Generative Fashion for Indian Clothing

Harshil Jain, Rohit Patil, Utsav Jethva, Ronak Kaoshik, Shaurya Agarawal, Ritik Dutta, Nipun Batra
IIT Gandhinagar, Gujarat, India
{jain.harshil,rohit.patil,jethva.utsav,kaoshik.ronak,shaurya.agarawal,ritik.dutta,nipun.batra}@iitgn.ac.in

ABSTRACT

Deep learning-based innovations particularly GANs (Generative Adversarial Networks) have recently been shown to be of great success in the field of Fashion modeling for various use cases such as pose, face generation. A popular work, FashionGAN, is able to generate images with modified clothing as per natural language description. The designers can easily test their clothing ideas on a particular model and fabricate their designs according to the attractiveness. FashionGAN makes use of the DeepFashion dataset which largely contains clothing styles of the Western countries. Currently, no dataset caters to Indian style and clothing. Hence, with this work we present a dataset of 12k images and descriptions pertaining to the Indian culture. We also propose a baseline 2 step GAN model inspired from FashionGAN, for clothing modifications as per natural language description. Deep learning-based innovations in the Indian Fashion context being a relatively new area of research, we hope our work will be a starting initiative towards the same and helps other researchers.

KEYWORDS

Generative Fashion, GAN, Domain Transfer

1 INTRODUCTION

It would be a great idea to visualize the transformation of an outfit on the body of a person to some other outfit using just the description of the expected output. A person could think how he/she would like like by trying various different types of clothing virtually without actually trying each of the out. The primary requirement of such a model is to preserve the structural coherence and at the same time generate images of reasonable quality as the input image. Conforming to requirements such as retainment of shape of the wearer, not limiting the shape of outfit to the clothing which was originally worn by the wearer can be challenging.

Generative Adversarial Networks (GAN) [4] is an appealing architecture to use primarily because of the immense success it has shown in image processing tasks. DCGAN [11] has been used to produce realistic images and it also allowed conditioning of descriptions in the form of text on the image. However, it is not suitable for generative fashion because it doesn't address structural coherence. To overcome this, we have used Wasserstein GAN (WGAN) [1] for our model. We use a two step GAN model - the first being the production of a segmentation map and the second one for rendering the final image with textures as per the description provided by the user.

1.1 Related Work

Generative Adversarial Networks (GAN) [4] have shown impressive results for generation of new images, e.g. faces, indoor scenes, etc. There exist several studies to transfer an input image to a new one. GAN has been used for various tasks like super resolution of an image [8], neural style transfer [6], transferring domains of images [5], etc. We explore a method that uses a segmentation map of the image which is very useful for mapping various body parts and thus producing the final image having all parts replaced correctly as per the textual descriptions. Recently, there have been several studies that have explored improved image generation by stacking GANs on top of one another. [12] use conditional GAN model to perform domain transfer, however, we use textual descriptions to generate the output given the image using the two step GAN architecture.

1.2 Our Contribution

In this paper, we try to develop a generative fashion model on our curated dataset of 12K images for clothing in the Indian context. We have used a two step GAN model - the first being the production of a segmentation map and the second one for final image rendering with textures as per the textual description provided by the user.
Outline: Section 2 describes the dataset we curated for conducting the experiments. Section 3 explains the components from the pipeline and architecture of our model. Section 4 and Section 5.2.2 describes the experimental setup and results.

2 DATASETS

2.1 Dataset Statistics

We collected our dataset from Myntra[1] and Amazon[2] using selenium library. Both the sources are leading fashion e-commerce website in India having millions of cloths listed on their platform. We scraped a total of 12304 images along with their description from the two above mentioned sources. Our dataset consists of 6459 images of Men and 5845 images of women in various outfits. Our dataset has a very little bias in terms of gender at the same time has a very rich distribution of different types of clothing. We collected images with numerous types of clothing style including Indian ethnic wear as well as modern western style. Table 1 extensively describes our dataset along with the statistics.

Men		Women	
Category	Count	Category	Count
Sherwani	404	Kurta-Kurtis	826
Kurta	1766	Sarees	860
Formal	870	Lehenga	936
Casual Shirts	868	Dress	896
T-Shirts	902	Shirt	621
Ethnic Mix	1649	Ethnic Mix	1706
Total	**6459**	**Total**	**5845**

Table 1: Data distribution among various categories for both the genders

[1] www.myntra.com
[2] www.amazon.in

2.2 Dataset Collection

We used selenium webdriver to scrape data from these websites by collecting the links to the images rendered on the their web pages as well as we collected description of those images along with the link. Then we downloaded the images from the collected links. Selenium webdriver came very handy while scrapping the images from Myntra because Myntra does not load below images down the page unless we scroll down. It renders the images only as we scroll down. So we programmed selenium to scroll down the pages and collect the links of the images simultaneously. Example of our dataset can be seen in Figure 2.

