



A
Project Report
on
Neural Style Transfer
submitted for partial fulfillment for the award of
BACHELOR OF TECHNOLOGY
DEGREE
in
Computer Science
By
Palak Singh (2000290120103)
Ragini Rani (2000290120121)
Kalash Jain (2000290120080)

Under the Supervision of
Mr. Akash Goel
Assistant Professor
Department of Computer Science
KIET Group of Institutions, Ghaziabad
Affiliated to
Dr. A.P.J. Abdul Kalam Technical
University, Lucknow
May 2024

DECLARATION

We hereby declare that this submission is my work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material that to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Signature:

Name:	Palak Singh	Ragini Rani	Kalash Jain
Roll No. :	2000290120103	2000290120121	2000290120080

Date:



CERTIFICATE

This is to certify that the Project Report entitled “**Neural Style Transfer**” which is submitted by **Palak Singh, Ragini Rani, Kalash Jain** in partial fulfillment of the requirement for the award of degree B.Tech. in the Department of Computer Science of Dr. A.P.J. Abdul Kalam Technical University, Lucknow is a record of the candidate’s own work carried out by him under my supervision. The matter embodied in this report is original and has not been submitted for the award of any other degree.

Supervisor Signature

Mr. Akash Goel
(Assistant Professor)

Department of
Computer Science

Date:

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of the B.Tech Project undertaken during B.Tech. Final Year. We owe special debt of gratitude to **Assistant Professor Akash Goel**, Department of Computer Science, KIET, Ghaziabad, for his constant support and guidance throughout the course of our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only his cognizant efforts that our endeavors have seen light of the day.

We also take the opportunity to acknowledge the contribution of **Dr. Ajay Kumar Shrivastava, Head of the Department of Computer Science**, KIET, Ghaziabad, for his full support and assistance during the development of the project. We also do not like to miss the opportunity to acknowledge the contribution of all the faculty members of the department for their kind assistance and cooperation during the development of our project.

Signature:

Name:	Palak Singh	Ragini Rani	Kalash Jain
Roll No. :	2000290120103	2000290120121	2000290120080

Date :

TABLE OF CONTENTS

Content	Page No.
DECLARATION	
CERTIFICATE	
ACKNOWLEDGEMENTS	
ABSTRACT.....	i
LIST OF FIGURES.....	ii
LIST OF TABLES.....	iii
LIST OF ABBREVIATIONS.....	iv
 CHAPTER 1 INTRODUCTION.....	 1-7
1.1 Introduction To Project.....	1-3
1.2 Project Category.....	4-5
1.3 Objective.....	5
1.4 Scope.....	5-6
1.5 Structure of Report.....	6-7
 CHAPTER 2 LITERATURE REVIEW.....	 8-14
2.1 Literature Review.....	8-11
2.2 Research gap.....	12-13
2.3 Problem Formulation.....	14
 CHAPTER 3 PROPOSED SYSTEM.....	 15-16
3.1 Proposed System	15
3.2 Unique Features of The System (Difference from Existing System).....	15-16

CHAPTER 4. REQUIREMENT ANALYSIS AND SYSTEM SPECIFICATION.....	17-25
4.1 Feasibility Study (Technical, Economical, Operation).....	17
4.2 Software Requirement Specification.....	18-19
4.2.1 Data Requirement.....	18
4.2.2 Functional Requirement.....	18
4.2.3 Performance Requirement.....	19
4.2.4 Maintainability Requirement.....	19
4.2.5 Security Requirement.....	19
4.3 SDLC Model Used.....	19-21
4.4 Workflow Diagram.....	21
4.5 System Design Using Dfd Level 0.....	22
4.6 System Design Using Dfd Level 1.....	23
4.7 Er Diagrams.....	24
4.8 Use Case Diagram.....	25
CHAPTER 5. IMPLEMENTATION (METHODOLOGY).....	26-37
5.1 Methodology.....	26-32
5.2 Algorithm.....	32-33
5.3 Technology Used.....	34-36
5.2 Dataset Description.....	36-37
CHAPTER 6. TESTING, AND MAINTENANCE	38-40
6.1 Testing Techniques.....	38
6.2 Test Cases Used	38-40
CHAPTER 7. RESULTS AND DISCUSSIONS.....	41-47
7.1 User Interface Representation.....	41
7.2 Key Findings.....	41-42
7.3 Snapshots of Input Image.....	43-44

7.4	Snapshots of Output Image.....	45-46
7.5	Presentation of Results.....	46
7.6	Performance Evaluation.....	47
CHAPTER 8. CONCLUSION AND FUTURE WORK.....		48-52
8.1	Conclusion.....	48
8.2	Future Work.....	48-50
8.3	References.....	51-52
RESEARCH PAPER ACCEPTANCE PROOF.....		53
CERTIFICATES.....		54-55
RESEARCH PAPER.....		56-68
PROOF OF PATENT PUBLICATION.....		69

ABSTRACT

NST, or Neural Style Transfer has revolutionized the field of image processing by allowing the amalgamation of artistic styles to photographs. First introduced by Gatys et al., NST relies on a slow and iterative optimization process. However, recent advances have introduced faster and more efficient approaches, such as Adaptive Instance Normalization (AdaIN) and Johnson's method.

Gatys's method, which laid the foundation for NST, uses a CNN (Convolutional Neural Network) to extract information about the content of an image and artistic features or style of an image. This is based on minimizing the dissimilarity between the feature representations of content images and stylized images. This approach, although revolutionary, is very time consuming.

AdaIN introduced a revolutionary approach by reinterpreting version normalization to quickly combine content and style from arbitrary images. It eliminates the need for laborious optimization, allowing for real-time style transfer with great flexibility.

Johnson's method takes a different route by using perceptual loss and a pre-trained network for high-level feature comparison. This allows for precise training using network capabilities and real-time image transformation, giving the best of both worlds. This paper provides a comprehensive look at NST techniques and their evolution, shedding light on the future of image processing and style transfer.

