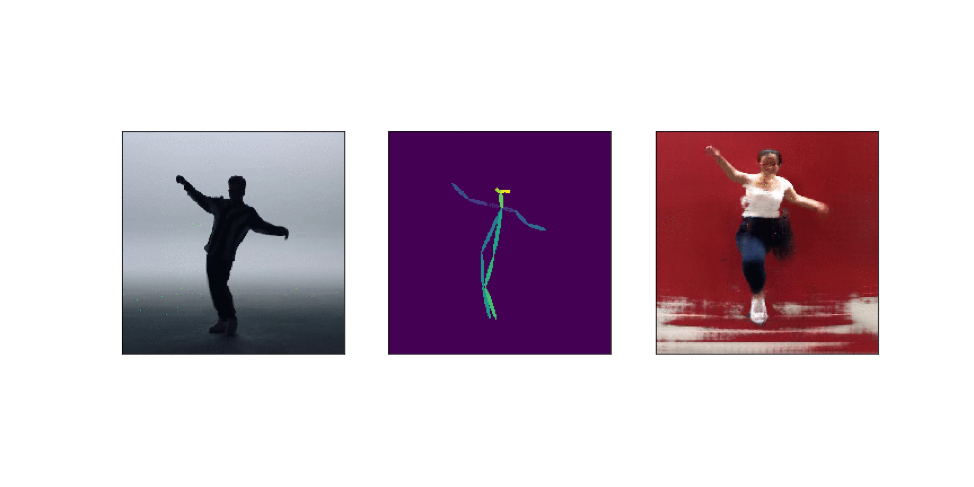
*A picture containing red

Description automatically generatedFigure 4: Synthesized Images created by Generator after Training for Different Number of Epochs*

The end result includes a comparison of two videos generated using a generator trained for 50 epochs and a second generator that was trained for 200 epochs.



*Figure 5: A GIF created after a generator is trained for 50 epochs* 2



*Figure 6: A GIF created after a generator is trained for 200 epochs [[2]](#footnote-2)*

**NEXT STEPS**

1. **Refine Model with Training.** The quality of our results is dependent on the number of epochs used to train both the VGG19 and GANs model. With a longer training time, we would be able to produce more clear images as well as a more realistic human in the image.

Secondly, the training dataset generated from the Target video only included 125 frames. With a larger training dataset that captures a larger variety of poses and movements of the Target, the generator G is expected to have a better performance.

Thirdly, no facial or hand movement enhancement is performed at this stage. Facial and hand movements require more details to be captured and a potential consideration is to perform separate pose estimation for hand and face to improve our results.

1. **Incorporate Temporal Smoothing.** Currently, certain frames in our video are especially choppy when there are large gaps in movement from frame to frame. This is because our model assumes that each frame is independent whereas in reality they are connected by time. Also, the background near the human moves along with the human emotions, because the human is not perfectly captured by our model due to existing noise. Temporal smoothing will produce a smoother frame to frame transitions as well as produce a more stable background. Instead of feeding our discriminator individual pairs (frame and associated pose stick figure), we would instead generate the individual pair (xt, G(xt)) plus a second output of the frame and associated pose stick figure at time t-1 (xt-1, G(xt-1)). Now, the discriminator is tasked with determining the difference in realism and temporal coherence between the ‘fake’ and ‘real’ sequences of (xt, G(xt), xt-1, G(xt-1)).

**REVIEW AND REFLECTIONS**

***Lessons* *Learned***

The most important lesson learned is that good things take time and patience. In addition to the VGG19 and GANs model we trained successfully, there were about five or more rounds of training that failed while training overnight, losing dozens of hours of training time. We lost many hours to fixing side problems that emerged in the process of completing our project. Whether due to a memory error, bug, code logic flaw, each failure was taken in stride.

Throughout the process, we realized that due to the computational constraints, Artificial Intelligence or Machine Learning techniques can be very expensive when applied in business contexts. A few major companies are exploring the potential of using machine learning to solve business problems, which requires large amounts of investments in computational power, data storage and talents.

We also learned that collaboration is more complicated than dividing up work but more energizing when done right, especially in a dynamic development environment. Though certain parts of the project could be handed off to the next person, many parts were intertwined and had multiple complex outputs leading to the next input. Furthermore, even though each existing Github repository we came across reproduced Everybody Dance Now, they each had unique nuances that made combining them difficult. We attempted splitting up work by each tackling a separate Github reproduction, however, found it challenging to streamline all of our work due to the unique flow of each Github and synchronizing our efforts across different Google Cloud/AWS instances.