Figure 2: Left figure has description: KISAH Men Pink and White Solid Straight Kurta. Right figure has description: Women Navy Blue and Silver Embroidered Straight Kurta

3 METHODOLOGY

Our model is based on Generative Adversarial Network [4]. A general GAN consists of a Generator G and a Discriminator D. They are trained simultaneously with following objective function

$$\min_G \max_D \mathbb{E}_{I\sim p_{\text{data}}}[\log D(I)] + \mathbb{E}_{\mathbf{z}\sim p_z}[\log(1 - D(G(\mathbf{z})))] \quad (1)$$

3.1 Architectural Overview

Our problem statement is the same as the one defined in FashionGAN [2]. We are given a picture of a person and a text description as the Input, we have to generate an image of the same person (with preservation of pose and body structure) wearing the outfit described in the description.

In the training process, the same image acts as Input and output. For the task, we need some extra information about the picture to preserve the overall body structure of the person. Hence we extracted out segmentation maps S_0 of each image I_0 in our database using the existing trained model of Self-correction Human Parsing [9]. These segmentation maps consist of 18 labels like Hair, Face, Upper-clothes, etc. For more details regarding appearance, we converted the description in our dataset to word embeddings w for each image.

Hence, our problem can be described as following, given description w and segmentation map S_0 of the image I_0, we have to generate image I_{out}. We are using the same idea for the architecture as the one described in FashionGAN [2]. As shown in Figure 3, there are two steps for the framework, generating a segmentation image and, from that segmentation image, generate a coloured image of the person. So the two generator are as follows,

$$G_{seg}(z_1, S_{0_{downsampled}}, w_1) \rightarrow \tilde{S} \quad (2)$$

$$G_{img}(z_2, \tilde{S}_{downsampled}, w_2) \rightarrow I_{out} \quad (3)$$

G_{seg} is the generator in step-1 and G_{img} is the generator in step-2, z_1 and z_2 are noise vectors, w_1 and w_2 are text embeddings (they are slightly different, described in next section)

3.2 Generation of updated segmentation map (G_{seg})

The step-1 Generator takes as an input a downsampled segmented image ($S_{0_{downsampled}}$), and text embeddings w_1 and gives an updated segmentation image with changed attributes such as clothes outlines like short/long sleeves, shirt changed to kurta, etc. Here are the details for the same. The input image of dimension (64×64×3) is first converted to segmented image of dimension ($64 \times 64 \times 3$) using human-parsing[ref]. Then the image is downsampled to (30×30×3). We have the text embeddings w_1 (R^{786}) and Noise vector z_1 (R^{100}). Here the text description used for creating embedding does not contain certain attributes like the colour as they are not relevant for this stage. The segmented image is flattened and concatenated vector ($S_{0_{downsampled}}, w_1, z_1$) is given as input to the generator model. The generator gives updated segmented image $\tilde{S}$ ($64 \times 64 \times 3$) as output. The discriminator of this GAN takes the segmented image as input for real samples.

3.3 Generation of coloured image from segmented image (G_{img})

The second stage generator takes a downsampled segmented image and text embedding as Input and generated texture rendered image of the person wearing the described outfit. Same as the previous GAN, the Input segmented image $\tilde{S}$ downsampled to $30 \times 30 \times 3$. We have the text embeddings w_2 (R^{786}) and noise vector z_2 (R^{100}). Similar to the previous model, the concatenated vector ($\tilde{S}_{downsampled}, w_2, z_2$) is given input to the GAN to generate texture rendered image I_{out}. The discriminator takes real coloured images as real Input for training purpose.

4 EXPERIMENTS

4.1 Model Architecture

We conducted few experiments for figuring out the best architecture and embeddings. First we tried using one-step GAN architecture. In one-step GAN architecture, we simply give our image and text embedding as our input to the generator WGAN and WGAN generates the new image according to the new description text embeddings.

