Virtual Try-On

Subhajit Ghosh

Advisor: **Dr. Kaustuv Nag**

Department of Computer Science and Engineering
Indian Institute of Information Technology Guwahati

This dissertation is submitted for the degree of
Master of Technology

IIIT Guwahati November 2022

to my loving parents…

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Subhajit Ghosh
November 2022

Acknowledgements

This project report on "Virtual Try-On" brings me a sense of accomplishment and great delight. I acknowledge Dr. Kaustuv Nag for the invaluable guidance and support provided throughout this project. I also thank my friends for the cherished time spent during this degree. My appreciation also goes out to my parents for their unwavering encouragement and support throughout all my studies.

Abstract

With the key increase in the clothing industry on commerce platforms like Amazon, Flipkart, or in offline stores, each competitor wants to utilise immersive technology to offer consumers personalization, improve sales results, and reduce the number of returns. Virtual try-on allows customers to "try on" items such as clothes, makeup, and many more. A customer sitting in his or her home can try on how they will look in a given item or product without physically buying it. Here we use a self-supervised conditional generative adversarial network-based framework comprised of a fabricator and a unified virtual try-on pipeline with a segmenter, warper, and fuser. We use the viton dataset, which consists of 16,253 pairs of images of a person and a clothing item, to train and illustrate the effectiveness of the proposed method.

Table of Contents

List of Figures

Chapter 1

Introduction

Nowadays, people purchase many products online and spend more on them, especially fashion items like clothes. Today, different styles and categories of clothing are easy to find with a few mouse clicks. Despite the convenience of online shopping, customers are often concerned about how a particular fashion item image on a website or an app will fit them. With the recent progress in virtual try-on technologies, people can have a better online shopping experience by accurately imagining themselves wearing clothing from online stores. Furthermore, not only online shopping but also physical shopping demands virtual try-on technologies. Customers can save time by avoiding the fitting room with try-on technologies designed for mobile applications.

Previous virtual try-on methods focused on aligning a clothing item with a person and limiting its ability to fully exploit the complex pose, shape, and skin color of the person, as well as the underlying structural details of the clothing, such as logo, texture, and embroidery, which is very important to a photo-realistic virtual try-on.

In this project, we examine Fill-in-Fabrics (FIFA) [5]. Fill in Fabrics is a self-supervised conditional generative adversarial network model. It is a body-aware inpainting framework for image-based virtual try-on. The proposed FIFA framework can synthesize a more realistic logo, texture, and embroidery of the target clothing and tackles well-person images with complex poses, including hands occlusion and waist position. The FIFA model achieves state-of-the-art results on the standard VITON dataset for virtual try-on of clothing items and is effective at handling complex poses and retaining the texture and embroidery of the clothing.

1.1 What is Virtual Try-On ?

In layperson's terms, a person virtually tries on products such as clothing, jewelry, and makeup using mobile phones or other devices, known as virtual try-on. Virtual try-on represents a practical application of Generative Adversarial Networks (GANs) and pixel translation.

1.2 Why Virtual Try-On ?

It has a gigantic effect on the fashion industry. It helps consumers in making superior choices. Consumers get to explore the item at their possess pace until they discover the proper choice. Clothing things account for the majority of returns in online shopping. Customers return clothing due to neglected desires, and each return of the thing encompasses a noteworthy natural effect amid fabricating, bundling, and transportation. In financial terms, companies frequently waste returns as a cheaper choice than restocking things.

Chapter 2

Literature Review

2.1 Generative Adversarial Networks (GANs)

2.1.1 What are Generative Model?

Unsupervised learning approaches are used in generative models. Generative models are used to generate new data from samples that are not similar to the example.

2.1.2 What are Generative Adversarial Networks (GANs)?

The concept of Generative adversarial networks [1] are inspired by game theory, where the generator and the discriminator compete with each other. We illustrate a block diagram of generative model in fig. 2.1.

The principle of generator G is to generate fake data as much as possible so that it looks like an actual data sample, while discriminator D is to correctly distinguish

