Indian Institute of Technology Jodhpur

Course: Deep Learning

Course Code: CSL4020

Project Report

Style Transfer in Fashion Images using Cloth Segmentation and Salience maps

Team Members:

Atharva Date (B22AI045)
Singamsetti Manikanta Varshit (B22AI038)
Chittiprolu Bhala Vignesh (B22AI015)
Shashwat Meena (B22BB037)

Contents

1 Introduction

Style Transfer in Fashion Images enables users to visualize how clothing items would look on them using deep learning models. This project applies neural style transfer to clothing images, preserving texture and structure while seamlessly blending stylized garments with real-world images. The system leverages advanced deep learning architectures and image processing techniques to achieve high-quality, visually appealing results suitable for real-time applications.

2 Methodology

2.1 Pipeline Overview

The overall pipeline consists of three main steps:

1. **Saliency Map Generation and Cloth Segmentation**
2. **Neural Style Transfer**
3. **Blending the Stylized Image**

Figure 1 illustrates the entire pipeline.

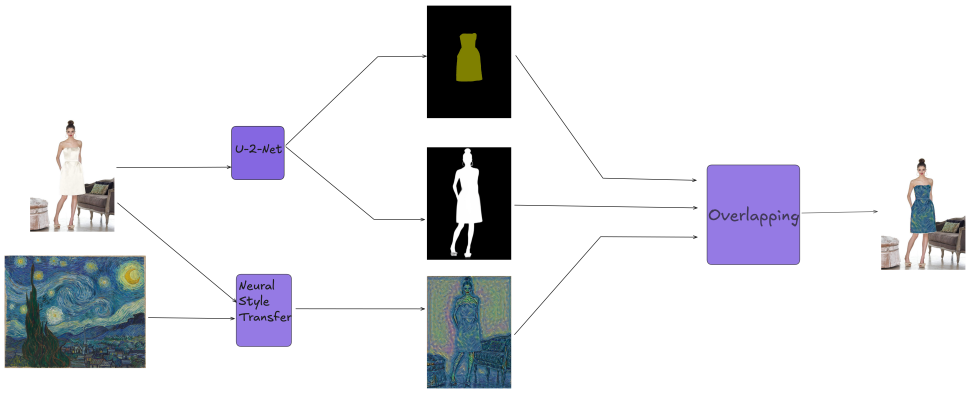

Figure 1: Overview of the pipeline

2.2 Saliency Map Generation and Cloth Segmentation

This stage involves generating a saliency map to highlight prominent clothing regions and segmenting the cloth components from the input image. We utilize variants of the U-2-Net model, designed for salient object detection.

2.2.1 Model Architecture

The U-2-Net (U-square Net) features a nested U-structure composed of Residual U-blocks (RSU). Each RSU, similar to a mini U-Net, uses residual connections for multi-scale feature extraction:

- **Segmentation Model**: Configured with 3 input channels (RGB) and 4 output channels for four classes: background, upper-body clothing, lower-body clothing, and full-body clothing.
- **Saliency Model**: Configured with 3 input channels and 1 output channel, generating a single-channel probability map.

2.2.2 Implementation Details

- **Cloth Segmentation**: The model is loaded from the checkpoint `cloth_segm_u2net_latest.pth` using PyTorch. Input images are resized to 768x768 pixels, normalized, and processed to generate a segmentation mask.
- **Saliency Map Generation**: The saliency model, loaded from `saved_models/u2net/u2net.pth`, processes resized images (320x320 pixels) to produce a normalized saliency map, which is then resized to the original dimensions.

Figure 2: Saliency map generation and cloth segmentation

2.3 Neural Style Transfer

The segmented clothing regions are transformed using neural style transfer to impart artistic styles.

2.3.1 Model Architecture

The style transfer is based on the VGG-19 architecture, a deep convolutional neural network used to extract content and style features from images. The architecture consists of:

- extbf16 convolutional layers: Each followed by a ReLU activation function.
- extbf5 max-pooling layers: Used for spatial downsampling.
- extbfFeature extraction: Only the convolutional layers are used, while the fully connected layers (classifier) are discarded.