We did not get good results from one-step GAN architecture, so used two-step GAN inspired from FashionGAN paper. In two-step GAN architecture, during the first step first generator WGAN generates a new segmentation map according to text embeddings of new description without color and then in the second step second generator WGAN maps the color on the new segmentation map created in the first step using text embeddings of new description

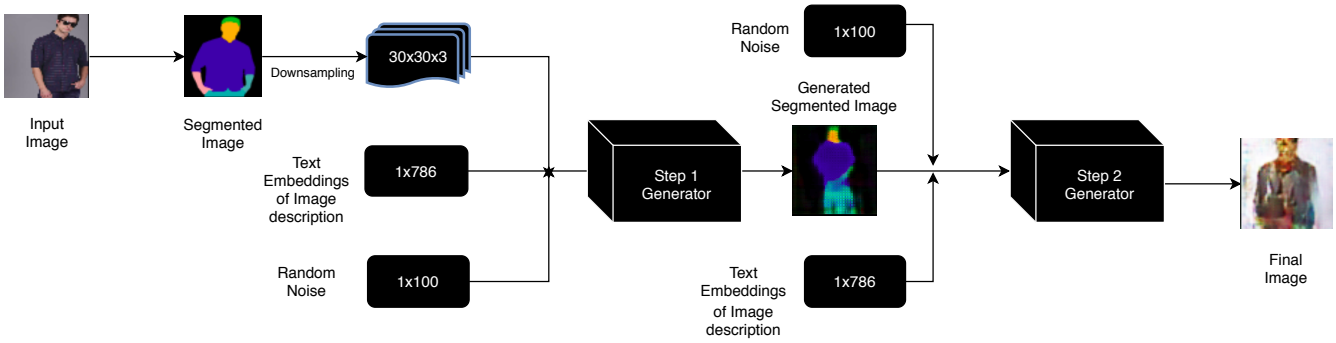

Figure 3: Proposed Architecture

with color. We trained both stages separately on the dataset of same set of 10000 images for 80 epochs.

4.2 Text Embedding

We used three different types of embeddings during our experiments. We used Doc2Vec [7], BERT [3] and TF-IDF one-hot vector embeddings [10] in our experiments. We started with using Doc2Vec embeddings in the initial stages of our experiments and then switched to BERT embeddings for getting better results. But After going through some more related research papers, we found out that one-hot encoding gives better results in this type of problem statement. So we used TF-IDF one-hot vector embeddings and we got our best results with TF-IDF one hot embeddings.

5 RESULTS

5.1 Results from Single Step GAN

Figure 4 shows the generated samples while training the single step GAN.

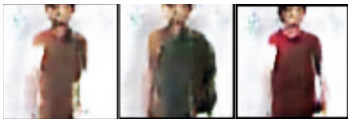

Figure 4: Results from Single Step GAN

Here the body structure of the person is not preserved. Also the results were not affected by the text embeddings provided at input. Hence we moved to implementing two-step architecture.

5.2 Results from the Two-Step Framework

5.2.1 *Step-1 (G_{seg}) Results.*

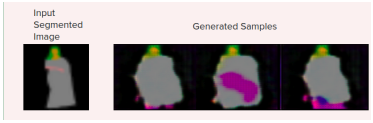

Figure 5: Results from the First Step

Here, as shown in Figure 5, we can see that the body structure is preserved, and the outfit of the person is affected as required.

5.2.2 *Step-2 (G_{img}) Results.*

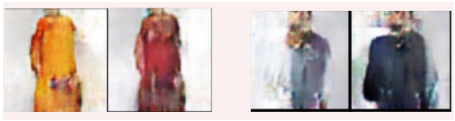

Figure 6: Results from the Second Step

Here, as shown in Figure 6, we can see that different textures are applied to same person. Though we were not able to achieve quality results from the step-2 of our architecture, the texture rendering is in somewhat correlation with the input text embeddings.

5.2.3 *End to end pipeline of our architecture.*

As shown in Figure 7, the results shown by two step architecture are better in terms of body shape preservation and texture variety. First image is input image along with the text which will be converted to embedding vectors, second image shows extracted segmentation maps from human-parsing [9], third image is the output of step-1,which then is fed as input to step-2 and the fourth image is the final result of the second stage.
Out first step of GAN (generation of segmented images) is working satisfactorily. Upon more training with increased complexity of architecture, better results could be achieved.