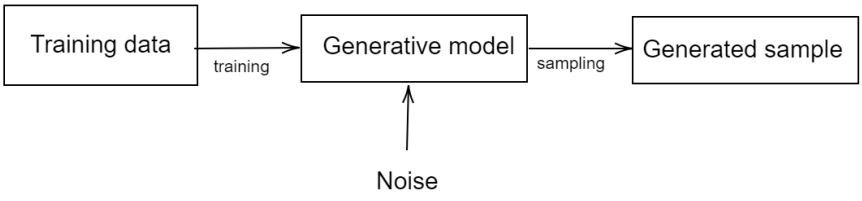


Fig. 2.1 Generative model

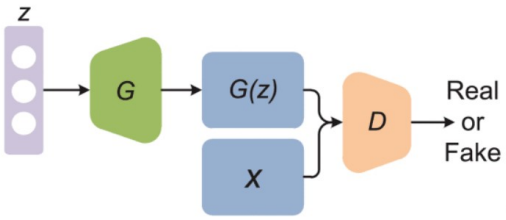


Fig. 2.2 The architecture of generative adversarial networks

real data from fake data. The generator's input is a random noise vector z (usually drawn from a uniform or a normal distribution). The noise is mapped to a new data space via the generator G to obtain a fake sample, G(z), a multi-dimensional vector. Again, the discriminator D is a binary classifier. It takes both real samples from the dataset and fake samples generated by the generator G as the input, and the output of discriminator D represents the probability that the sample is real rather than fake. The optimal state is reached when the discriminator D fails to determine whether the data comes from the real dataset or the generator. At this point, we obtain a generator model G, which has learned the distribution of real data. As two players in game theory, both the generator and the discriminator have their loss functions. We call these $J(G)$ and $J(D)$, respectively. The discriminator D is defined as a binary classifier, and the loss function is represented by the cross entropy as follows:

$$J^{(D)} = \left\{ -\frac{1}{2}\mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}[\log D(\mathbf{x})] - \frac{1}{2}\mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}}[\log(1-(D(G(\mathbf{z}))))]\right\} \tag{2.1}$$

where x represents the real sample, z represents a random noise vector, $G(z)$ is the data generated by the generator for the input noise z, and E represents expectation. The goal of D is to correctly determine the source of the data, so it wants $D(G(z))$ to approach 0 and $D(x)$ to approach 1, while the goal of G is to bring $D(G(z))$ to 1. Therefore, the loss of the generator can be derived by the discriminator:

$$J^{(G)} = -J^{(D)} \tag{2.2}$$

Let's assume we have human height and weight distribution. Assume the pink dots represent the original data and the light blue dots represent the fake data. The data will be placed in a linear space by Discriminator between 0 and 1. Initially, the points generated by the generator are spread (not overlapping) across the data we needed. As we can see in fig. 2.4, blue dots represent the points generated by the generator.

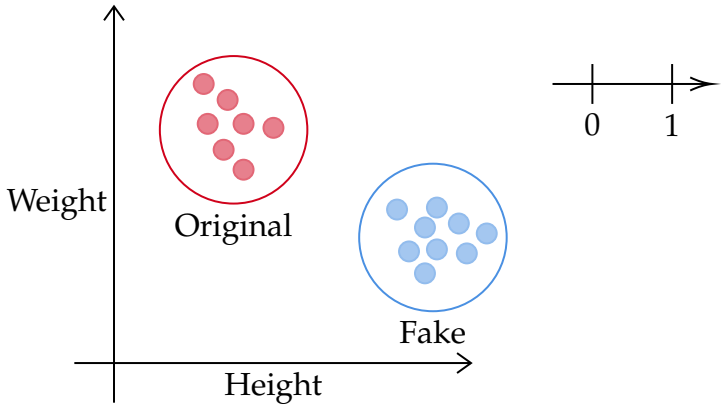


Fig. 2.3 Human height and weight distribution

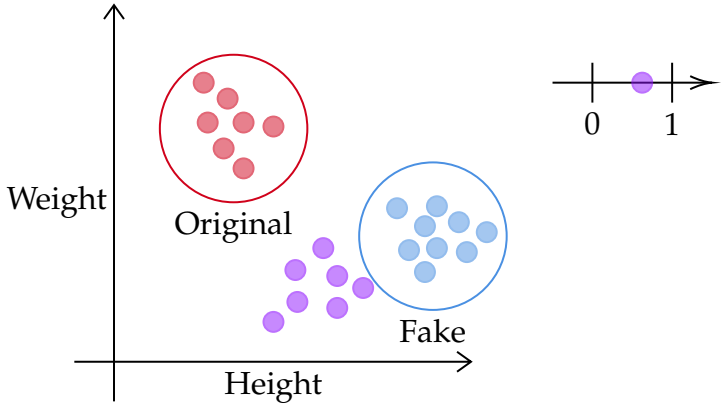


Fig. 2.4 Human height and weight distribution after few iterations

During initial iterations, the discriminator can easily tell the difference between the fake generated data and the original data. The discriminator will keep updating itself, as will the generator. The generator will try to get the data points closer to 1. The discriminator wants to minimise the loss function, and the generator wants to maximise the loss function. Eventually, the optimization problem of GANs is transformed into the following minimax game:

$$\min_G \max_D \left\{ \mathbb{E}_{\mathbf{x}\sim p_{\mathbf{x}}}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}}[\log(1 - D(G(\mathbf{z})))] \right\} \tag{2.3}$$

