Fashion Style Transfer

Group Name: Nash

Members: Shreya Yadav (2003134), Yuvraj Nagar (2003138)

Course: Deep learning with Computer Vision applications (CS 435)

Supervisor: Dr. Shitala Prasad

Table of Contents

Abstract

Fashion style transfer, inspired by the concept of neural style transfer, is a cutting-edge technique that enables the transfer of visual styles, textures, and patterns from one fashion image to another. Unlike conventional methods, which lack user control over local transfer positions and struggle to preserve clothing shapes accurately, our method empowers users to selectively transfer styles to specific clothing items. Moreover, we introduce an innovative outline loss function based on distance transform techniques, ensuring the faithful preservation of clothing shapes during style transfer. The proposed method leverages the power of deep learning, particularly VGG-19, to revolutionize fashion style transfer. Through a detailed analysis of existing approaches and practical demonstrations, this report provides insights into the capabilities, challenges, and future directions of fashion style transfer technology.

Introduction

1.1 Overview

This project addresses the need for localized control in clothing style transfer, allowing users to blend the style of one image with the shape of another to create unique garment designs. Users select a base clothing image and a style image, and our algorithm combines them while preserving the original garment shape. To achieve this, we introduce an outline image extracted using an interactive GrabCut algorithm, enabling precise selection of clothing regions. An outline loss function, informed by distance transform, ensures accurate preservation of clothing shapes, while total variation regularization smoothens boundaries. Our method empowers users to control style transfer locations interactively, ensuring that new styles are applied only to desired clothing parts, thus retaining original garment shapes while introducing fresh designs.

1.2 Neural Style Transfer

Neural style transfer is a deep learning technique that merges the content of one image with the style of another to create visually appealing artworks. It uses pre-trained convolutional neural networks, like VGG19, to extract both content and style features from input images. By minimizing the content and style differences between a content image and a style image, a new image is generated that retains the overall structure of the content image while adopting the artistic style of the style image. This process involves optimizing a loss function that balances content and style reconstruction, resulting in images that exhibit the content of one image and the artistic style of another.

1.3 Why use VGG19?

The choice of VGG19 for neural style transfer in fashion stems is because of its well-established performance in image feature extraction and representation learning. VGG19, with its deep architecture and pre-trained weights on large-scale image datasets, captures intricate patterns and textures effectively, making it suitable for discerning style nuances in fashion images. Additionally, VGG19 strikes a balance between model complexity and computational efficiency, enabling efficient style transfer without excessive computational overhead. Its widespread adoption and availability in popular deep learning frameworks further facilitate seamless integration and experimentation in fashion style transfer applications.

Background Research

2.1 CNN

A CNN is a specialized deep learning model for image analysis. It learns features from raw images, starting with basic shapes and gradually capturing more complex patterns. Using convolutional and pooling layers, CNNs efficiently extract spatial features. Activation functions introduce non-linearities, enabling complex learning. Fully connected layers at the end perform high-level reasoning. Through backpropagation, CNNs are trained to minimize prediction errors, making them effective for tasks like image classification. Example of CNN is depicted in Figure 1.

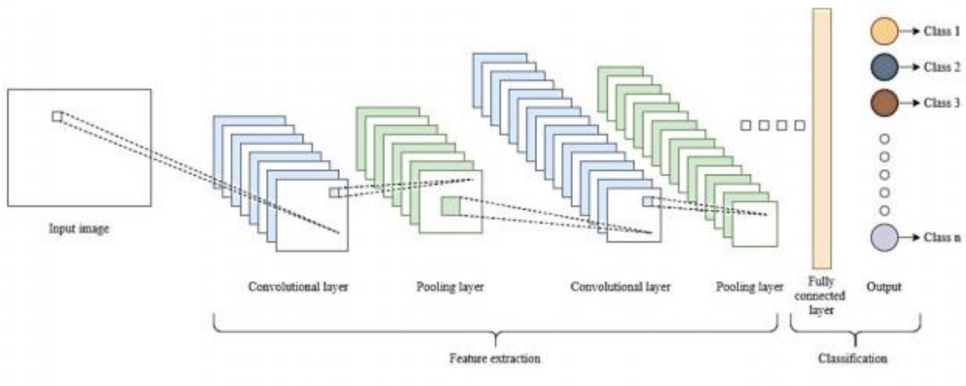

Figure 1: CNN Architecture

2.2 Neural Style Transfer

The fundamental idea of Neural Style Transfer is straightforward: minimize two distinct losses. The initial loss quantifies the disparity in content between the content image and the target image, while the subsequent loss quantifies the disparity in style between the style image and the style of the input image.