LIST OF FIGURES

Figure No.	Description	Page No.
Fig 1.1	Representation of content and style	2
Fig 1.2	Comparison of different methods	4
Fig 2.1	Style Transfer	14
Fig 3.1	Process of style transfer	16
Fig 4.1	Stages of SDLC	20
Fig 4.2	Workflow diagram	21
Fig 4.3	System Design Using Dfd Leve	22
Fig 4.4	System Design Using Dfd Level 1	23
Fig 4.5	Er Diagrams	24
Fig 4.6	Use Case Diagram	25
Fig 5.1	Gayts flowchart	27
Fig 5.2	AdaIN flowchart	29
Fig 5.3	Johnson flowchart	32
Fig 5.4	VGG-19 Representation	34
Fig 7.1	Interfaces of our website	43
Fig 7.2	Content Image 1	43
Fig 7.3	Style Image 1	43
Fig 7.4	Content Image 2	44
Fig 7.5	Style Image 2	44
Fig 7.6	Content Image 3	44
Fig 7.7	Style Image 3	44
Fig 7.8	Output Image 1	45
Fig 7.9	Output Image 2	45
Fig 7.10	Output Image 3	46

LIST OF TABLES

Table. No.	Description	Page No.
Table 2.1	Literature Review	8-11
Table 2.2	Research gap	12-13
Table 6.1	Test case table	38-40
Table 7.1	Comparative analysis of NST	46

LIST OF ABBREVIATIONS

NST:	Neural Style Transfer
CNN:	Convolutional Neural Network
GAN:	Generative Adversarial Network
VGG:	Visual Geometry Group (name of a specific CNN architecture often used in NST)
Gram:	Gram Matrix (used in style loss calculation)
NLP:	Natural Language Processing
MRF:	Markov Random Field
CIN:	Conditional Instance Normalization
AdaIn:	Adaptive Instance Normalization

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Neural style transfer is a technique in deep learning that combines the content of one image with the style of another image. It leverages the power of convolutional neural networks (CNNs) to achieve this. The process involves taking two input images: a content image and a style image, and producing an output image that retains the structure of the content image while adopting the visual style of the style image.

Here's a basic overview of how neural style transfer works:

Pre-Trained CNN: Neural style transfer typically utilizes a pre-trained convolutional neural network (CNN), such as VGGNet or ResNet. These networks have learned to extract features from images effectively.

Content and Style Representation: The pre-trained CNN[3] is used to extract both content and style representations from the input images. The content representation captures the highlevel content features of the content image, while the style representation captures the textures, colors, and patterns that define the visual style of the style image.

Optimization Process: The goal is to generate an output image that matches the content of the content image and the style of the style image. This is achieved by defining a loss function that measures the difference between the content representation of the output image and the content representation of the content image, as well as the

difference between the style representation of the output image and the style representation of the style image.

Gradient Descent: The optimization process involves iteratively updating the pixels of the output image to minimize the content and style loss. This is typically done using gradient descent or some variant thereof.

Output Image: The final output image is the result of the optimization process. It retains the content structure of the content image while incorporating the visual style of the style image.

Neural style transfer[14] has applications in various fields, including art generation, photo editing, and visual effects. It allows for the creation of visually appealing images with artistic styles inspired by famous artists, paintings, or even arbitrary style images.

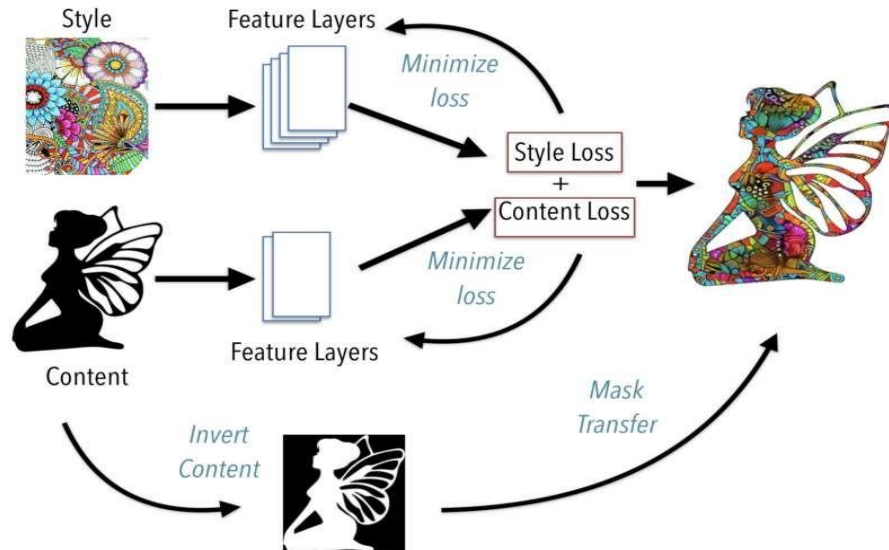


Figure 1.1 Representation of content and style loss

Neural Style Transfer, or NST, is like giving your photos a super cool makeover. It's like mixing different art styles with your pictures to make them look awesome and totally unique. It's like turning your

regular photos into something really special. NST is like magic for pictures!.

Neural style transfer in Gatys[1] et al. "A Neural Algorithm of Artistic Style." Their method involves defining a loss function that captures both the content and style of an image. The content loss measures the difference in content between the generated image and a content image, while the style loss measures the difference in style between the generated image and a style image. By minimizing these losses, the algorithm can generate an image that combines the content of one image with the style of another.

Johnson et al. proposed an improvement to neural style transfer in their paper "Perceptual Losses for Real-Time Style Transfer and Super-Resolution." They introduced the idea of using a pretrained convolutional neural network (CNN), such as VGG-19, to compute the content and style losses. This approach allows for faster training and inference compared to Gatys et al.'s method.

AdaIN is a technique introduced by Huang and Belongie in their paper "Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization." AdaIN[2] allows for arbitrary style transfer by adaptively normalizing the activations of one image to match the statistics (mean and standard deviation) of another image. This technique enables real-time style transfer and provides more flexibility in choosing style images. This approach is remarkably faster than the previous method by Gatys et al., with no loss of flexibility in transferring to new styles.

Overall, these approaches and techniques have significantly advanced the field of neural style transfer, making it possible to generate artistic images that combine the content of one image with the style of another in real-time or near-real-time.

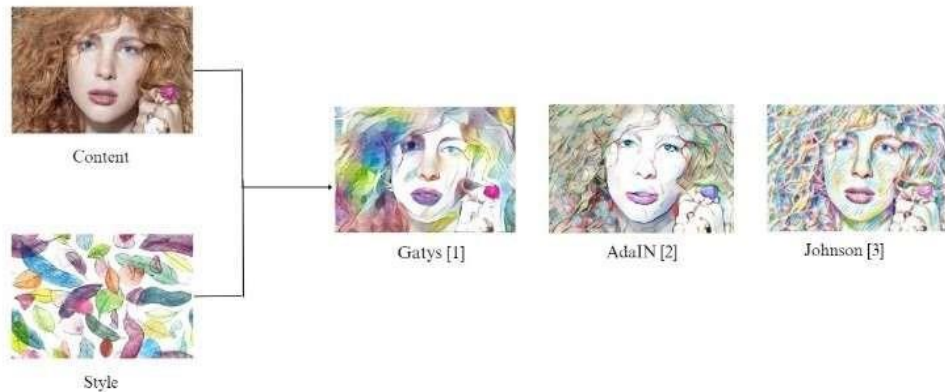


Figure 1.2 Comparison of different methods

1.2 PROJECT CATEGORY

Machine Learning Web Application Development with Neural Style Transfer (NST)

This category encapsulates the essence of our project, highlighting the integration of machine learning, specifically neural style transfer, into a web application. Here's a breakdown:

1. **Machine Learning:** Our project involves the utilization of machine learning techniques, particularly neural style transfer, to combine the content of one image with the style of another image. This encompasses the training and deployment of a machine learning model capable of transforming images according to specified artistic styles.
2. **Web Application Development:** By utilizing Streamlit, we're developing a web application that provides a user-friendly interface for users to interact with the neural style transfer model. This includes features such as image uploading, style selection, and real-time generation of stylized images. The web application aspect

emphasizes accessibility and ease of use for users accessing the functionality through a web browser.