2.2 Patch Routed Spatially Adaptive GAN

Patch-routed spatially-adaptive GAN (PASTA-GAN) [3] is a virtual try-on system that can transfer garments between an input cloth and a target person in an unsupervised manner. It has a patch-routed disentanglement module for retaining

the texture of the garment. Using the source person's key points, it first makes the garment into normalized patches, thus removing the inherent spatial information of the garment, and then reconstructs the normalized patches onto the wrapped garment. It then has spatially adaptive residual blocks that guide the generator to synthesize more realistic garment details. Patch-routed spatially-adaptive GAN (PASTA-GAN) [3], can generate high-quality try-on images when faced with a large variety of garments.

2.2.1 Patch-routed Disentanglement Module

The synthesis network was trained using image reconstruction. It takes a person's image as input to extract the feature of the person's representation to reconstruct the original person's image and features of the intact garment. The features of the intact garment entangle the garment, and style with the spatial information in the original image. Garment style refers to the color and category of the garment, for example, long sleeve or short sleeve. The spatial information refers to the relative size of the garment patch in the person's image, orientation, and location. It divides the garment into normalized patches to remove the inherent spatial information about the garment.

2.2.2 Attributed-decoupled Conditional StyleGAN2

StyleGAN2 is used for image generation. PASTA-GAN [3] used the main architecture of StyleGAN2. The normalized patches are projected to the style code, through a style encoder followed by a mapping network, which benefits from the disentanglement module. The target head and pose are transferred into a feature map, which encodes the target person's identity by the identity encoder.

2.2.3 Spatially-adaptive Residual Module

It addresses the issue of misalignment between the warped garment and the target garment mask. As shown in fig. 2.5 the orange and green regions represent the region to be inpainted need to be removed. It uses a garment encoder and three spatially-adaptive residual blocks with a feature inpainting mechanism to modulate intermediate features by leveraging the inpainted warped garment feature.

Fig. 2.5 Illustration of misalignmnet between the wraped garment and target garment shape [3]

2.3 Adaptively Generating Preserving Image Content

Towards Photo-Realistic Virtual Try-On by Adaptively Generating Preserving Image Content [4], it settles the issue of the semantic and geometric distinction between the target dress and referenced pictures beside the occlusions between the torso and limbs. It classifies pictures within the taking in three ways:

1. Easy level sample: image is presented with a standard posture i.e face forward and hands down.
2. Medium level sample: images present the twisting of the body torso or one of the hands overlapping with the body.
3. Hard level sample: images show both torso twisting and two hands blocking in front of the body.

It consist of 3 modules semantic generation module (SGM), clothes warping module (CWM) and content fusion module (CFM).

2.3.1 Semantic Generation Module (SGM)

It generates a mask of the body parts and a mask of the warped clothing region via semantic segmentation resulting in semantic alignment of the spatial layout. Here it train a body parsing conditional generative adversarial network (cGAN) (G_1) to generate M_W^S mask of body parts w = {head, arms, bottom} by using the information from fused map M^F, the posed map M_P and target clothing image T_C. It is tractable

to get the estimated clothing region. M_W^S mask of body parts, the posed map M_P and target clothing image T_C are combined to generate synthesized mask of the clothes M_C^S using conditional generative adversarial network (cGAN) G_2. For each stage, the CGAN loss can be formulated as:

$$\mathcal{L}_1 = \mathbb{E}_{x,y}[\log(\mathcal{D}(x,y))] + \mathbb{E}_{x,z}[\log(1-\mathcal{D}(x,\mathcal{G}(x,z)))] \tag{2.4}$$

where x indicates the input and y is the ground-truth mask. z is the noise. Apart from L_1 we have L_2 pixel wise cross entropy loss, it improves the quality of synthesized mask from generator. Therefore overall objective function for each stage of the mask generation module is given as

$$\mathcal{L}_m = \lambda_1 \mathcal{L}_1 + \lambda_2 \mathcal{L}_2. \tag{2.5}$$

Constants λ_1 and λ_2 are the trade-off parameters for each loss term in (2.5), which are set to 1 and 10, respectively in this experiments.

