Video Outfit Transfer

Andrew Jong, Teng-Sheng Moh
Department of Computer Science
San José State University
San José, CA
andrew.m.jong@gmail.com, teng.moh@sjsu.edu

***Abstract*—In this project, we partially reimplement outfit transfer for video, based on the work of SwapNet [1]. By successfully training the warp stage on video, we show that the previously proposed transfer technique for images has potential to apply to subjects in motion. In addition, specifically for video applications, we show the model need not be trained on a comprehensive dataset; instead the model only needs videos for the source outfit and target subject.**

This project represents the first steps of a much larger project for robust, context-aware, temporally consistent video outfit transfer; a large section is dedicated to future work.

I. Introduction

Image-to-image translation is a method that transfers images from one domain to another, similar to translation between languages. This started as art style transfer from paintings to photos [2]. Several works in the past two years have demonstrated the immense power of image-to-image translation. For example, Chan et al. [3] showed how to perform motion transfer in dance, and showed that transfer may be locally applied to an individual subject in video.

Since it is shown that advanced transfer techniques are possible, this project attempts the following goal: to transfer clothing from one subject to another. This means the proposed model will transfer the clothing worn by one person to another, with realistic placement on the target. This application is expected to have high relevance in the online shopping industry.

The leading work in this task is *SwapNet* by Raj et al. [1]. SwapNet was published in 2018 at the ECCV conference, and is the first to successfully transfer outfits over images between two subjects of arbitrary pose. More details of SwapNet may be found in Section III.

II. Related Work

A. Generative Adversarial Networks

The basis of SwapNet is the Generative Adversarial Network (GAN). Goodfellow et al. [4] introduced the GAN model for generative processes, and showed its potential for image generation. A GAN simultaneously uses a generator and discriminator network in its training process. Specifically, while the generator tries to generate realistic images, the discriminator attempts to discriminate between fakes generated by the generator and real images from the training set. A combined loss function enables the generator and discriminator to train adversarially, resulting in incredibly realistic results. The conditional GAN is the basis of many modern generative techniques.

A year later, Mirza and Osindero [5] expanded on GANs by proposing a condition on the generator, i.e. conditional GANs (cGANs). cGANs allow generated samples to be conditioned and customized on an input. Isola et al. [6] further proved that cGANs could be applied to image-to-image translation, and developed the code framework `pix2pix` for convenient implementation.

B. Outfit Generation and Transfer

SwapNet is not the only work that has dealt with clothing generation and parsing. Other work is described below.

ClothNet was proposed by Lassner, Pons-Moll, and Gehler [7]. It is a generative model that generates random clothing samples with conditional inputs of pose and color. ClothNet achieves realistic results; however, ClothNet focuses purely on clothing generation rather than clothing transfer.

DeepGarment (Danerek et al. [8]) is a parametric clothing extractor and transfer technique, that transfers clothing between individuals. DeepGarment uses an intermediary 3D parametric model to accomplish this task. However, the parametric model imposes limits on realism of the garment, and looks obviously "rendered". Solving outfit transfer without a parametric representation widens the possible boundaries for generation.

VITON, short for Virtual Try-On Network, from Han et al. [9] is one work that takes garments and places it on individuals. However, VITON starts from well-segmented, isolated clothing pieces and does not attempt subject-to-subject transfer. Subject-to-subject transfer is a harder task, because the network must learn the target clothing representation from a non-standard form from subjects in any pose.

None of the above work, nor SwapNet, has attempted realistic video-to-video outfit translation using non-parametric methods.

III. SwapNet Overview

The details of SwapNet may be found in the original paper. However, this section provides a brief summary for the reader's convenience.

SwapNet is trained on the DeepFashion dataset [10], which contains images of many individuals wearing different clothing. It uses two preprocessing networks to extract body and clothing segmentations from each individual. Individuals

wearing different clothing will have different clothing segmentations, while all individuals will share the same parts from the body segmentation. The common ground of body segmentation is used to transfer the different outfit variations. SwapNet divides into two conditional GAN modules for training: (1) the warping module, and (2) the texturing module. SwapNet uses these two networks in conjunction to achieve outfit transfer. See Figure 1 for details with a diagram.