- 3. Neural Style Transfer (NST):** Neural style transfer serves as the core algorithmic component of our project. It enables the transformation of images by combining the content of one image with the style of another, resulting in visually appealing outputs that reflect the artistic characteristics of the chosen style image.

In essence, our project involves the development of a web application that leverages machine learning, specifically neural style transfer, to provide users with an interactive platform for applying artistic styles to their images in real-time.

1.3 OBJECTIVE

- **Artistic Image Generation:** Use neural style transfer to create visually appealing images with artistic styles inspired by famous artworks, painters, or specific visual styles.
- **Content Preservation:** Ensure that the generated images retain the meaningful content and recognizable objects from the original input images.
- **Style Replication:** Replicate the distinctive artistic styles, including textures, colors, and patterns, from reference images to create visually appealing and stylistically consistent output.
- **Photo Editing and Enhancement:** Apply neural style transfer techniques to enhance photographs by transferring the visual style of a reference image onto the original photo.

1.4 SCOPE

By studying / discussing with the team members it was found that our project can be use for photo and video editing. However it can be added to the following projects:-

- Artistic Expression
- Virtual Reality and Gaming
- Advertising and Branding
- Interior design,
- Fashion

1.5 STRUCTURE OF REPORT

The project report has been divided into multiple chapters with each chapters getting divided into multiple sections and sub-sections.

Chapter 1 contains the introduction where a brief introduction of the project is stated. It also contains the category to which the project belongs to.

In **Chapter 2**, we have discussed the previous research work that has happened in this field as well as the problem statement. Apart from this, we have also mentioned the number of objectives that the system proposes to achieve.

Chapter 3 contains the detailed discussion of the project, its functionalities and the methodology used to create the proposed system. The various unique features of the project that differentiates this project with the existing ones, are also mentioned in the designated section.

In **chapter 4**, we have performed the feasibility study and mentioned the various parameters and evaluated the system's technical, economical and operational feasibility. Along with this, we have also included the software requirement specification document that contains data requirement, functional requirement, performance requirement, maintainability requirement as well as security requirement. This chapter also explains the Software Development Life Cycle (SDLC) model that has been used. It also illustrates the various diagrams like System Design using DFD Level 0 and Level 1, Use Case Diagram and ER Diagram.

Chapter 5 includes the implementation details of the proposed system like languages, tools and technologies that have been used to create the system whereas chapter 6 is dedicated to the testing and maintenance of the project.

Chapter 6 serves as the cornerstone for ensuring the longevity and reliability of the project, delving deep into the intricate processes of testing and maintenance.

In **Chapter 7** the results and discussion is done that contains the user interface representation, description of various modules of the project and snapshots of the system.

The final **Chapter 8** is dedicated to the conclusion and future scope of the system. The project report concludes with the references section and research paper and proof of patent publication.

CHAPTER 2

LITERATURE REVIEW

2.1 LITERATURE REVIEW

Table 2.1 This table represent the methodologies and key findings of research paper

Sr. No	Reference	Author(s)	Year	Research Focus	Methodology	Key Findings
1.	Wei and Levoy, 2000	L.-Y. Wei, M. Levoy	2000	Fast Texture Synthesis using Tree Structured VQ	Tree Structured Vector Quantization (VQ)	Proposed a fast texture synthesis algorithm using tree structured VQ
2.	Mahendran and Vedaldi, 2014	A. Mahendran, A. Vedaldi	2014	Understanding Deep Image Representations by Inverting	Convolutional Neural Networks (CNNs)	Explored the inversion of deep image representations for understanding
3.	Simonyan and Zisserman, 2014	Karen Simonyan, Andrew Zisserman	2014	Very Deep Convolutional Networks for Image Recognition	Deep Convolutional Networks	Introduced a deep architecture for large-scale image recognition

4.	Yang et al., 2014	C.-Y. Yang, C. Ma, M.-H. Yang	2014	Single-Image Super-Resolution: A Benchmark	Super-Resolution Algorithms	Established a benchmark for single-image super-resolution algorithms
5.	Liu et al., 2015	F. Liu, C. Shen, G. Lin	2015	Deep Convolutional Neural Fields for Depth Estimation	Deep Convolutional Neural Networks (CNNs)	Proposed deep CNNs for depth estimation from a single image
6.	Gatys et al., 2016	L. A. Gatys, A. S. Ecker, M. Bethge	2016	Image Style Transfer using CNNs	Convolutional Neural Networks (CNNs)	Developed an approach for artistic style transfer using CNNs
7.	Johnson et al., 2016	J. Johnson, A. Alahi, L. Fei-Fei	2016	Perceptual Losses for Real-Time Style Transfer	Convolutional Neural Networks (CNNs)	Introduced perceptual loss for real-time style transfer and super-resolution
8.	Dosovitskiy and Brox, 2016	Alexey Dosovitskiy, Thomas Brox	2016	Generating Images with Perceptual Similarity Metrics	Deep Networks, Perceptual Metrics	Proposed a method for image generation based on perceptual similarity metrics

9.	Li and Wand, 2016	C. Li, M. Wand	2016	Precomputed Real-time Texture Synthesis with M-GANs	Markovian Generative Adversarial Networks (M-GANs)	Introduced precomputed real-time texture synthesis using M-GANs
10.	Huang and Belongie, 2017	X. Huang, S. Belongie	2017	Arbitrary Style Transfer in Real-Time	Adaptive Instance Normalization (AdaIN)	Achieved real-time arbitrary style transfer with adaptive normalization
11.	Elad and Milanfar, 2017	Michael Elad, Peyman Milanfar	2017	Style Transfer via Texture Synthesis	Image Texture Synthesis	Explored style transfer through texture synthesis, achieving realistic results
12.	Hou et al., 2017	Q. Hou, M.-M. Cheng, X. Hu, A. Borji, Z. Tu, P. Torr	2017	Deeply Supervised Salient Object Detection with Short Connections	Deep Learning, Short Connections	Proposed a deeply supervised approach for salient object detection with short connections
13.	Jing et al.,	Y. Jing,	2020	Neural Style	Neural Style	Presented a comprehensive