2.3.2 Clothes Warping Module (CWM)

It wraps the target clothing image according to the wrapped clothing mask. It yields geometric matching yet character-retentive clothing images. Here we give T_C target clothing image and M_C^S target clothing mask as input to Spatial transformation Network (STN) to learn a mapping between them. The wrapped clothing image T_C^W is transformed by the parameters learned from STN. Here the loss term is given as:

$$\begin{aligned}\mathcal{L}_3 = \sum_{p\in\mathbf{P}} \lambda_r \left| \|pp_0\|_2 - \|pp_1\|_2 \right| + \left| \|pp_2\|_2 - \|pp_3\|_2 \right| \\ + \lambda_s \left(\left| S\left(p,p_0\right) - S\left(p,p_1\right) \right| + \left| S\left(p,p_2\right) - S\left(p,p_3\right) \right| \right)\end{aligned} \tag{2.6}$$

As given in (2.6) $p(x,y)$ represent a certain sample control point and $p_0(x_0,y_0)$, $p_1(x_1,y_1)$,$p_2(x_2,y_2)$ and $p_3(x_3,y_3)$ are the top, bottom, left and right sampled control points of $p(x,y)$. The wrapping loss can be represented as L_w, which measures the loss between the wrapped clothing image T_C^W and it is ground truth I_C.

$$\mathcal{L}_w = \mathcal{L}_3 + \left\| \mathcal{T}_c^W - \mathcal{I}_c \right\|_1. \tag{2.7}$$

Here, T_C^W is the wrapped clothing image and I_C is the ground truth.

2.3.3 Content Fusion Module (CFM)

It integrates the information from previous modules to adaptively determine the generation or preservation in the output synthesized image.

2.3.4 Quantitatively score the complexity of image

It categorized images in three level easy, medium and hard as follows:

$$\mathcal{C} = \frac{1}{N} \sum_{t \in \mathcal{M}_{p'}}^{N} \left\| t - \frac{\sum_{t \in \mathcal{M}_{p'}}^{N}}{N} \right\|_1 \tag{2.8}$$

Here, M_p represent points of left/right arm, left/right shoulder, left/right hand and torso, $t = (x_t, y_t)$ is a certain pose point, $N = 7$ indicates the number of reference point if $C < 68$ it means layout intersection becomes complicated and if $C > 80$ it means standard posture.

2.4 Fill in Fabrics

Fill in fabrics: Body-aware self-supervised inpainting for image-based virtual try-on [5] is a self-supervised conditional generative adversarial network model that can handle the complex pose of a reference person while preserving the target clothing details. It comprised a fabricator and a virtual try-on pipeline with a segmenter, warper, and fuser. Given a partial input, the fabricator uses it to reconstruct the full clothing details and learn the overall structure of the clothing (i.e., the full and half sleeves). The wrapper uses this as a pretext task. Segmenter is used to predict the mask of the referenced person's body parts as well as the masked target clothing regions. The wrapper is used to wrap the target clothing image so that it fits the masked clothing region, intending to capture the reference person's pose and shape. Fuser integrates the outputs from the segmenter and wrapper to synthesize the final try-on image.

2.4.1 Fabricator

It reconstructs the full target clothing image $\hat{T}_c$, given the partial target clothing region $T_{partial}$. It learns to represent the overall structure of the clothing while reconstructing the missing regions (i.e., fill in the correct pixels that make sense in

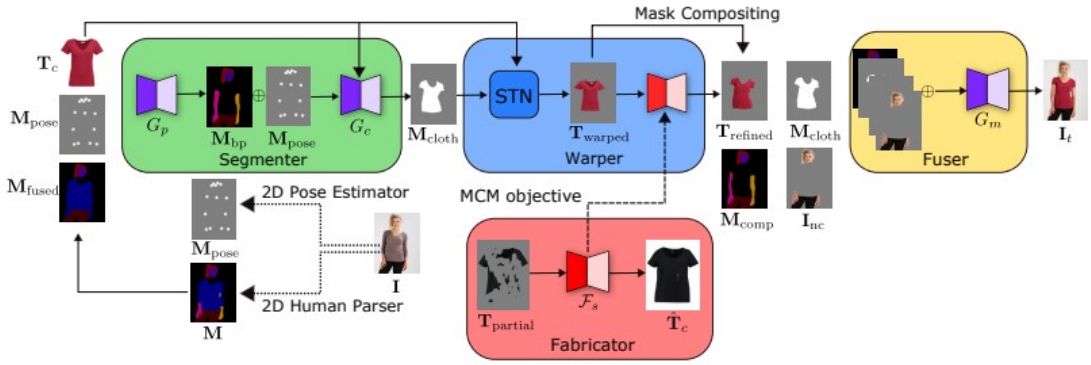


Fig. 2.6 Schematic layout of FIFA framework [5]

the context). It train an encoder-decoder network F_S to reconstruct the reconstructed target clothing $\hat{T}_c$, for a given $T_{partial}$, to yield the original target clothing image T_c by minimizing the L_1 error:

$$\mathcal{E} = \left\| \hat{\mathbf{T}}_c - \mathbf{T}_c \right\| \tag{2.9}$$

2.4.2 Segmenter

The work of the segmenter is to preserve the body parts of the person during the synthesis process and to predict the semantic layout of the target clothing. It uses a publicly available human parser on the referenced person image I, to generate 18-keypoint pose heatmap M_{pose} and its associated mask M. It uses two conditional generative adversarial networks (CGAN) G_p and G_c. First in G_p it takes M_{fused}(the arms and torso regions are merged to form a fused map), M_{pose} and T_c (target cloth) to generate different person body part mask M_{bp}. The M_{bp} (body part mask), M_{pose} and T_c (target cloth) is used as input for G_c to generate target clothing region M_{cloth}. The adversarial loss is given by

$$\mathcal{L}_{\text{CGAN}} = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}(\mathbf{x})}[\log D(\mathbf{x} \mid \mathbf{y})] + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}[\log(1 - D(G(\mathbf{z} \mid \mathbf{y})))]. \tag{2.10}$$

In (2.10) G and D are the generator and discriminator, x and y are the input and ground-truth mask, and z is a noise. It use pixel-wise cross-entropy loss L_{CE} for better semantic segmentation results from generator. Therefore, the overall objective is defined as

$$\mathcal{L}_{\text{mask}} = \alpha_1 \mathcal{L}_{\text{CGAN}} + \alpha_2 \mathcal{L}_{\text{CE}}, \tag{2.11}$$

In (2.11) α_1 and α_2 are non-negative regularization parameters.

2.4.3 Warper

It is used to deform the target clothing region to fit the mask of the clothing region to the person's pose while also preserving the embroidery and texture of the target clothing. It gives T_c (target cloth image) and M_c (target clothing region) as input to STN (spatial transform network) and generates T_{warped}, during the generation process it makes use of the learning from fabricator to generate $T_{refined}$. The overall loss function includes L_{CGAN} (losses for the refinement net-work i.e. pre-trained from the encoder-decoder network F_s), perceptual loss L_{VGG} and $L_{MS-SSIM}$ a multi-scale structural similarity constraint.

$$\mathcal{L}_{\text{refined}} = \beta_1 \mathcal{L}_{\text{CGAN}} + \beta_2 \mathcal{L}_{\text{VGG}} + \beta_3 \mathcal{L}_{\text{MS-SSIM}}\,, \tag{2.12}$$

where β_1, β_2, and β_3 are regularization parameters.

2.4.4 Fuser

The Fuser merges the target clothing region, refined clothing image, composited body part mask, and body part image with the original clothing region masked out to produce the final try-on image. It take four input $T_{refined}$ from wraper, M_{cloth} from segmenter, M_{comp} and I_{nc}.

$$\mathbf{M}_{\text{comp}} = \big(\big(\mathbf{M}_{\text{bp}} \odot \mathbf{M}_{\text{oc}}\big) + \mathbf{M}_{\text{obp}}\big) \odot (\mathbf{J} - \mathbf{M}_{\text{cloth}}) \tag{2.13}$$

$$\mathbf{I}_{\text{nc}} = (\mathbf{I} - \mathbf{M}_{\text{oc}}) \odot (\mathbf{J} - \mathbf{M}_{\text{cloth}}) \tag{2.14}$$

$$\mathcal{L}_{\text{fuser}} = \gamma_1 \mathcal{L}_{\text{CGAN}} + \gamma_2 \mathcal{L}_{\text{VGG}}. \tag{2.15}$$

In (2.13) $\odot$ is an element-wise multiplication, M_{obp} is the original body part mask, M_{oc} is the clothing mask, M_{bp} and M_{cloth} are from segmenter. In (2.14), J is an all-ones matrix, and I is the person image. The overall loss function is given in (2.15), where γ are the hyper-parameters.

2.5 GarmentGAN

Garment GAN: Photo-realistic adversarial fashion transfer [2] is a multi-stage foundation that creates an exact likeness of the appearance of a user wearing a requested

costume, where the preferred costume is fitted seamlessly onto the person's body. In the first stage of this framework, a shape transfer network synthesizes a segmentation map associated with a person wearing the target clothing piece. The product of this network supplies information within the geographic boundaries and body parts that guide the final image's combining. In the second stage, a Thin-plate-spline transmission warps the wanted fabric to address the misalignment between the random body pose and the input clothing part. This warped cloth image is first passed through an appearance transfer network, which generates the final RGB-colorspace image of the reference person wearing the desired cloth preserving fine details in the clothing It handles complex body posture, hand signals, and occlusions with modern strategy of concealing semantic division maps. Settle issues related to the misfortune of target clothing characteristics (e.g. collar shape) amid the article of a clothing exchange.