Figure 2: Example of Neural Style Transfer

2.3 Algorithm for fashion style transfer

Algorithm for fashion style transfer: Modified version of Neural Style Transfer algorithm

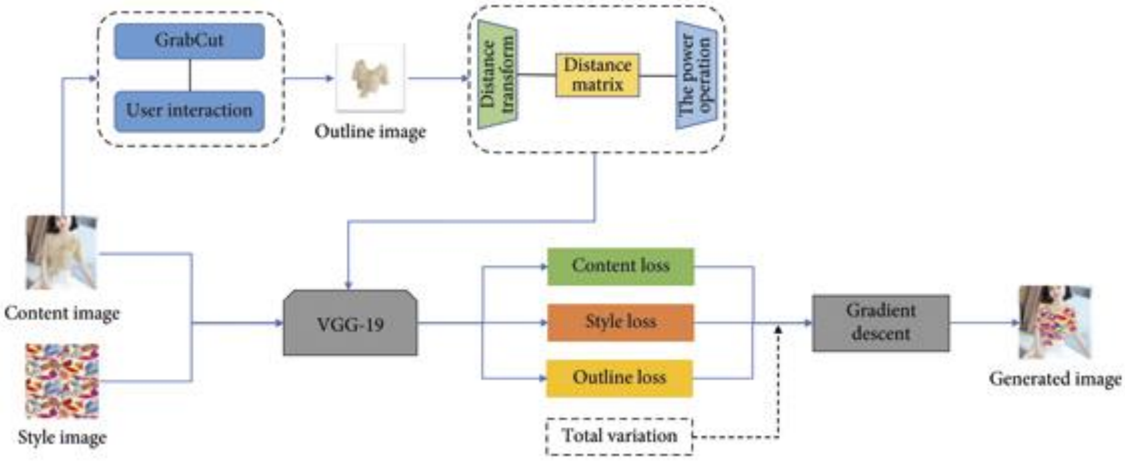

Figure 3: Framework implemented in our project

Step 0: Importing modules

Step 1: Taking inputs: Content and Style Image

Step 2: Outline Image Generation: Employ the GrabCut algorithm to segment clothing from content image

a. **Distance Transform:** Distance transform is applied to generate spatial relationship of pixels.

b. **Outline Feature Enhancement:** Enhances the outline feature by applying pixelwise power to the distance transform matrix.

Step 3: Loss Calculation: Calculate the content, style and outline loss based on their respective features

a. **Feature Extraction:** VGG-19 is used to extract and store features.

b. **Total loss computation:** Combine all the losses into a total loss function, incorporating total variation regularization to encourage smoothness and coherence in generated image.

Step 4: Gradient Calculation: L-BFGS optimizer for loss minimization.

a. **Style Transfer:** Apply optimization algorithm to minimize the total loss function and generate a new image that combines the desired style from the style image with the clothing shape from the content image

Step 5: Creating final image: Created using cv operations on base image, stylized image from the previous step and our outline image

Design

3.1 Implementation

Primary goal of this project is to apply style image transfer on the base/content image using generated outline image (with distance transform). For extraction of features VGG19 is used. While VGG19 is primarily designed for image classification tasks, our focus diverges from classification to image transformation. Therefore, we have selectively omitted layers associated with classification, as our goal centres on image transformation rather than categorization.

The following tools were used for implementing the project:

- Kaggle
- Python
- Packages: Keras, Numpy, Pandas, OpenCV (cv2)

VGG-19 model is used here with IMAGENET weights.

3.2 VGG-19 Architecture

VGG19 is a model, with weights pre-trained on **ImageNet**. **ImageNet**, is a dataset of over 15 millions labelled high-resolution images with around 22,000 categories. ILSVRC uses a subset of ImageNet of around 1000 images in each of 1000 categories. In all, there are roughly 1.3 million training images, 50,000 validation images and 100,000 testing images.