	2020	Y. Yang, Z. Feng, J. Ye, Y. Yu, M. Song		Transfer: A Review	Transfer Techniques	review of neural style transfer methods
14.	Khattar et al., 2021	H. Khatteer, S. Arif, U. Singh, S. Mathur, S. Jain	2021	Product Recommendation System for E-Commerce using Collaborative Filtering	Collaborative Filtering, Textual Clustering	Developed a recommendation system integrating collaborative filtering and textual clustering for E-Commerce
15.	Sharma et al., 2022	S. Sharma, S. Gupta, D. Gupta, S. Juneja, G. Singal, G. Dhiman, S. Kautish	2022	Recognition of Gurmukhi Handwritten City Names Using Deep Learning	Deep Learning, Cloud Computing	Implemented a system for recognizing handwritten Gurmukhi city names using deep learning and cloud computing

2.2 RESEARCH GAPS:

Table 2.2 Limitations of research paper

References	Research Gaps
Gatys et al., 2016	Limited exploration of general image style transfer beyond artistic applications.
Huang and Belongie, 2017	Need for addressing depth estimation challenges, especially in complex scenes.
Johnson et al., 2016	Research gap in assessing the generalizability and limitations of image generation based on perceptual similarity metrics.
Wei and Levoy, 2000	Exploration of salient object detection with short connections in complex visual environments.
Elad and Milanfar, 2017	Limited exploration of collaborative filtering and textual clustering integration for recommendation systems in diverse E-Commerce scenarios.
Simonyan and Zisserman, 2014	Lack of in-depth analysis on the scalability and efficiency of very deep convolutional networks.
Dosovitskiy and Brox, 2016	Research gap in standardized metrics for evaluating and comparing the performance of neural style

	transfer techniques.
Mahendran and Vedaldi, 2014	Limited understanding of the interpretability and practical implications of inverted features.
Liu et al., 2015	Research gap in addressing depth estimation challenges, especially in complex scenes.
Li and Wand, 2016	Limited exploration of practical applications and challenges in inverting visual representations with CNNs.
Yang et al., 2014	Need for a more comprehensive benchmark that covers diverse scenarios in single-image super-resolution.
Jing et al., 2020	Lack of standardized metrics for evaluating and comparing the performance of neural style transfer techniques.
Hou et al., 2017	Limited exploration of salient object detection with short connections in complex visual environments.
Mahendran and Vedaldi, 2014	Limited understanding of the interpretability and practical implications of inverted features.

2.3 PROBLEM FORMULATION

The problem we wanted to solve in this project was how to improve the process of transferring the artistic style of one image to the content of another image using neural networks while maintaining the integrity of the content and achieve high quality results.

In other words, the central challenge to be addressed is how to make the neural type conversion process more efficient, productive and user-friendly, ensuring that it produces images that the tone is visually appealing while preserving the important elements of the original content.

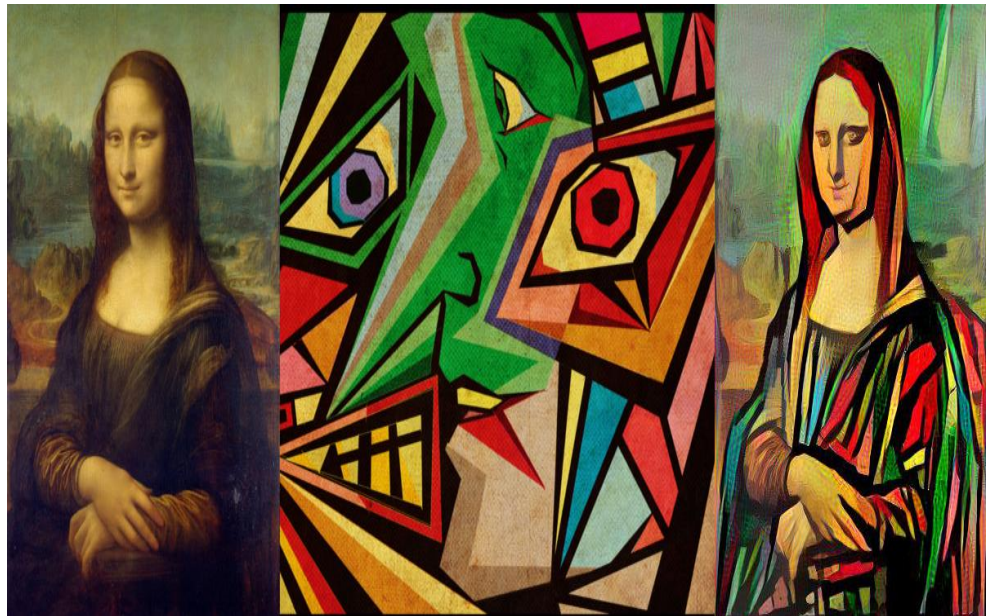


Figure 2.1 Style transfer

CHAPTER 3

PROPOSED SYSTEM

3.1 PROPOSED SYSTEM

Our proposed system is a web-based platform that leverages neural style transfer technology to allow users to create stylized images. Neural style transfer is a technique that involves applying the artistic style of one image (style image) to the content of another image (content image).

In our system, users will have the ability to upload two images: the content image, which contains the subject matter they want to stylize, and the style image, which defines the artistic style they wish to apply. The system will then use neural network algorithms to blend the content of the input images while preserving the style characteristics, resulting in a stylized output image.

3.2 UNIQUE FEATURES OF THE SYSTEM

Easy Input: Our system simplifies the stylization process by allowing users to easily upload their content and style images through a user-friendly interface.

Customization: Users have the flexibility to customize their stylized images by adjusting parameters such as style intensity and output resolution. This allows users to fine-tune their images to their preferences.

Fast Processing: The system utilizes cutting-edge neural network algorithms to ensure fast processing times, enabling users to view their stylized images within moments of uploading their input images.

Downloadable Results: Once the stylization process is complete, users can conveniently download their stylized images directly from the website. This makes it easy for users to share their creations or use them in various projects.

Inspiration Gallery: To inspire creativity and provide guidance, our platform includes an inspiration gallery featuring a curated collection of example style transfers. This allows users to explore various artistic styles and content combinations for inspiration.