Chapter 3

Methodology

In this work, first, we try to reconstruct the full clothing details and learn the overall structure of clothing (i.e. full and half sleeve) using a fabricator and an encoder-decoder network. Then we allow the user to input the referenced person image and then used an online available $2D$-Human parser, and $2D$ pose estimator to generate human parsing and an 18-keypoints pose map. After this, we asked the user for the target cloth, which he want to try on the referenced person. Then we trained a conditional generative adversarial network (CGAN) to generate a body part mask and then another CGAN to generate a target clothing region. Then it uses a spatial transformer network to naturally deform or wrap the target clothing onto the target clothing region. For further refinement, it uses learning from a fabricator. In the end, we have a fuser that combined all the outputs generated from the previous module and outputs the final result.

3.1 2D Human parser and pose estimator

In human parsing, we fragment a human picture into semantic parts such as the head, torso, arms, and legs. $2D$ posture estimation appraises the areas of the body joints in $2D$ space. The $2D$ posture gives the X and Y coordinates for each key point. Once the referenced individual picture I is given, we create the human parser M and posture outline M_{pose}.

3.2 GAN Loss

Then two conditional generative adversarial networks (CGAN) G_p and G_c are used to generate the body part mask M_{bp} and target clothing region M_{cloth}. This not only helps retain detailed features and predict accurate body part masks but also helps generate better try-on results. The GAN loss is given by

$$\mathcal{L}_{\text{CGAN}} = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}(\mathbf{x})}[\log D(\mathbf{x} \mid \mathbf{y})] + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}[\log(1 - D(G(\mathbf{z} \mid \mathbf{y})))] \tag{3.1}$$

In (3.1) G and D are the generator and discriminator, x and y are the input and ground-truth mask, and z is a noise. It also uses pixel-wise cross-entropy loss L_{CE} for better semantic segmentation results from a generator. Therefore, the overall objective is defined as

$$\mathcal{L}_{\text{mask}} = \alpha_1 \mathcal{L}_{\text{CGAN}} + \alpha_2 \mathcal{L}_{\text{CE}}. \tag{3.2}$$

Here, α_1 and α_2 are non-negative regularization parameters.

3.3 VGG perceptual loss

The loss of the fabricator module is GAN loss and perceptual loss. VGG perceptual loss ensures the target clothing and its warped version contain the same semantic content. It is a type of content loss and also an alternative to pixel-wise losses. It is based on the ReLU activation layers of the pre-trained 19-layer VGG network. The VGG loss is given by

$$l_{VGG/i.j} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} \left(\phi_{i,j}\left(I^{HR}\right)_{x,y} - \phi_{i,j}\left(G_{\theta_C}\left(I^{LR}\right)\right)_{x,y} \right)^2 \tag{3.3}$$

where W_{ij} and H_{ij} are dimensions of the respective feature maps, I^{LR} is the reference image, G_{θ_C} is the reconstructed image.

3.4 Images Generated in a Sequence using ACGPN model

The proposed work is to generate images in a sequence and check their mean squared error (MSE) which should ideally converge. The images generated on the ACGPN model with initially an input cloth and an image, generate output1, and using input cloth and output1 we generate the output2 and calculate the MSE between them using (4.1). We have taken a 9 observation of images shown in fig. 4.15 and calculate the MSE shown in table 4.3, which shows that the MSE values are not converging. We propose to use the loss to train a model.

Chapter 4

Experiments, Results, and Discussions

4.1 Experimental Design

4.1.1 Experimental Settings

We have used Pytorch, Torchvision, Torchaudio, JupyterLab, OpenCV-Python, Matplotlib, Sklearn, Pycocotools, TensorBoard, PyWavelets, and TensorboardX. We used a large, collected dataset, the "VITON" dataset, consisting of front-view women's images and front-view top clothing images. First, we set up all the dependencies required to train the FIFA Try On model. We get the Vgg19 model for computing VGG-based distance loss while training. The fabricator is then trained with niter and niter decay, with both parameters set to 10. After training the fabricator, we train the model, and here we also set the niter and niter decay values to 10. After training, we generate model weights, which are used for testing. The train dataset consists of $14,221$ images and the test dataset consists of $2,032$ images.

4.1.2 Data Sets

The model is trained using the VITON dataset. It has $19,000$ image pairs, each of which includes a front-view women's image and a top clothing image. After removing the invalid image pairs, it yields $16,253$ pairs, further splitting into training and testing sets.

(a) Input image (b) Input cloth (c) Output cloth

Fig. 4.1 Applying self trained FIFA Try-On model when trained with 100 image dataset.