The selected layers for content and style representation are:

- extbfContent Features: Extracted from deeper layers (e.g., extttconv4_2).
- extbfStyle Features: Extracted from multiple layers (extttconv1_1, extttconv2_1, extttconv3_1, extttconv4_1, extttconv5_1).

2.3.2 Implementation Details

- **Input Processing**: Images are loaded using PIL.Image, converted to tensors with values between 0 and 1, and resized to 512x512 (or 128x128 if using CPU). Normalization is applied using:

```
cnn_normalization_mean = torch.tensor([0.485, 0.456, 0.406])
cnn_normalization_std = torch.tensor([0.229, 0.224, 0.225])
```

- **Loss Functions**:
  - **Content Loss**: Calculated as Mean Squared Error (MSE) between the feature representations of the content image and the generated image at a selected layer.
  - **Style Loss**: Computed using the Gram matrix, which captures correlations between feature maps. The loss is calculated as:

```
def gram_matrix(input):
    a, b, c, d = input.size()
    features = input.view(a * b, c * d)
    G = torch.mm(features, features.t())
    return G.div(a * b * c * d)

class StyleLoss(nn.Module):
    def __init__(self, target_feature):
        super(StyleLoss, self).__init__()
        self.target = gram_matrix(target_feature).detach()

    def forward(self, input):
        G = gram_matrix(input)
        self.loss = F.mse_loss(G, self.target)
        return input
```

- **Model Selection**: VGG-19 is loaded using:

```
cnn = vgg19(weights=VGG19_Weights.DEFAULT).features.eval()
```

  Only the convolutional layers are used for feature extraction.

- **Optimization**: The input image (instead of model parameters) is optimized using L-BFGS or Adam, with the loss function:

$$\mathcal{L} = \alpha \cdot \mathcal{L}_{content} + \beta \cdot \mathcal{L}_{style} \tag{1}$$

  where α and β control the balance between content and style.

Figure 3: Neural Style Transfer

2.4 Blending the Stylized Image

The final step integrates the stylized clothing region into the original image using a saliency-based blending technique.

2.4.1 Blending Process

The blending operation is performed pixel-wise:

$$C_{\text{final}}(x, y) = M(x, y) \cdot S(x, y) + (1 - M(x, y)) \cdot C(x, y) \tag{2}$$

where:

- $M(x, y)$ is the saliency map value (normalized between 0 and 1),
- $S(x, y)$ is the stylized pixel,
- $C(x, y)$ is the original content pixel.

This approach ensures that areas with high saliency values take the stylized appearance while areas with lower saliency preserve more of the original content.

Figure 4: Blending of the stylized image

2.5 Implementation Environment

The project was developed in Google Colab using Python:

- **Frameworks**: PyTorch for U-2-Net models (Cloth Segmentation and Saliency Maps) and VGG-19 for style transfer.
- **Libraries**: `torch`, `torchvision`, `PIL`, and `numpy`.

To access the model weights, use this link: `https://drive.google.com/drive/folders/1FswqwdYqUvuS_kFIk-BnBK2nBh2MYrAY?usp=sharing`

3 Results

The approach was evaluated based on the visual quality of the final blended image. The saliency-based blending method significantly enhances clothing boundaries, ensuring a natural transition between the stylized garment and the background.

Figure 5: Example: Full pipeline applied to another image

4 Conclusion

This project presents a robust high-resolution virtual try-on system leveraging deep learning. By integrating U-2-Net for precise cloth segmentation and saliency mapping, fast neural style transfer for real-time stylization, and a saliency-based blending method for natural integration, our approach produces high-quality results suitable for applications like virtual try-ons and fashion design previews. Future work will focus on optimizing real-time performance and enhancing segmentation accuracy for complex clothing patterns.

References

[1] U-2-Net Model: (Used for Cloth segmentation and Saliency Map)
Code: https://github.com/xuebinqin/U-2-Net
Paper: https://arxiv.org/abs/2005.09007

[2] DataSet (Used to train our U-2-Net Model):
https://www.kaggle.com/c/imaterialist-fashion-2019-FGVC6/data

[3] Neural Style Transfer:
https://pytorch.org/tutorials/advanced/neural_style_tutorial.html

[4] Saliency-Guided Image Style Transfer:
https://ieeexplore.ieee.org/abstract/document/8794904