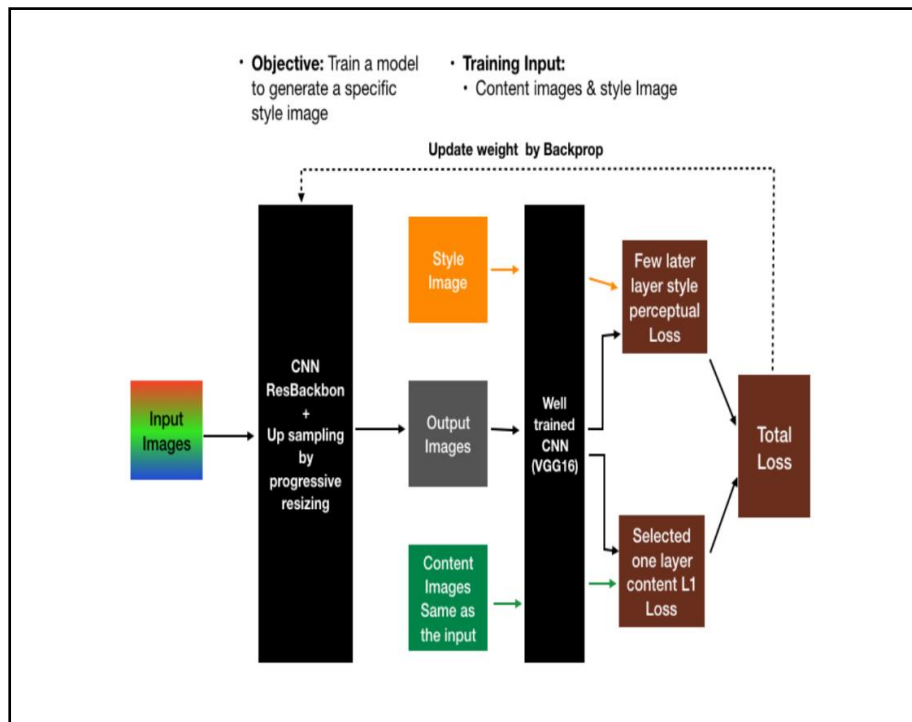


Figure 3.1 Process of style transfer

CHAPTER 4

REQUIREMENT ANALYSIS AND SYSTEM SPECIFICATION

4.1 FEASIBILITY STUDY:-

4.1.1 Technical Feasibility

Available Technology: Assess the availability of appropriate neural network architectures, deep learning frameworks (e.g., TensorFlow, PyTorch), and image processing libraries.

Hardware Requirements: Determine the computational resources (e.g., CPU, GPU) required for training and inference, and assess their availability within the organization.

Integration Challenges: Identify potential challenges in integrating the system with existing software systems, databases, or APIs.

4.1.2 Economical Feasibility

Development Costs: Estimate the cost of acquiring software licenses, development tools, and hardware infrastructure for system development.

Operational Costs: Assess ongoing expenses such as hosting, maintenance, and support services.

4.1.3 Operational Feasibility

User Acceptance: Assess the willingness of end-users to adopt the system and their satisfaction with its functionality and performance.

Organizational Impact: Evaluate the impact of implementing the system on existing business processes, workflows, and organizational culture.

Training and Support: Determine the training and support requirements for end-user and IT staff to ensure effective system adoption and maintenance.

4.2 SOFTWARE REQUIREMENT SPECIFICATION DOCUMENT

4.2.1 Data Requirement:

- **Input Images:** The system should accept two input images: a content image and a style image.
- **Output Image:** The system should generate an output image that combines the content of the content image with the style of the style image.

4.2.2 Functional Requirement:

- **Feature Extraction:** Extract features from the content image and the style image using a pre-trained convolutional neural network (CNN).
- **Loss Calculation:** Calculate the content loss to ensure that the generated image preserves the content of the content image.
- **Optimization:** Use an optimization algorithm (e.g., gradient descent) to minimize the total loss by adjusting the pixel values of the generated image.
- **Output Generation:** Generate the final output image that combines the content and style based on the optimized pixel values.

4.2.3 Performance Requirement:

- **Response Time:** The system should generate the output image within a reasonable time frame, considering the complexity of the neural style transfer process.
- **Resource Utilization:** Ensure efficient utilization of computational resources (CPU/GPU) during the style transfer process.

4.2.4 Maintainability Requirement:

- **Modularity:** The system should be modular, allowing for easy updates and modifications to individual components such as feature extraction, loss calculation, and optimization.
- **Documentation:** Provide comprehensive documentation for the system architecture, algorithms used, and codebase to facilitate future maintenance and enhancements.

4.2.5 Security Requirement:

- **Data Privacy:** Ensure that input images are processed securely and that the generated output image does not leak sensitive information.
- **Access Control:** Implement access control mechanisms to restrict unauthorized access to the system and its components.

4.3 SDLC MODEL TO BE USED

Software development life cycle (SDLC) is a structured process that is used to design, develop, and test good-quality software. SDLC, or software development life cycle, is a methodology that defines the entire procedure of software development step-by-step.

A software development life cycle (SDLC) consists of 6 stages: Planning, Analysis, Design, Implementation, Testing and Maintenance.

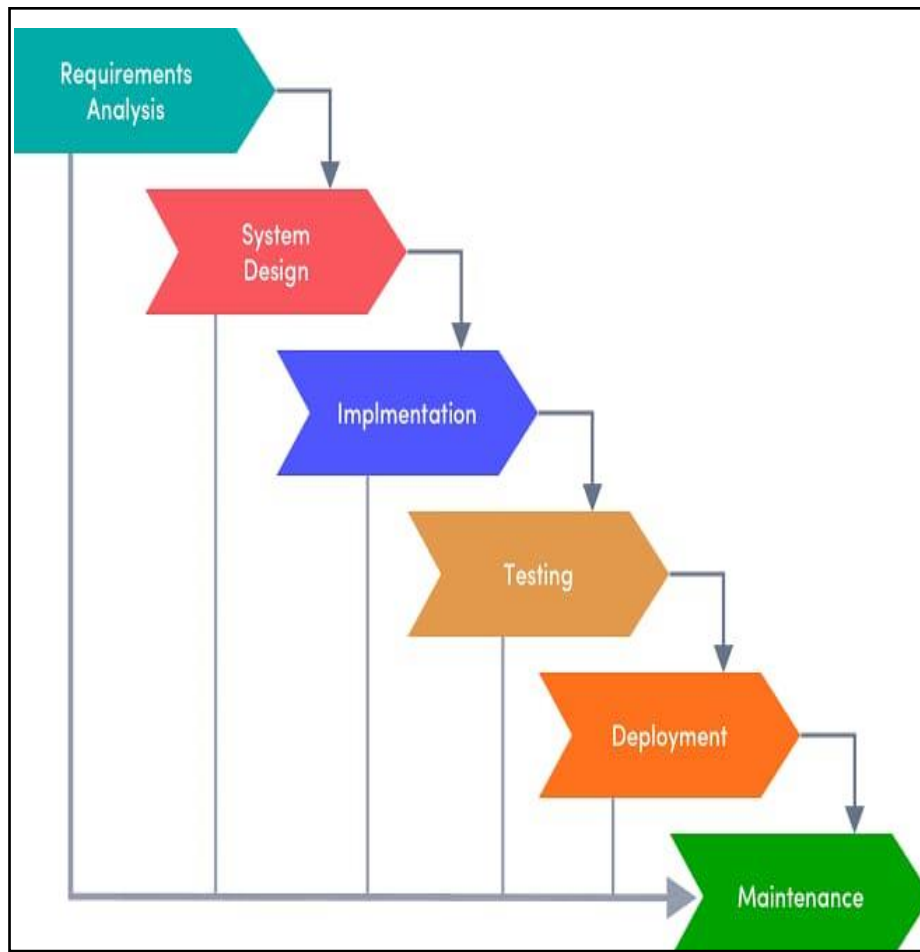


Fig. 4.1. Stages of Software Development Life Cycle (SDLC)