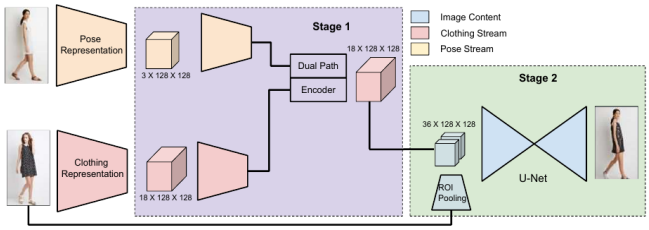


Fig. 1: From [1]: "Our pipeline consists of two stages: (1) the warping stage, which generates a clothing segmentation consistent with the desired pose and (2) the texturing stage, which uses clothing information from desired clothing image to synthesize detailed clothing texture consistent with the clothing segmentation from the previous stage."

Fig. 2: Sample frames from the video dataset. The top row features subject A, the bottom features subject B. Notably, subject A wears long sleeves while subject B wears short sleeves.

IV. Implementation

A. Data Collection

We recorded two videos for the dataset, one of subject A and one of subject B. Both subjects wear visually distinct outfits, but stand against the same background. The background is complex, featuring a textured wall, windows, bench, and bush. In each video, the subject walks around and strikes various poses to add variety to the dataset; this includes turning in circles and sitting. The videos were recorded vertically with a Samsung Galaxy S9 phone camera at 1080x1920p resolution and 30 frames per second. Both videos were recorded for approximately 5 minutes. We used the *ffmpeg* Linux command-line tool to convert *.mp4* video format to *.png* images, yielding 9615 images for subject A and 7637 images for subject B. Images were downsized to 540x960 resolution to fit on the available GPU memory. Each image file was named with its sequential frame number. Images were then cropped to 512x512 to a region such that each subject's face was still in frame.

B. Preprocessing Networks

SwapNet originally used *LIP_SSL* [11] and *Unite-the-People* [12] networks for clothing and body segmentation. We replace those networks with newer, more accurate networks. For clothing segmentation, we used *LIP_JPPNet* [13]. LIP_JPPNet is an updated version of LIP_SSL with higher segmentation accuracy. SwapNet originally described representing the 18 clothing segmentation labels as 18-channel probability maps. We instead opt to directly use the 3-channel RGB output instead of the probability maps used in SwapNet. Despite that this representation is significantly less discretized than the one-label-per-channel method, 3-channel RGB is still able to provide enough information to generate the desired warp transform.

For the body segmentation network, we choose to use *WSHP*, proposed by [14]. *WSHP*, in contrast to *Unite-the-People*, runs in real time. This lowers computation time to achieve the final result. One potential drawback of *WSHP* is that it does not always segment faces accurately; in provided examples and in our own results, the face segmentation often drops part of the chin. However, the SwapNet paper claimed convincing results could still be obtained even with noisy segmentations from the preprocessing networks, and so the inaccuracy of *WSHP* did not cause much concern. Like the original SwapNet, the body segmentation is left as 3-channel RGB images.

To run inference with both preprocessing networks, we used the Google Colab cloud GPU platform. Segmentation samples were extracted from each frame from the videos. By running four Colab instances in parallel, inference completed within an hour for each network.

C. Warp Stage

1) Training Samples: The warp stage takes as input (1) a source clothing segmentation for the desired clothes S_{cs} and (2) a body segmentation for the target pose T_{bs}. The output (3) is the desired clothes warped to the desired pose T_{cs}. During training, we train using a single video, thus the source S and target T are the same.

Fig. 3: Body and clothing segmentation network outputs for both subjects. Left to right: body segmentation subject A, clothing segmentation subject A, body segmentation subject B, clothing segmentation subject B,

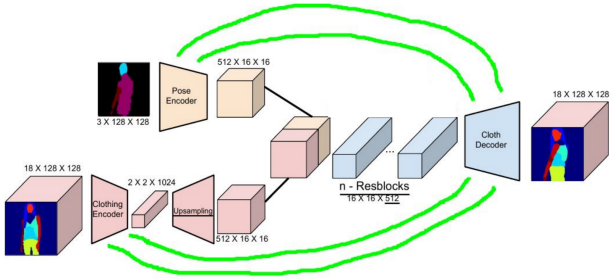


Fig. 4: The SwapNet warp stage diagram with additional lines drawn in green to indicate dual-path skip connections.