4.2 Experimental Results

4.2.1 Generation of Image using FIFA-TryOn

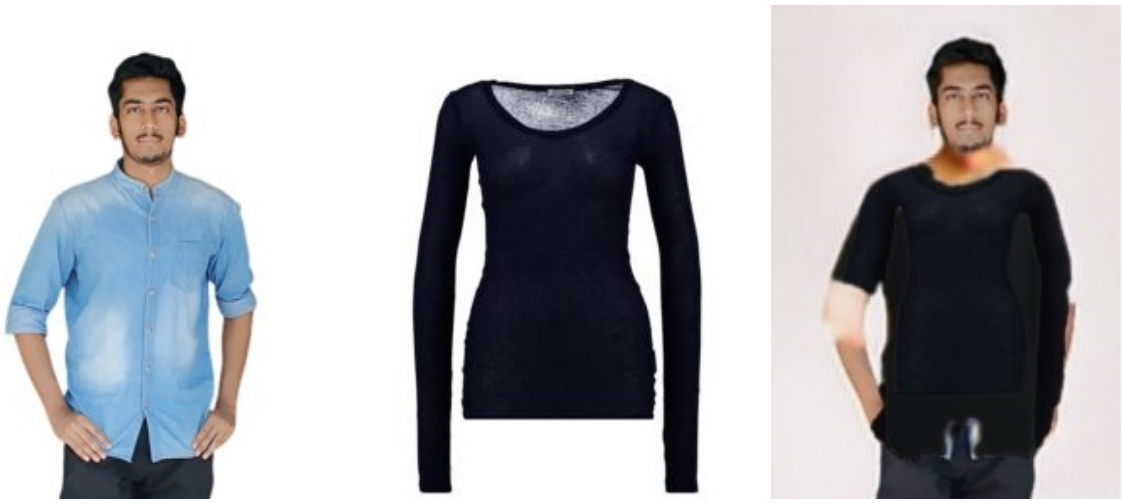
Once the model is train it generates the following weights as names Latest_net_G.pth, Latest_net_G1.pth, Latest_net_G2.pth, Latest_net_U.pth.

4.3 Visual Comparisons

The ACGPN model fig. 4.3 , the FIFA model fig. 4.4, and the trained FIFA fig. 4.5 model are all applied to the same easy-level input sample and input cloth in this case. We can see the ACGPN model has modified the texture (i.e., the collar) of the cloth, while the FIFA model hasn't modified any texture of the cloth.

The ACGPN model fig.4.6 , the FIFA model fig.4.7, and the trained FIFA fig.4.8 model are all applied to the same medium-level input sample and input cloth in this case. We can see the ACGPN model is not able to clearly generate the collar of the cloth as compared to FIFA model, but is able to generate the left arm more accurately than the FIFA model.

The ACGPN model fig. 4.9 , the FIFA model fig. 4.10, and the trained FIFA fig. 4.11 model are all applied to the same hard-level input sample and input cloth in this case. We can see the ACGPN model is able to generate the target image more accurately than FIFA.



(a) Input image (b) Input cloth (c) Output cloth

Fig. 4.2 Applying self trained FIFA Try-On model

(a) Input image (b) Input cloth (c) Output cloth

Fig. 4.3 Applying ACGPN model on easy level sample.

At fig. 4.12 we can clearly see that ACGPN model is able to generate the logo on the input image more accurately than FIFA model.

Structural Similarity Index (SSIM) s used as a metric to measure the similarity between two given images. An SSIM score of 1.00 indicates perfect structural similarity, as is expected out of identical images. Table 4.1 show the SSIM score between ACGPN, FIFA and Self-trained model on VITON-Dataset.

Consider we have m number of models, represented as I_1, I_2, I_3, I_4, ..., I_m and n number of target dresses T_1, T_2, T_3, T_4, ... T_n. Let ϕ denotes the trained pipeline of fill in fabrics model composed of segmenter, wrapper and fuser. We are trying

(a) Input image (b) Input cloth (c) Output cloth

Fig. 4.4 Applying FIFA Try-On model on easy level sample.

(a) Input image (b) Input cloth (c) Output cloth

Fig. 4.5 Applying trained FIFA Try On model on easy level sample.

Method	SSIM
ACGPN	0.845
FIFA	0.886
Self-trained FIFA	0.818

Table 4.1 Performance comparison of ACGPN, FIFA and Self-trained FIFA on VITON Dataset

to train the model according to eq.4.1. Fig.4.13 shows using ACGPN model output image generated with input image1 and input cloth1 and in fig. 4.14 it uses the

(a) Input image (b) Input cloth (c) Output cloth

Fig. 4.6 Applying ACGPN Try On model on medium level sample.