For the neural style transfer project, the Agile Model would be most suitable. This model allows for iterative development and refinement of the system in response to feedback and changing requirements. Since neural style transfer involves experimentation with various hyperparameters and visual inspection of results, an iterative approach enables rapid prototyping, testing, and refinement of the system until satisfactory performance is achieved. Additionally, the Agile Model promotes collaboration between developers and stakeholders, ensuring that the final system meets the desired objectives effectively.

4.4 WORK FLOW DIAGRAM

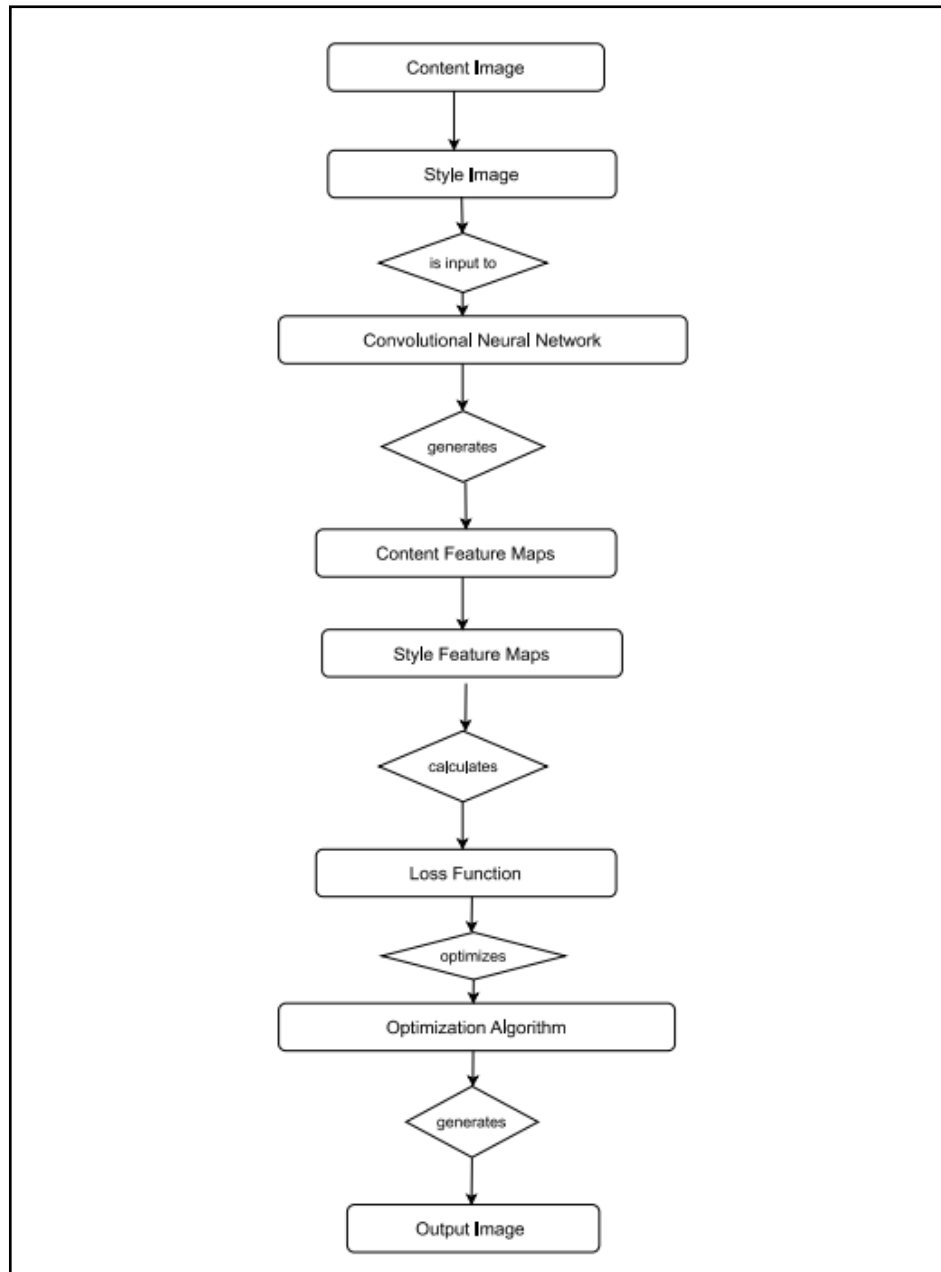


Fig. 4.2 Workflow Diagram

4.5 SYSTEM DESIGN USING DFD LEVEL 0

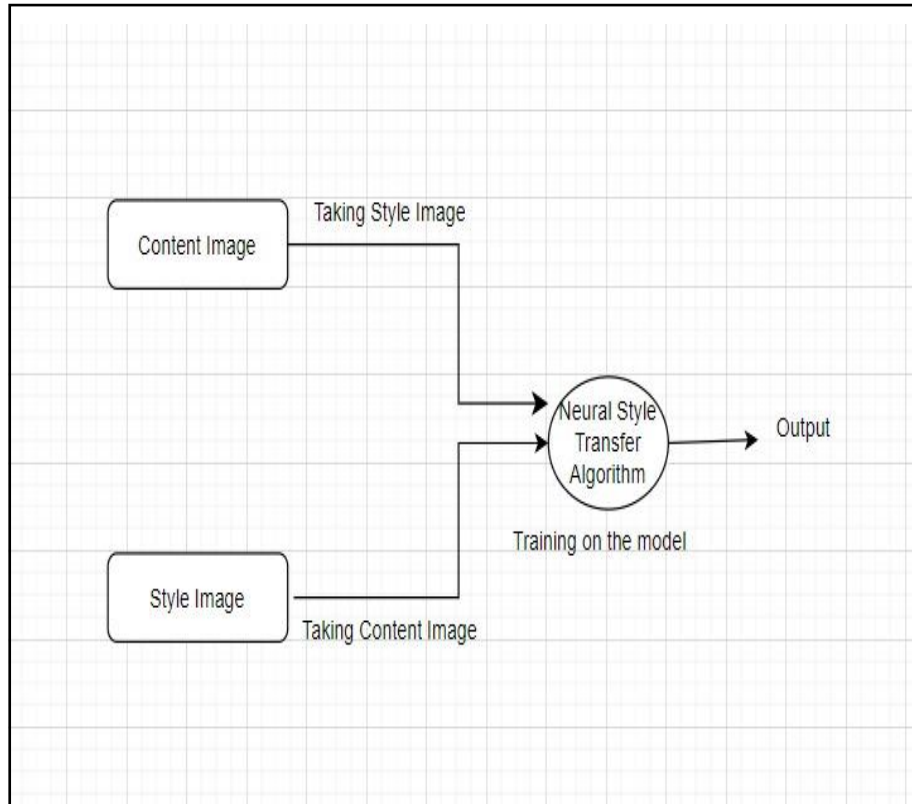


Fig. 4.3 DFD Level 0