Each training batch iterates over these input-output triplets. By doing so, the warp network learns to arbitrarily pose any clothing segmentation based on a given body segmentation. During inference, the source of the body segmentation may be swapped for the target subject that we wish to place clothes on. This allows the warp stage network to properly "fit" the desired clothing on the target.

To obtain samples of these data triplets, we use the pre-processed outputs extracted via the networks described above. Each video frame has a matching body and clothing segmentation; this provides part 2, a body segmentation for the desired pose, and part 3, the output of the desired clothes in the desired pose.

To obtain the clothing segmentation to be warped, we select a random frame with at least distance t from the current frame number. We set this value to 100 frames. This ensures the input clothing segmentation is not too similar to the target clothing segmentation, so that the warp stage does not simply learn identity-mappings during training.

The original SwapNet work did not use a video dataset from which they could sample many input clothing segmentations from the same subject; thus, to augment training data, they performed random transforms on each channel within their 18-channel probability maps. Because the video source in our implementation does provide many samples of the same subject, we need not perform such granular augmentation. Instead, we performed simple affine augmentations directly on the clothing segmentation image.

2) Generator Architecture: Like the original work, we implement the warp stage as a dual-path U-Net [15]. A diagram is provided in Figure 4 to illustrate the skip concatenations from dual inputs in the U-Net.

Remaining architecture details are based on the warp stage diagram, aside from minor differences. The original work used low resolution 128x128 images; as our video was recorded at higher resolution, we use images four times larger at 512x512. Dropout with $p = 0.5$ is inserted in the intermediate encoded layers.

For generator loss, because our work directly uses segmentation images instead of 18-channel probability maps, we modify the original 18-class cross entropy function into pixel-wise cross entropy loss; the function reshapes the output tensor into a vector and sums the cross entropy loss per element. The total generator loss, similar to SwapNet, was computed as a weighted sum that combined the discriminator loss and pixel-wise cross entropy loss ($\lambda_{adv} = 0.2$).

3) Discriminator Architecture: Training of the GAN also requires a discriminator. The SwapNet authors gave no guidelines for the discriminator, so our design follows the DRAGAN [16] discriminator implementation. We use a simple network consisting of four sequential *convolutional-LReLU-dropout* blocks, followed by a dense and sigmoid layer. The discriminator takes as input both the body and clothing segmentation concatenated on the channel axis as a single tensor. In this way, the discriminator may take advantage of information from both the input and condition for its binary classification task. The discriminator uses binary cross entropy loss.

D. Texturing Stage

1) Training Samples: The texture stage requires ROI pooling based on the bounds of the preprocessed body segmentation. ROI bounds are stored as *.csv* files containing: body part ID, xmin, xmax, ymin, ymax. The ROI bounds are then clipped to match the 512x512 dimensions of each sample image.

Sampling the texture dataset behaves similarly to that of the warp dataset, in that it first chooses a target texture image at index i, finds the matching ROI bounds by filename, then chooses a random input texture image at a minimum threshold distance t.

Unfortunately, the ROI pooling layer is not built natively into PyTorch. Current attempts to integrate third party implementations are in progress. [17]

V. Experiments and Results

This section summarizes the experiments, train process, and results from the implementation. We collected two videos, and so performed two experiments: (1) to transfer clothes from subject A to subject B, and (2) to transfer clothes from subject B to subject A. Because only the warp stage is working at this time, these results pertain only to the warp stage.

A. Hyperparameters

The Adam optimizer was used to train the warp stage network. Learning rate was set to 0.0002, beta-1 to 0.5, and