(a) Input image (b) Input cloth (c) Output cloth

Fig. 4.7 Applying FIFA Try On model on medium level sample.

output image1 as an input to generate output image2. The mean squared error value generated by comparing output image1 and output image2 is 9.6.

$$\begin{aligned} \hat{I}_{ij} &= \Phi\left(I_i, T_j\right); \quad i = 1, 2, \ldots, \quad m; j = 1, 2, \ldots, n \\ \mathcal{L} &= \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{n} \left\| \Phi\left(\hat{I}_{ij}, T_k\right) - \hat{I}_{ik} \right\| \end{aligned} \tag{4.1}$$

(a) Input image

(b) Input cloth

(c) Output cloth

Fig. 4.8 Applying self trained FIFA Try On model on medium level sample.

(a) Input image

(b) Input cloth

(c) Output cloth

Fig. 4.9 Applying ACGPN Try On model on hard level sample.

(a) Input image (b) Input cloth (c) Output cloth

Fig. 4.10 Applying FIFA Try On model on hard level sample.

(a) Input image (b) Input cloth (c) Output cloth

Fig. 4.11 Applying self trained FIFA Try On model on hard level sample.

(a) Input image

(b) ACGPN Output cloth

(c) FIFA Output cloth

Fig. 4.12 Applying ACGPN and FIFA model on the input cloth to check the accuracy of logo generation.

(a) Input image

(b) Input cloth

(c) Output cloth1

Fig. 4.13 Generating output image1 using ACGPN model with input cloth1 and input image1.

(a) Output image1 (b) Input cloth (c) Output image2

Fig. 4.14 Generating output image2 using ACGPN model with input cloth1 and output image1.

Input	FIFA	ACGPN
Input1	10.41	9.6
Input2	14:43	13.21
Input3	14:55	20.9
Input4	22	11.87
Input5	19.65	23.19
Input6	27.92	22.12
Input7	21.4	14.96
Input8	14.28	11.4
Input9	23.05	21.18
Input10	22.12	15.05

Table 4.2 Mean Squared Error (MSE) of FIFA and ACGPN models on output image generated obtained with 4.1

Input1	Input2	MSE
Output image1	Output image2	14.23
Output image2	Output image3	9.33
Output image3	Output image4	12.09
Output image4	Output image5	11.16
Output image5	Output image6	14.44
Output image6	Output image7	8.11
Output image7	Output image8	17.22
Output image8	Output image9	12.84

Table 4.3 Mean Squared Error (MSE) of ACGPN models on output image generated obtained with (4.1) on fig. 4.15.

(a) Output image1 (b) Output image2 (c) Output image3

(d) Output image4 (e) Output image5 (f) Output image6

(g) Output image7 (h) Output image8 (i) Output image9

Fig. 4.15 Generating images in a sequence using ACGPN.

Chapter 5

Conclusions and Future Scopes

First, we examine a pre-trained Adaptive Content Generating and Preserving Network (ACGPN) model. Then, we train the Fill In Fabrics (FIFA) model with a small set of 100 images and the entire set of $14,221$ training images. We have observed the structural similarity index measure using 2032 test images when we train the model with the whole dataset. We have now examined the abilities of both the ACGPN and the FIFA models to preserve identities using an identity loss. Suppose the image of a person wearing some dress is P, the image of a dress is D, and $\Phi(P, D)$ denotes the output of a virtual try-on model, where $\Phi(\cdot, \cdot)$ denotes either an ACGPN or a FIFA model. Then, the identity loss is as follows

$$\mathcal{L}_{identity} = ||\Phi(P, D) - \Phi(\Phi(P, D), D)||_2^2 . \tag{5.1}$$

The future work includes:

1. Implementation of the loss measure as defined in (4.1) to train the model.
2. Train the model on Zalando-Dataset, it contains $34,928$ frontal-view human (including man and woman) and clothing.

References

[1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc. 3

[2] Raffiee, A. H. and Sollami, M. (2021). Garmentgan: Photo-realistic adversarial fashion transfer. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 3923–3930. IEEE. 11

[3] Xie, Z., Huang, Z., Zhao, F., Dong, H., Kampffmeyer, M., and Liang, X. (2021). Towards scalable unpaired virtual try-on via patch-routed spatially-adaptive gan. *Advances in Neural Information Processing Systems*, 34:2598–2610. 5, 6, 7

[4] Yang, H., Zhang, R., Guo, X., Liu, W., Zuo, W., and Luo, P. (2020). Towards photo-realistic virtual try-on by adaptively generating↔preserving image content. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7847–7856. 7

[5] Zunair, H., Gobeil, Y., Mercier, S., and Hamza, A. B. (2022). Fill in fabrics: Body-aware self-supervised inpainting for image-based virtual try-on. 1, 9, 10