Neural Style Transfer App

Shashank Yadav (22123041)

Overview :

Neural Style Transfer (NST) represents a sophisticated optimization approach designed to merge two distinct images: a content image conveying subject matter and a style reference image, perhaps an iconic artwork. The goal is to synthesize an output image that retains the content of the original while emulating the artistic style of the reference image.

This project harnesses the power of a pre-trained VGG19 model, leveraging its ability to extract nuanced features of both content and style. Through iterative adjustments, the content image gradually adopts the stylistic essence encapsulated within the reference image. Facilitating seamless interaction, the application is crafted with Streamlit, a versatile Python library renowned for its capability to effortlessly create and distribute bespoke web applications tailored for machine learning and data science endeavors.

Deployment :

You can access the deployed version of the app using the following link:

https://neural-style-transfer-charizard001.streamlit.app/

Video Demo :

Watch the video demo to see the app in action:

https://www.youtube.com/watch?v=ZNvfwTfchpM

Features :

- **Upload a content image and a style image :** Easily upload images directly from your device.
- **Generate a stylized image :** Combine the content of the content image with the style of the style image using neural style transfer.
- **Display the stylized image :** View the resulting image directly within the web application.

Model Details :

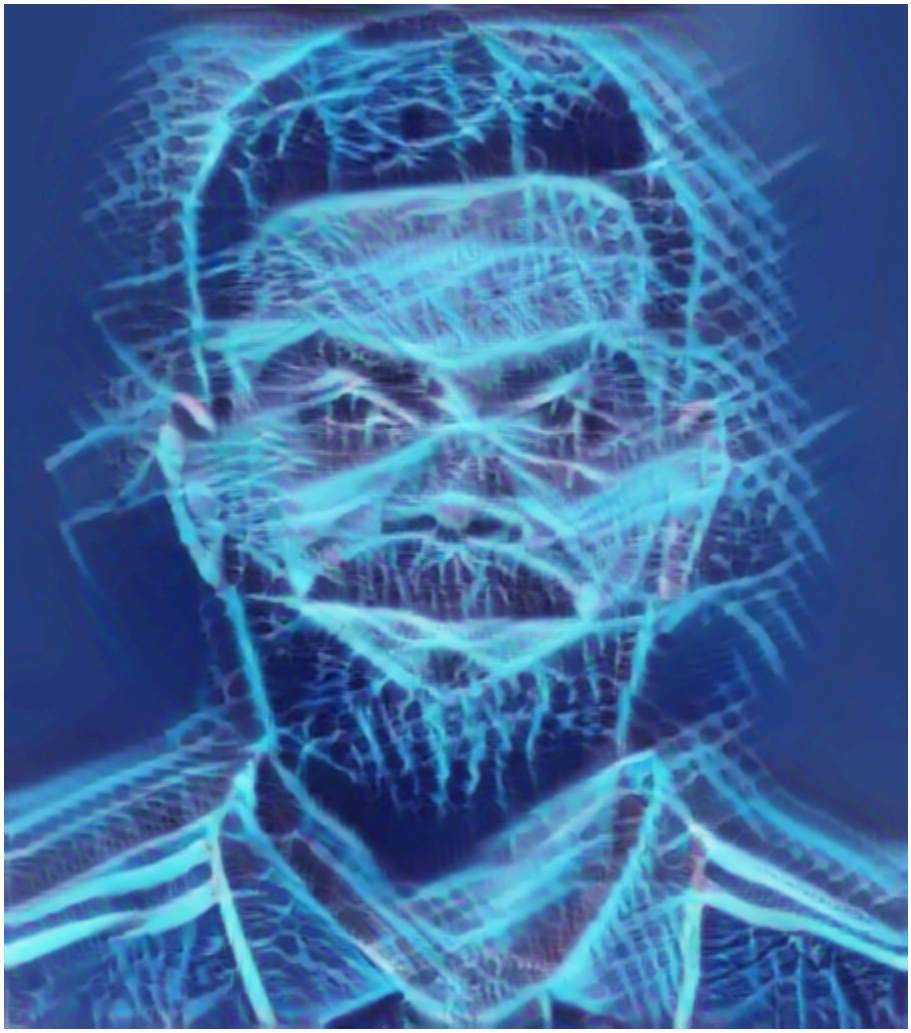
The neural style transfer model uses a pre-trained VGG19 network to extract features from the content and style images. It optimizes the generated image to match the content features of the content image and the style features of the style image. The optimization is done using gradient descent.

The model consists of the following key components:

- `tensor_to_image`: Converts a tensor to a PIL image.
- `load_img` and `load_preprocess_img`: Load and preprocess images.
- `deprocess`: Converts processed images back to their original form.
- `vgg_layers`: Extracts features from the VGG19 model.
- `get_content_loss`, `gram_matrix`, `get_style_loss`: Computes content and style losses.
- `get_feature_representations`: Extracts features from content and style images.
- `compute_loss`, `compute_grads`: Computes the total loss and gradients.
- `run_style_transfer`: Main function to perform style transfer.

Example Results :

Here are some examples of content images, style images, and the resulting stylized images:



Installation :

1. Clone the repository:

```
git clone https://github.com/yourusername/neural-style-transfer-app.git
cd neural-style-transfer-app
```

2. Install the required dependencies:

```
pip install -r requirements.txt
```

3. Run the Streamlit application:

```
streamlit run streamlit_app.py
```

Usage :

1. Open your web browser and go to `http://localhost:8501`.
2. Upload a content image and a style image using the file uploader.
3. Click the "Generate Stylized Image" button to perform neural style transfer.

4. The stylized image will be displayed on the page.

Conclusion :

This project showcases the impressive potential of neural style transfer, blending artistic styles with photographic content to create visually striking images. Utilizing the pre-trained VGG19 model alongside TensorFlow's optimization routines, the application delivers remarkable results. With a Streamlit interface, it remains accessible and user-friendly, inviting users of all skill levels to explore the fascinating world of neural style transfer.

Future Recommendations :

For future improvements and features, consider the following recommendations:

- **Adjustable parameters:** Allow users to adjust parameters such as content weight, style weight, and the number of iterations directly from the UI.
- **Multiple styles blending:** Enable blending of multiple style images with the content image.
- **Real-time processing:** Implement optimizations for faster processing and real-time previews.
- **Enhanced UI/UX:** Improve the user interface and experience with more interactive features and better error handling.
- **Deploy online:** Deploy the application on a cloud platform like Heroku or AWS to make it accessible to a wider audience.

Acknowledgements :

This project utilizes the following resources and libraries:

https://www.tensorflow.org/
https://www.streamlit.io/
https://keras.io/api/applications/vgg/