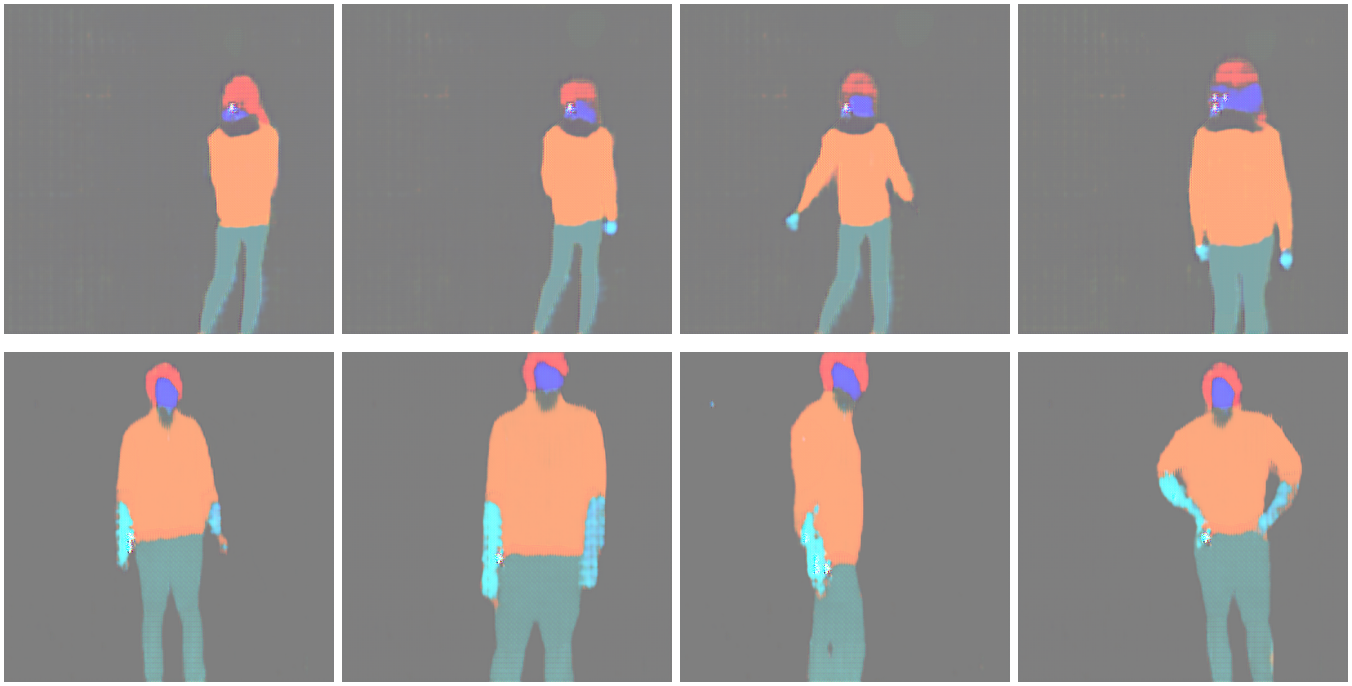


Fig. 5: Inference results on unseen body segmentations for both trained models. Note subject A originally wore long sleeves, while subject B originally wore short sleeves. Top row: transfer subject A's clothes (long sleeves) to subject B. Bottom row: transfer subject B's clothes (short sleeves) to subject A. In both instances, the sleeve length is successfully transferred.

beta-2 to 0.99. We used the max batch size (8) that would fit on the available GPU.

Initial train trails of 200 epochs turned out to be too high, as the discriminator quickly outperformed the generator. On the first training run, the generator created convincing output at around epoch 15-20. But as training continued, the discriminator loss fell to 0, and as a result the generator began generating ridiculous output in an attempt to compensate. We solved this issue by first limiting epochs to 20, and by setting an automatic stopping threshold to stop training if discriminator loss fell below 0.05 or generator loss increased above 3.0. The model was set to save every 1000 steps, and to draw a sample to observe progress every 500 steps.

B. Train Progress

Training steadily progressed over 15-20 epochs. At step 0, the generator simply concatenated the body and input clothing segmentations. Around step 3000, the network began "ghosting" out the input clothing segmentation's original shape in preference for a clothing segmentation closer to the target. In between steps 4000 and 19,000, the generator produced and refined artifacts and noise from the image.

Please see Figure 6 for detailed images of train progress.