Another consideration is we could have better employed agile methods when completing the tasks. This would allow the teammates to have more communication and collaborate more effectively.

Collaboration done right meant having productive brainstorming sessions where blocks of time were dedicated to tackling a step of our project together. We each contributed creative solutions and reduced time spent on debugging.

***Surprises***

Our goal for the project was to reproduce the skeleton of our source video dancing; however, we exceeded that by being able to transfer the skeleton motion from the Source to the Target. We attribute the success to getting over the huge learning curve as well as obtaining guidance from the existing work done by other researchers. Before diving into the project, we had assumed the skeleton to Target segment of the project would be more difficult to complete than the source to skeleton segment, however, the opposite was the case.

***Hard/Easy Project Parts***

The two hardest parts of the project were the environment setup and parsing through code reproductions to understand how the code flows together. We faced many initial hardware issues, including operating system incompatibility. After trying our own laptops as well as setting up cloud environments, we finally found Google and Amazon’s Deep Learning pre-configured environments with PyTorch, Caffe, and Cuda driver preinstalled with compatible versions. This also allowed us to collaborate on the same files, much like on Github. Furthermore, in understanding the code, we worked through a Jupyter notebook demo that was provided by CUHKSZ-TQL’s Github reproduction[[3]](#footnote-3). The challenge was transferring the code from 2.7 to 3.6 as well as dealing with missing files and black box packages that were used.

The easiest part of the project was preparing the data for training and transferring the skeleton dancing to the target dancing. Filming the video and modifying the video into the frames and correct dimensions were straightforward. We also tried tuning a few hyperparameters to see if we could improve our own results. We believe transferring the skeleton to the target dancing was relatively simpler than the rest of the project because at that point we had gained a deeper understanding of the subject matter and could navigate/change any of the code to suit our needs.

***Potential Changes***

In generating ideas for a potential project, we were most excited for the Everybody Dance Now project due to the real world implications; however, none of us had the technical foundation in deep learning to create the project ourselves. Much of our time was spent developing an understanding of the theory governing models before trying out the models for ourselves. What we could have done differently at the idea generation phase of the project would be to gather our technical abilities as a team and ascertain projects that are feasible under a one-month time constraint and our abilities.

Another aspect we could have approached differently is using more GPUs. Due to the amount of time lost to model training and failed model attempts, we believe that more time could have been spent on the quality of our output given we used more GPUs.

**RELATED WORK**

***Single Person Pose Estimation***

Traditionally human pose estimation has been carried out by using a combination of local observations and spatial distances between body parts. These models utilized either tree-structured graphical algorithms which helped encode the spatial similarities observed, or non-tree models that used augmented edges to further encode the symmetry, long-range relations, and occlusion into the estimated models.

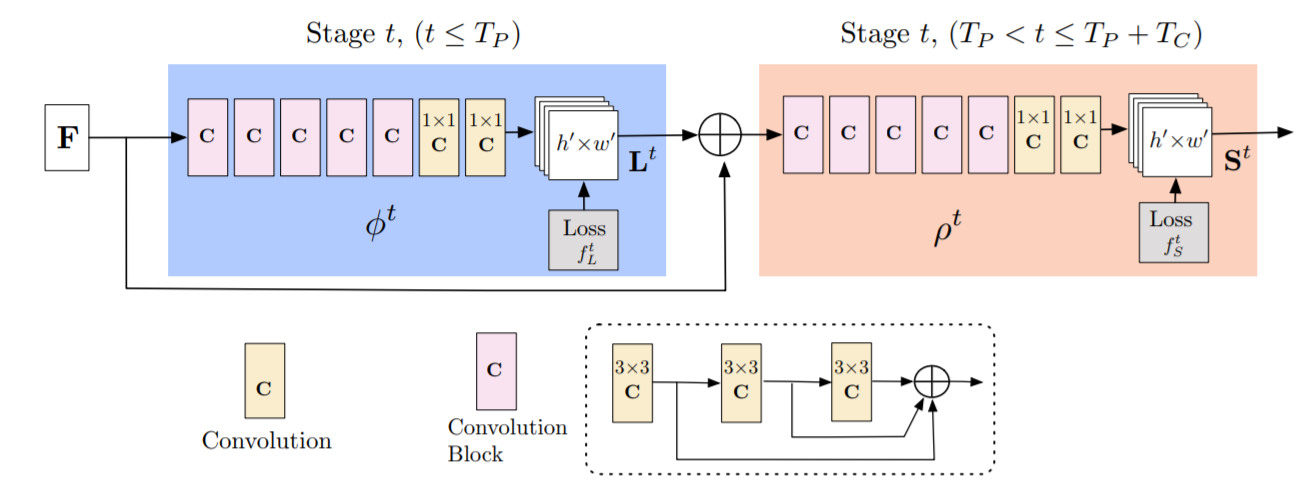
To obtain reliable prediction CNNs have been widely used for body pose estimations: one of the most successful implementations was by Tompson et al. who used them for to jointly learn the parameters with the network. This work was further improved upon by Pfiser et al. who designed networks with large receptive fields to include global spatial dependencies in their models.

