PROJECT ABSTRACT

Neural Style Transfer App using PyTorch

A PyTorch neural style transfer app that restyles user photos in the manner of famous paintings, delivered as a built-to-order student project with code, trained weights, report, PPT and viva kit.

01. Title

Neural Style Transfer App using PyTorch

02. Introduction / Background

In 2015, Gatys, Ecker and Bethge showed that a convolutional network could separate the content of a photograph from the style of a painting and recombine them — "A Neural Algorithm of Artistic Style" (CVPR 2016) became one of the most reproduced results in deep learning. Quality is judged visually and no numeric scores are claimed: a frozen pre-trained VGG-19 supplies the representations: deep-layer activations describe content, while Gram matrices of feature maps describe style. Gradient descent then optimizes a target image until it carries the photo's layout in the painting's brushwork. This project turns that method into a complete PyTorch web application with a style-strength control and a fast feed-forward preview mode after Johnson et al. (2016).

03. Problem Statement

Neural style transfer is usually demonstrated as a one-off script with no interface and no controls. The project addresses this by building a full application: a Flask demo where a student can upload a content photo, choose or upload a style painting, dial the style strength, watch the optimization loss fall, and download the result — a complete, explainable deep-learning build suitable for review and viva.

04. Objectives

- Implement Gatys-style optimization transfer in PyTorch with a frozen pre-trained VGG-19 and Gram-matrix style representation.
- Build a Flask demo app with content upload, a style gallery, a style-strength slider, a live loss curve and downloadable results.
- Add a fast feed-forward preview mode after Johnson et al. for the built-in styles.
- Deliver the complete student kit: source code, weights, a styled-results gallery, report, PPT and viva Q&A document.

05. Proposed System / Working

The system has three stages. (1) Feature extraction: the content photo and style painting pass through a frozen pre-trained VGG-19; content features come from a deep layer, style features from Gram matrices of several layers. (2) Optimization: a target image initialized from the content photo is updated by gradient descent to minimize the weighted sum of content and style losses, with the style-strength slider setting the weighting; the loss curve is plotted live. (3) Preview and output: an optional feed-forward network gives near-instant previews for built-in styles, and the finished image is shown side by side with its sources and made downloadable.

06. Technologies / Components Used

- Python 3.10, PyTorch, torchvision with pre-trained VGG-19
- PIL, NumPy (image loading, transforms, saving)
- Matplotlib (live loss-curve display in the demo)
- Flask demo web app with upload, style gallery and slider UI
- Gram-matrix style representation after Gatys et al.
- Optional feed-forward transfer network after Johnson et al.

07. Key Features

- Content-photo upload with style selection from a built-in gallery
- Style-strength slider trading photo fidelity against style strength
- Custom style upload for any painting or texture
- Live content/style loss curve during optimization
- Fast feed-forward preview mode for built-in styles
- Side-by-side content, style and result comparison
- Downloadable full-resolution styled output

08. Applications / Use Cases

- Creative photo editing demos and digital-art experiments (prototype scope)
- Teaching material for CNN feature hierarchies, Gram matrices and loss design
- Reference build for other optimization-based generative student projects
- Starting point for video or real-time style-transfer experiments

09. Results & Scope

Style transfer is an artistic process with no ground truth, so this build makes no numeric claims at all. Quality is judged visually: the deliverable is a gallery of styled results plus the optimization loss curve from the run, which documents that the content/style tradeoff behaved as designed. Scope is single-image stylization of user photos; optimization takes minutes per image on CPU, the feed-forward preview

covers only built-in styles, and the result is an educational prototype rather than a commercial photo editor.

10. Conclusion

Neural Style Transfer App using PyTorch delivers a complete generative deep-learning project: a landmark method (Gatys et al.), a real PyTorch implementation with a frozen VGG-19, an interactive demo with meaningful controls, and a fast preview mode. The visible loss curve and the content-versus-style tradeoff give the student concrete, honest material to explain and defend in the report and viva.

11. References

- Gatys, L. A., Ecker, A. S. and Bethge, M. — A Neural Algorithm of Artistic Style (arXiv:1508.06576, CVPR 2016).
- Johnson, J., Alahi, A. and Fei-Fei, L. — Perceptual Losses for Real-Time Style Transfer and Super-Resolution (arXiv:1603.08155, ECCV 2016).
- Simonyan, K. and Zisserman, A. — Very Deep Convolutional Networks for Large-Scale Image Recognition (arXiv:1409.1556), the VGG-19 architecture.
- PyTorch and torchvision documentation (pytorch.org) — pre-trained VGG-19 weights and autograd tooling used in the build.