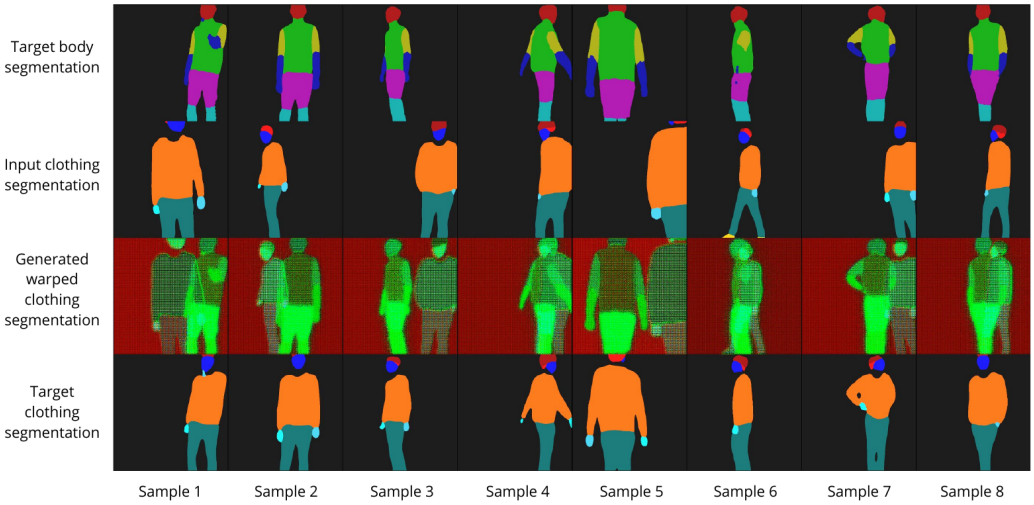
C. Inference Results

We ran inference sequentially through the frames of both videos. For transferring clothes from subject A to subject B, the task is to warp subject A's clothes to subject B's pose. Thus for inference from A to B, I set the input to be subject A's clothing segmentations and subject B's body segmentations. The same technique applies vice-versa for transferring clothes from subject B to subject A. In both inference runs, the model is given body segmentations outside of its training knowledge.

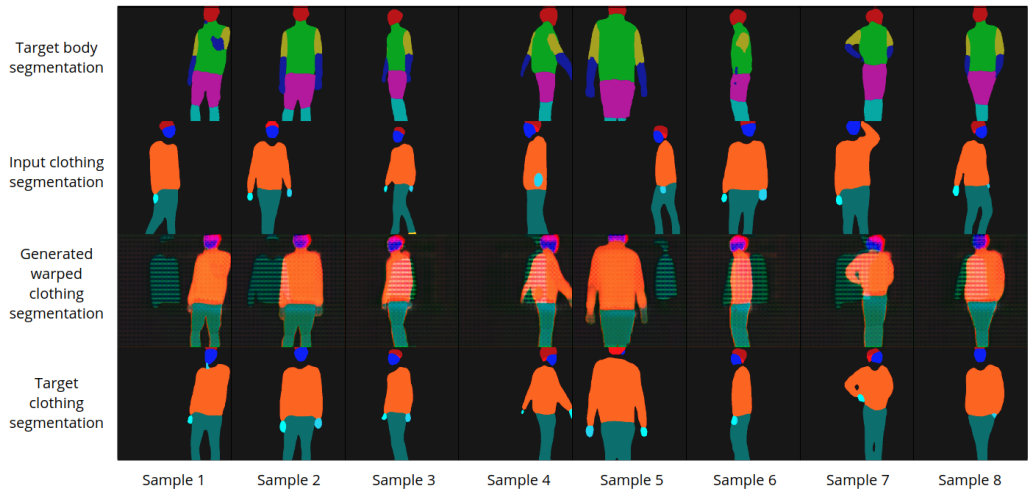
The inference results show the clothing segmentations successfully transferred between subjects. In the output that projects subject A's *long* sleeves onto subject B's pose, subject B now displays long sleeves. Similarly, in the output that projects subject B's *short* sleeves onto subject A's pose, subject B now displays short sleeves (see Figure 5).

However, there are undoubtedly limitations in the current work. The long sleeves appear to have transferred better than short sleeves, as the short sleeves sometimes flicker in between short and long colors (cyan and orange, respectively). The short sleeves are also less cleanly shaped than the long sleeves and often produce artifacts. This may be because the long sleeves match the color with the rest of the torso, while the short sleeves are a very different color; the network may have a harder time with such a drastic difference. Another possible explanation is that the short sleeves dataset had nearly 2000 less frames than the long sleeves dataset.