The above figure shows the 0 Level Data Flow Diagram (DFD) of Neural Style Transfer.

This is the highest-level DFD, which provides an overview of the entire system. It shows the major processes, data flows, and data stores in the system, without providing any details about the internal workings of these processes.

The major process is NST Process and the external entities are Content image and Style image. Data flow is shown with arrows that describe the route the data takes between the external entities, processes and data stores.

4.6 SYSTEM DESIGN USING DFD LEVEL 1

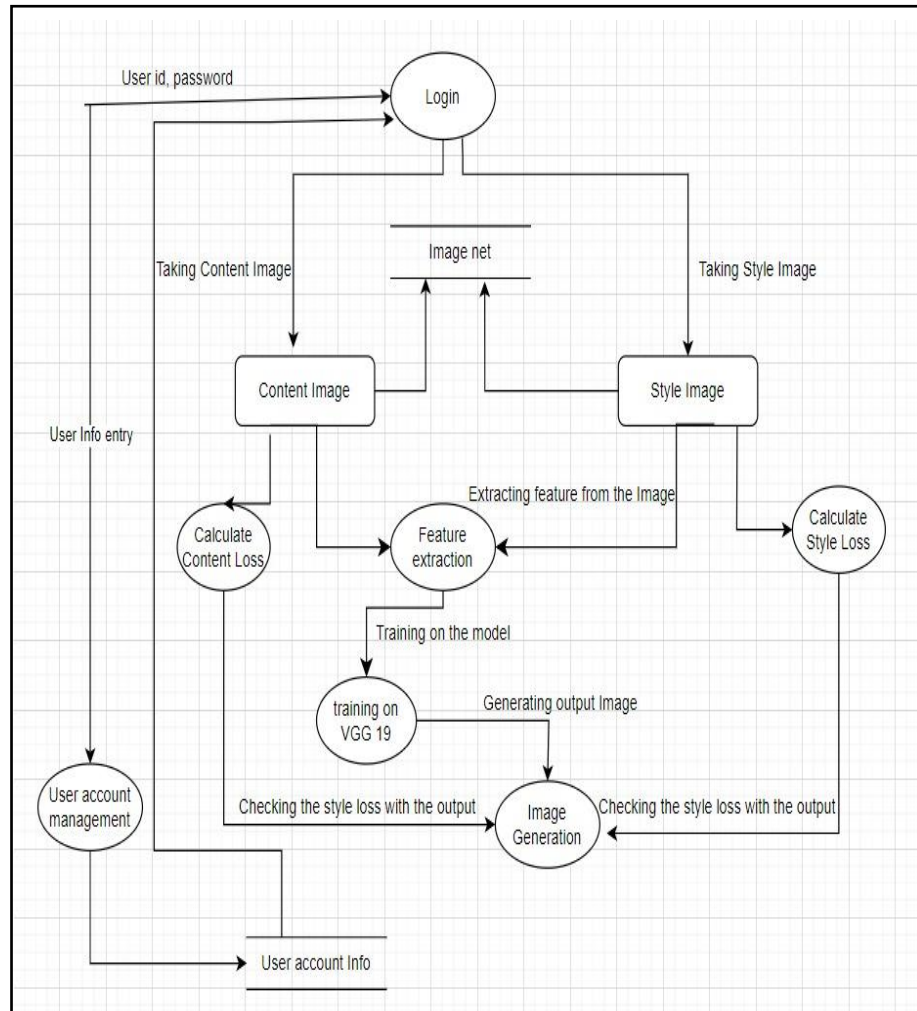


Fig 4.4 DFD Level 1

The above figure shows the 1 Level Data Flow Diagram (DFD) of Neural Style Transfer.

This is the highest-level DFD provides a more detailed view of the system by breaking down the major processes identified in the DFD level 0 into sub-processes.