Wei et al. proposed a convolutional pose machine architecture which used sequential prediction framework, These helped preserve multi-modal uncertainty from previous iterations by incorporating global context to the the models developed till that time.

However, all of these pose estimation models were limited to a single person where the location and the scale of the person of interest was explicitly mentioned in training.

***Multi-Person Pose Estimation***

Till the release of OpenPose paper, most of the approaches followed a top-down approach where first the object of interest was detected( in this case multiple people), and then the model iteratively figured out the pose of the detected object. This approach has a major flow, where while local the spatial dependencies are being captured, global dependencies between different objects are taken into consideration.



Some approaches that considered this hypothesis in their model were those created by Eichner et al. and Pishchulin et al. Eicher proposed that these interdependencies can be included by taking the depth of the pictorial structures into account using a person detector first. Pichchulin took this approach a step further by reversing the model to a bottom-up one that jointly detects part-detection candidates and their association with individual objects (persons) identified in the model using pairwise scores regressed from spatial offsets of detected parts. The benefit is that the dependency on person detection framework is removed, instead we solve using integer linear programming over fully connected graph- this however becomes a NP hard problem, making optimization troublesome.

This problem was solved by Insafutdinov et al. who utilized ResNet for stronger part detectors generating image dependent scores and thereby reducing the runtime by a huge extent when compared to Pichchulin’s approach. For normal computation however this still took several minutes per second with a limit of 150 part proposals. The pairwise representations used here are essentially offset vectors between every body parts and are hence difficult to regress precisely. They thus ended up using logistic regression to convert the pairwise scores to probability scores instead.

For our case, in the Real-Time Multi-Person Pose Estimation model, several extensions of this previous work on pose detection were made. It is proven that Part Affinity Fields is critical and sufficient for high accuracy. Furthermore removing the body part confidence map can further help define better poses by increasing the network depth. This helps reduce the computation time involved by a huge extent without compromising on the accuracy. Combining both resources not only help reduce the inference time but also helps better the facial keypoint detection used in the models.

We decided to go with the Multi-Person Pose estimation model over a single person estimation because the research regarding Part Affinity Fields is the most modern and accurate model to date. Also, we believe that in the future, human to human motion transfer will eventually extend to multi-person videos.

For pix2pixHD translation, we are already using the most modern and accurate models, minus a few of the additional body part enhancements (i.e. temporal smoothing and wrist/hand image enhancements).

**WORK DISTRIBUTION**

Karan: presentation outline, final paper Related Work section, set up Google Cloud.

Cindy: pix2pixHD image translation and transfer to generate results, final paper Data/Results sections, general edits of the paper, presentation slides related to pix2pixHD training and transfer processes.

Michell: Pose Estimation, final paper Introduction/Architecture Overview sections, general edits of the paper, presentation slides related to pose estimation training and overall edits of the presentation slides.

**APPENDIX**

Code and more visualizations are available on: [Michell's Github - Everybody Dance Now](https://github.com/ml42322/Everybody-Dance-Now)

1. <https://github.com/NVIDIA/pix2pixHD> [↑](#footnote-ref-1)
2. Available at Project GitHub Repo: <https://github.com/ml42322/Everybody-Dance-Now> [↑](#footnote-ref-2)
3. <https://github.com/CUHKSZ-TQL/EverybodyDanceNow_reproduce_pytorch> [↑](#footnote-ref-3)