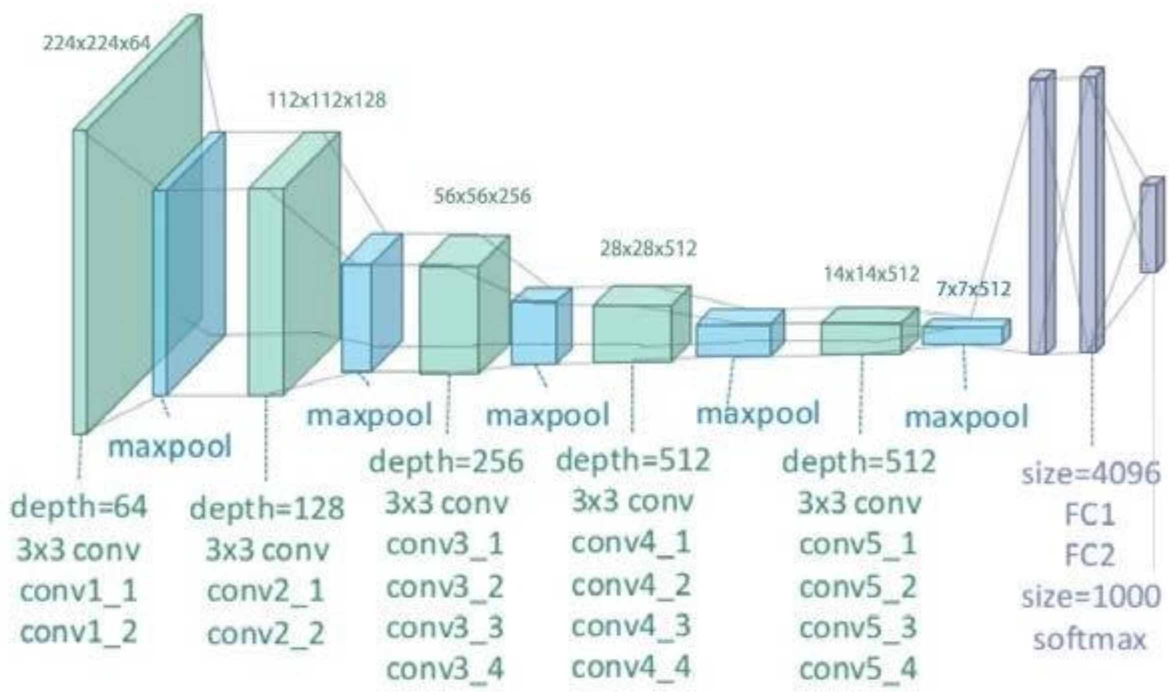

Figure 4: VGG-19 architecture with its layers mentioned below

- **Input:** The input to the VGG-19 network is an image typically represented as a 3-dimensional array of pixel values (standard 224 X 224).
- **Convolutional Layers:** The network contains 16 convolutional layers (contains 3 extra convolutional layers as compared to VGG-16). VGG's convolutional layers employ a compact receptive field of 3x3, capturing both horizontal and vertical movements in the image. Moreover, these layers utilize 11 convolution filters to linearly transform the input. Following this, a Rectified Linear Unit (ReLU) is applied. The convolution stride is fixed to 1 pixel so that the spatial resolution is preserved after convolution.
- **Hidden Layers**: ReLU is extensively utilized in the hidden layers of the VGG network. Local Response Normalization (LRN), although an option, is generally avoided due to its tendency to augment memory consumption and training duration, without a significant enhancement in overall accuracy.
- **Fully Connected Layers**: The VGGNet consists of three fully connected layers. The initial two layers comprise 4096 channels each, while the third layer encompasses 1000 channels, representing one channel for each class.

Project Implementation

We have used 'keras' deep learning framework for implementing this project. We also have used downloaded VGG19 weights for feature extraction.

Hyperparameters used in this project:

- # epochs: 100
- Input shape: (400, 400, 3)
- Optimizer: L-BFGS

4.1 Inputs

There would be two inputs to our project: Content Image and Style Image.

Content Image: The original image of the clothing item whose style is to be transferred.

Style Image: The image containing the desired artistic style to be applied to the clothing.

Figure 5: Content Image(Left) and Style Image(Right)

4.2 Outline Image Generation

The GrabCut algorithm is utilized in this project to enable precise control over the local transfer position of an image. Unlike other segmentation methods, GrabCut requires minimal user interaction, typically involving the simple action of dragging a rectangle around the specific clothing area for style transfer. This rectangle delineates the known background pixels outside and the unknown pixels inside, imposing essential constraints on segmentation. Subsequently, a model is employed to classify the unknown pixels as either foreground or background.