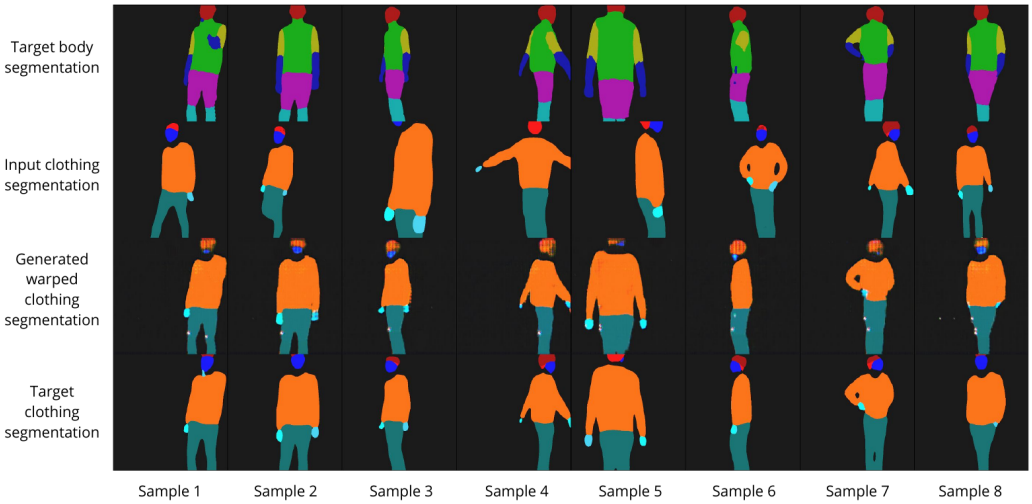
One task the generator completely fails to attain is the correct positioning of the face. This is expected; the body segmentation embeds no directional information, so the generator has no way to know whether a person (and their face) is facing backwards of forwards. The generator attempts to compensate by either always facing the face in one direction, or melding between face directions. Luckily, this does not pose a problem; in the texture stage, the original SwapNet authors simply copied over the face and hair pixels from the target



(a) At step zero, because no learning has occured yet, the generated output is simply the two inputs overlayed.



(b) At step $1,000$, the generator begins ghosting out the body segmentation shape.



(c) At step $18,500$, the generator has learned to warp the input clothing segmentation to the body segmentation, and is posed similarly to the target. Ghosting effects have disappeared.

Fig. 6: Train progress steps for subject A.

subject, independent of the generated image's face direction. When the texture stage is complete, we will also use this approach.

VI. Planned Future Directions

First, we wish to solve the ROI pooling problem. We recently received back an email from the SwapNet authors who suggested the third-party library they used, so we will begin there. Barring success with their recommended library, we could attempt to implement our own ROI pooling layer with CUDA code.

Second, we wish to collect a larger and more varied dataset. Our current dataset only has two subjects on the same background. In addition, while clothes of our subjects are different in texture, there could differ more in shape. Thus, we wish to collect longer videos of more subjects wearing significantly different clothes, such as summer and winter clothes. We could also collect different clothing types, such as flowing and rigid structures. Camera angles should also be changed and varied. By collecting this more substantial dataset, we will be able to create and validate more robust methods for future video-transfer work.

Third, we wish to use the 18-channel clothing segmentation maps as proposed in the original SwapNet paper, and compare this to our RGB channel results. We plan to obtain a large harddrive capable of holding the dataset, and a faster GPU to put training time within feasible limits.

Fourth, we wish to experiment with switching out the two preprocessing networks of SwapNet. It is possible more detailed representations would yield more accurate results. For example, we could use a 3D body segmentation or a clothing segmentation that outputs more clothing labels.

Fifth, we wish to implement temporal smoothing as proposed by Chan et al. [3], to reduce video artifacts and glitches. Temporal smoothing will help improve the realism of the network's output.

Sixth, we must find and implement suitable quantitative metrics for this type of semisupervised learning. Because there are no true labels, classification metrics such as F1-score do not apply. Re believe inception score also does not apply; part of the inception score depends on matching internal class distributions, but in this case, the generated distribution is *intended* to be significantly different from the original distribution because of the *transfer* nature of the task. Transferring would have little purpose if the objects of interest transferred to the same subject. We intend to look at work from Isola et al. [6] and Chan et al. [3] for ideas of quantitative scoring in the domain of image translation.