6 CONCLUSION

FashionGAN has a lot of real-world applications. It can be used to "create" new dresses or "recolour" existing dresses or "redressing" a model just based on the text description. This can reduce the dependence on fashion designers as well as models, thus reducing the price of clothes. Our model, based on training on 80 epochs is able to generate clothes and some of the parts of models' body, but due to lack of resources available, we were unable to train it further Our future works will include the proposal of new loss functions which can express our model's generative capabilities. We will

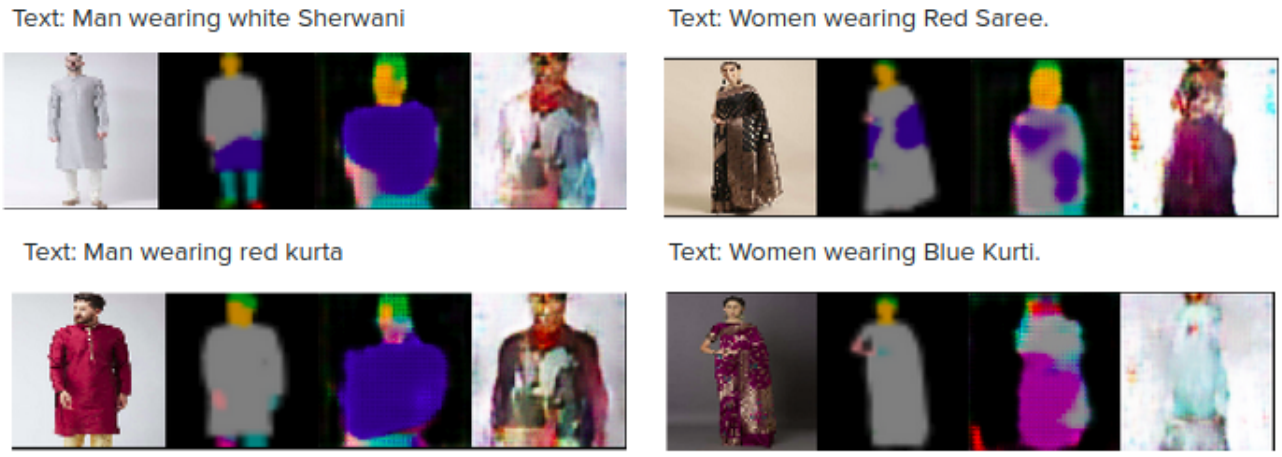

Figure 7: End to End pipeline: 4 figures in order are the input image, segmentation map from human parsing, output from Step 1 GAN and output from Step 2 GAN respectively.

also try to the opinions of general end-users and fashion designers on the clothing designs produced by our model, which will make our point that our generative model is able to produce human-like and human-wearable designs. We will also try to compare results with the current state of the art.In our future works, we include the collection of the large dataset which would be the benchmarks in this line of research. We collected a dataset of 12, 300 images which is very less as compared to FashionGAN's 80, 000 images. The data that represents different clothing styles of each part of India needs to be collected because, as of now the data is highly biased towards North Indian styles due to the biased present in the data source, the e-commerce sites. Hence new data sources need to be searched. Different detection and segmentation methods should be used to detect and segment different body parts of models as well as different clothes. The collected text data also needs augmentation which we will try to do in our future works.

REFERENCES

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein GAN. (2017). arXiv:stat.ML/1701.07875

[2] Yi Rui Cui, Qi Liu, Cheng Ying Gao, and Zhuo Su. 2018. FashionGAN: Display your fashion design using Conditional Generative Adversarial Nets. *Computer Graphics Forum* (2018). DOI: http://dx.doi.org/10.1111/cgf.13552

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. DOI: http://dx.doi.org/10.18653/v1/N19-1423

[4] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 2672–2680. http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[5] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. 2016. Image-to-Image Translation with Conditional Adversarial Networks. *CoRR* abs/1611.07004 (2016). arXiv:1611.07004 http://arxiv.org/abs/1611.07004

[6] Justin Johnson, Alexandre Alahi, and Fei-Fei Li. 2016. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. *CoRR* abs/1603.08155 (2016). arXiv:1603.08155 http://arxiv.org/abs/1603.08155

[7] Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. *CoRR* abs/1405.4053 (2014). arXiv:1405.4053 http://arxiv.org/abs/1405.4053

[8] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. 2016. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. *CoRR* abs/1609.04802 (2016). arXiv:1609.04802 http://arxiv.org/abs/1609.04802

[9] Peike Li, Yunqiu Xu, Yunchao Wei, and Yi Yang. 2019. Self-Correction for Human Parsing. (2019). arXiv:cs.CV/1910.09777

[10] Shahzad Qaiser and Ramsha Ali. 2018. Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents. *International Journal of Computer Applications* 181 (07 2018). DOI: http://dx.doi.org/10.5120/ijca2018917395

[11] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. (2015). arXiv:cs.LG/1511.06434

[12] Donggeun Yoo, Namil Kim, Sunggyun Park, Anthony S. Paek, and In-So Kweon. 2016. Pixel-Level Domain Transfer. *CoRR* abs/1603.07442 (2016). arXiv:1603.07442 http://arxiv.org/abs/1603.07442