Figure 6: Outline mask image after applying GrabCut on Content Image

4.3 Loss Calculation

Neural Stye Transfer, a technique that combines content and style features from input images using a pretrained convolutional neural network (CNN) to generate new images. This project focuses on applying this technique to fashion design, incorporating content images with complex backgrounds, outline images extracted via GrabCut, and style images. Through a CNN, the content, style, and outline features are integrated to produce stylized clothing images that preserve the original clothing shapes while incorporating the stylistic elements from the style images. The VGG-19 network model is utilized as the primary feature extractor in this process. VGG-19 feature layers that are used: block5_conv2, block1_conv1, block2_conv1, block3_conv1, block4_conv1, block5_conv1.

The total loss is calculated by summing up content loss, style loss and outline loss multiplied by their corresponding weights.

- **Content Loss**: It measures the difference in content between the content image and the generated image by comparing their feature representations at a certain layer of the VGG19 network.
- **Style Loss**: It quantifies the difference in style between the style image and the generated image. This is achieved by comparing the Gram matrices of their feature representations across multiple layers of the VGG19 network.
- **Outline Loss**: This loss is introduced to preserve the shape of clothing in the generated image. It is calculated based on the distance transform of the outline image extracted from the base image, ensuring that the outline shape is retained in the final output.

The total variation loss is calculated to smooth and denoise the boundary regions of the combination image, ensuring better visual coherence.

4.4 Gradient Calculation

The process involves initializing the combination image and iteratively updating its pixel values to minimize the total loss. At each iteration, the gradient of the total loss with respect to the combination image is computed using automatic differentiation provided by TensorFlow. The gradients are then used to update the pixel values of the combination image in the direction that reduces the loss. This update is repeated for 100 iterations, using a L-BFGS optimization algorithm. By iteratively adjusting the pixel values of the combination image, the optimization process gradually generates an image that preserves the content of the content image, incorporates the style of the style image, respects the outline of the outline image, and maintains smoothness through total variation regularization.

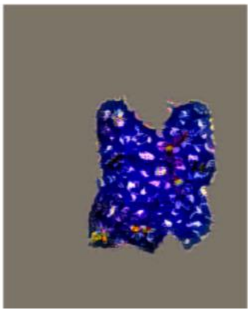

Figure 7: Style transfer on the masked part

4.5 Creating final image

This process involved applying the mask to isolate specific regions of interest from the content image. By bitwise AND between the content image and the inverse mask image, we retained only the pixels corresponding to the background regions and making cloth region 0. Blending the stylized image onto those areas with value 0 while preserving the rest of the image unchanged. The result is an image where the style transfer is applied only to the cloth region, while the rest of the content image remains unchanged.

Figure 8: Final image

Novelty

The novelty of this project lies in its approach to fashion style transfer, particularly in enabling users to exert precise control over the transfer of style to specific clothing regions. By introducing an interactive outline image extraction process and incorporating a customized loss function based on distance transforms, the project achieves the preservation of clothing shape while applying the desired style. This novel method empowers users to seamlessly blend artistic styles with clothing designs, fostering creativity and customization in the fashion industry.

Figure 9: Fashion style transfer implemented by Mishahal Palakuniyil

Figure 10: Fashion style transfer through our implementation

Conclusion

This project presents an innovative approach to fashion style transfer, allowing for the precise integration of artistic styles with clothing designs. By leveraging interactive GrabCut algorithm and distance transform-based outline extraction, the project enables users to selectively apply style to specific clothing regions while preserving the original shape. The utilization of VGG-19 network as an image feature extractor further enhances the accuracy and efficiency of style transfer. Through experimentation and implementation, the project demonstrates the effectiveness of the proposed method in generating unique and visually appealing clothing designs. Moving forward, this research opens avenues for further exploration and development in the realm of fashion technology, offering new opportunities for creativity and customization in the industry.

References

- Research Paper: https://www.hindawi.com/journals/cin/2020/8894309/
- Code reference for project: https://www.kaggle.com/code/basu369victor/style-transfer-deep-learning-algorithm
- CNN architecture (Figure 1): https://vitalflux.com/cnn-basic-architecture-for-classification-segmentation/
- Neural Style Transfer: https://medium.com/@ferlatti.aldo/neural-style-transfer-nst-theory-and-implementation-c26728cf969d
- Framework (Figure 3): https://www.hindawi.com/journals/cin/2020/8894309/fig1/
- VGG-19 architecture(Figure 4): https://www.researchgate.net/figure/llustration-of-the-network-architecture-of-VGG-19-model-conv-means-convolution-FC-means_fig2_325137356
- VGG layers: https://medium.com/@siddheshb008/vgg-net-architecture-explained-71179310050f
- Novelty(Figure 9): https://github.com/thebadcoder96/StyleTransfer