Seventh, currently inference is conducted by pulling matching frames from the reference video, but this is not ideal in cases where the relevant clothing part is occluded (such as a subject placing their arms behind their back). A better solution may be to either (1) choose a single frame with all clothing parts visible, (2) choose a nearest-neighbor frame, or (3) somehow encode information from the entire reference video. Ideally, we would implement a method that is robust to occlusions.

Eigth, a problem is the current implementation requires a full video for training on the clothing segmentation. Ideally, we would like to find a method to enable real-time inference via transfer learning of clothing information. Perhaps the network can be trained on a diverse dataset such as DeepFashion, then applied in real-time on the target with all the desired improvements above. This would make the network truly viable for consumers in online markets.

VII. Conclusions

Outfit transfer is an application that has mass potential in online shopping and virtual reality experience. This project is designed to eventually make that potential a reality. In this project, we successfully reimplemented the warp stage of SwapNet from scratch. We showed a video dataset may be used to train the generation network to produce video-results. This work represents the beginning of a much longer project of turning outfit transfer into a realistic reality; the door remains open to the described future work.

References

[1] Amit Raj et al. "SwapNet: Garment Transfer in Single View Images". In: *The European Conference on Computer Vision (ECCV)*. Sept. 2018.

[2] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. "Image Style Transfer Using Convolutional Neural Networks". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2016, pp. 2414–2423. DOI: 10.1109/CVPR.2016.265. URL: https://doi.org/10.1109/CVPR.2016.265.

[3] Caroline Chan et al. "Everybody Dance Now". In: *ArXiv e-prints* (Aug. 2018). arXiv: 1808.07371 [cs.GR].

[4] Ian Goodfellow et al. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani et al. Curran Associates, Inc., 2014, pp. 2672–2680. URL: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

[5] Mehdi Mirza and Simon Osindero. "Conditional Generative Adversarial Nets". In: *CoRR* abs/1411.1784 (2014). arXiv: 1411.1784. URL: http://arxiv.org/abs/1411.1784.

[6] Phillip Isola et al. "Image-to-Image Translation with Conditional Adversarial Networks". In: *CoRR* abs/1611.07004 (2016). arXiv: 1611.07004. URL: http://arxiv.org/abs/1611.07004.

[7] Christoph Lassner, Gerard Pons-Moll, and Peter V. Gehler. "A Generative Model of People in Clothing". In: *CoRR* abs/1705.04098 (2017). arXiv: 1705.04098. URL: http://arxiv.org/abs/1705.04098.

[8] R. Danerek et al. “DeepGarment : 3D Garment Shape Estimation from a Single Image”. In: *Comput. Graph. Forum* 36.2 (2017), pp. 269–280. DOI: 10.1111/cgf.13125. URL: https://doi.org/10.1111/cgf.13125.

[9] Xintong Han et al. “VITON: An Image-Based Virtual Try-On Network”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018.

[10] Ziwei Liu et al. “DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations”. In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[11] Ke Gong et al. “Look Into Person: Self-Supervised Structure-Sensitive Learning and a New Benchmark for Human Parsing”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[12] Christoph Lassner et al. “Unite the People: Closing the Loop Between 3D and 2D Human Representations”. In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. July 2017. URL: http://up.is.tuebingen.mpg.de.

[13] Xiaodan Liang et al. “Look into Person: Joint Body Parsing & Pose Estimation Network and a New Benchmark”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2018).

[14] Hao-Shu Fang et al. “Weakly and Semi Supervised Human Body Part Parsing via Pose-Guided Knowledge Transfer”. In: *CVPR* (2018).

[15] Bo Zhao et al. “Multi-View Image Generation from a Single-View”. In: *Proceedings of the 26th ACM International Conference on Multimedia*. MM ’18. Seoul, Republic of Korea: ACM, 2018, pp. 383–391. ISBN: 978-1-4503-5665-7. DOI: 10.1145/3240508.3240536. URL: http://doi.acm.org/10.1145/3240508.3240536.

[16] Naveen Kodali et al. “How to Train Your DRAGAN”. In: *CoRR* abs/1705.07215 (2017). arXiv: 1705.07215. URL: http://arxiv.org/abs/1705.07215.

[17] Jianwei Yang. *PyTorch-GAN*. https://github.com/jwyang/faster-rcnn.pytorch. 2018.