4.7 ER DIAGRAM

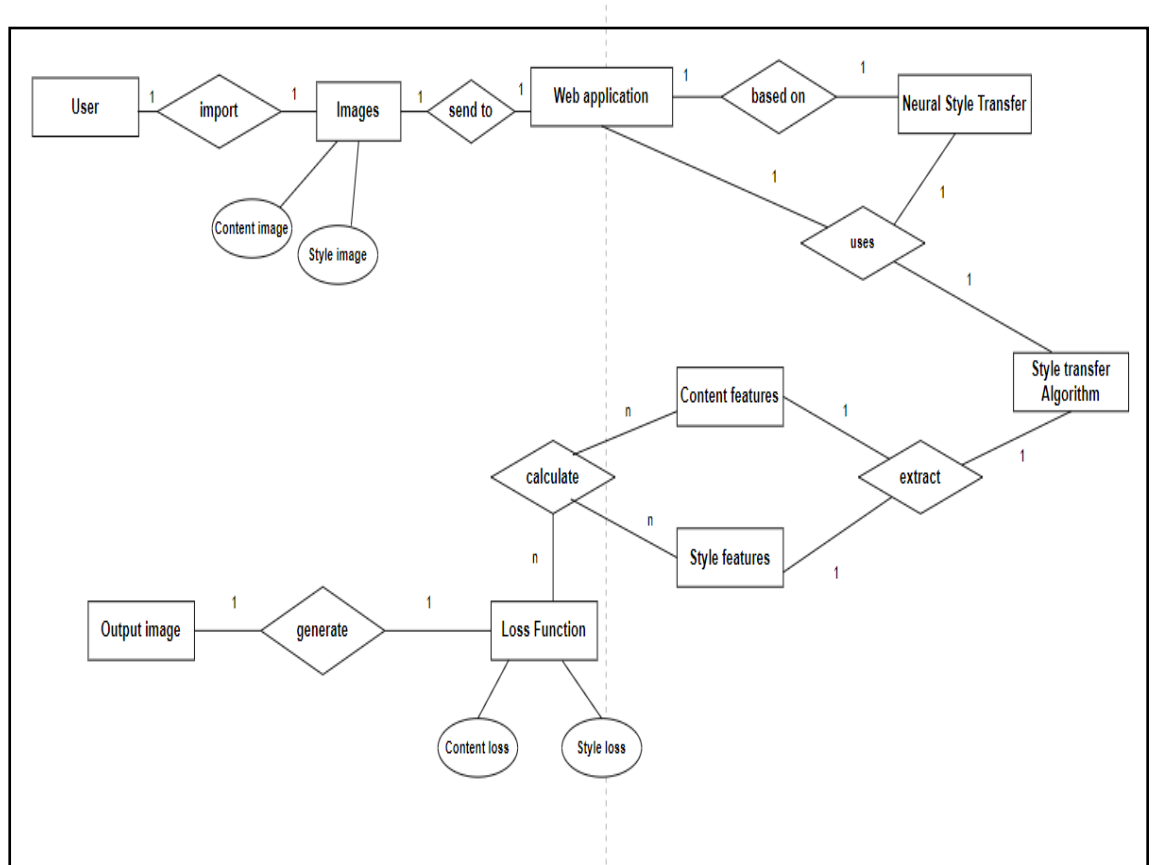


Fig 4.5 ER Diagram

In this ER Diagram of neural style transfer (NST) process facilitated by a web application, users begin by importing two images: a content image and a style image. These images are then processed through the web application, which employs a style transfer algorithm. This algorithm extracts features from both the content and style images. Subsequently, the algorithm calculates a loss function, which includes both style loss and content loss. Based on this calculated loss function, the web application generates an output image. This output image seamlessly combines the content of the content image with the stylistic elements of the style image, resulting in a visually appealing synthesis of the two.

4.8 USE CASE DIAGRAM

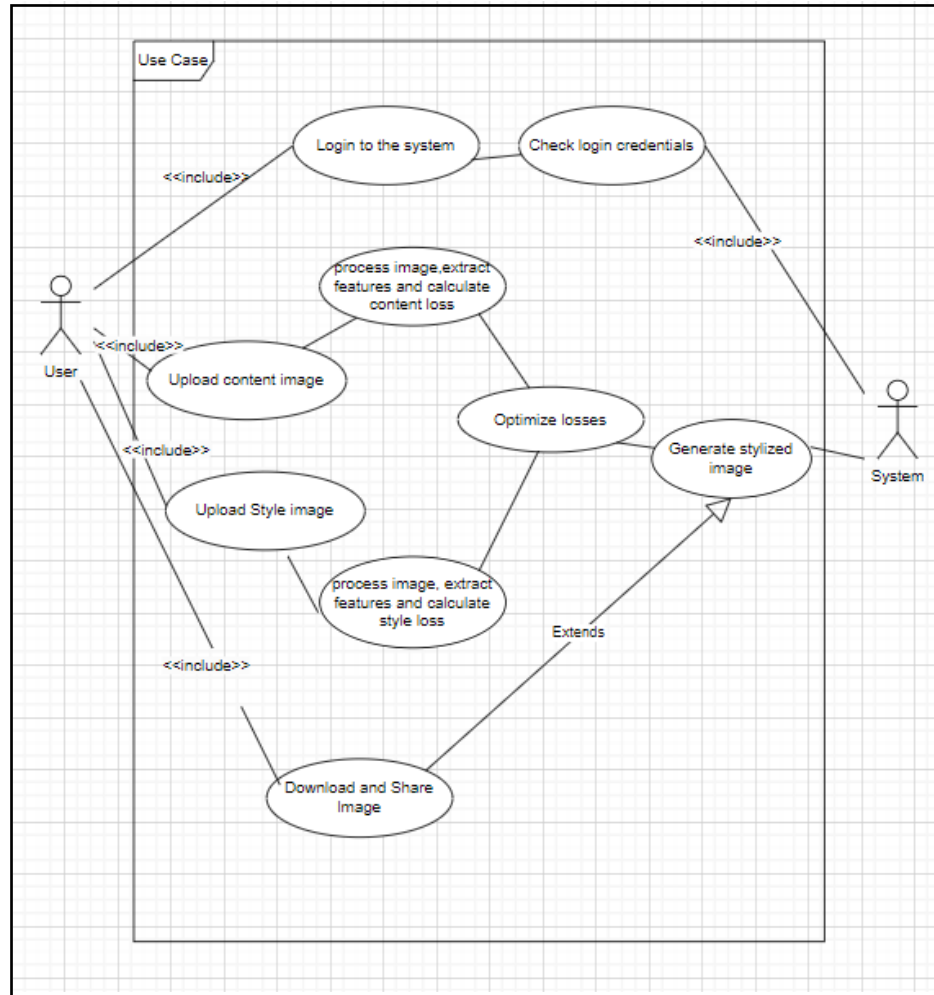


Fig 4.6 Use case Diagram

In this use case, users upload both a content and style image and log in to the system. The images undergo processing, feature extraction, and calculation of content and style loss. Following this, an optimization process occurs. Finally, the system generates the stylized image based on the optimized parameters.

CHAPTER – 5

IMPLEMENTATION

5.1 METHODOLOGY

- The Gatys Method [1] utilizes VGG-Network with 16 convolutional and 5 pooling layers, employing average pooling for gradient flow. It assesses feature maps at each layer, calculating content loss via squared error, and style loss through statistical property differences in mean and variance.
- The AdaIN Method [2] employs an encoder-decoder structure, utilizing AdaIN layers to align mean and variance of content and style feature maps. It employs pretrained VGG-19 for content and style loss calculation, emphasizing mean and variance disparity.
- Johnson's Method [3] involves an image transformation network (f_v) and a loss network (ϕ) using pre-trained VGG with 16 layers. It optimizes via gradient descent, employing perceptual loss functions for content and style reconstruction, emphasizing feature maps and Gram matrices.

GAYTS METHOD:-

The total loss function (L_{total}) is defined as the sum of the content loss ($L_{content}$) and the style loss (L_{style})[1]. The coefficients α and β determine the balance between content and style. The goal is to minimize this combined loss.

$$L_{total} = \alpha * L_{content} + \beta * L_{style}$$

Content Loss:

The content loss ($L_{content}$) measures the difference between the feature maps of the original image and the generated image. It ensures that the content of the generated image resembles the content of the original image.

Style Loss: The style loss (L_{style}) is the sum of style losses across multiple layers, weighted by coefficients. Style loss is calculated based on the differences in Gram matrices between the style image and the generated image.

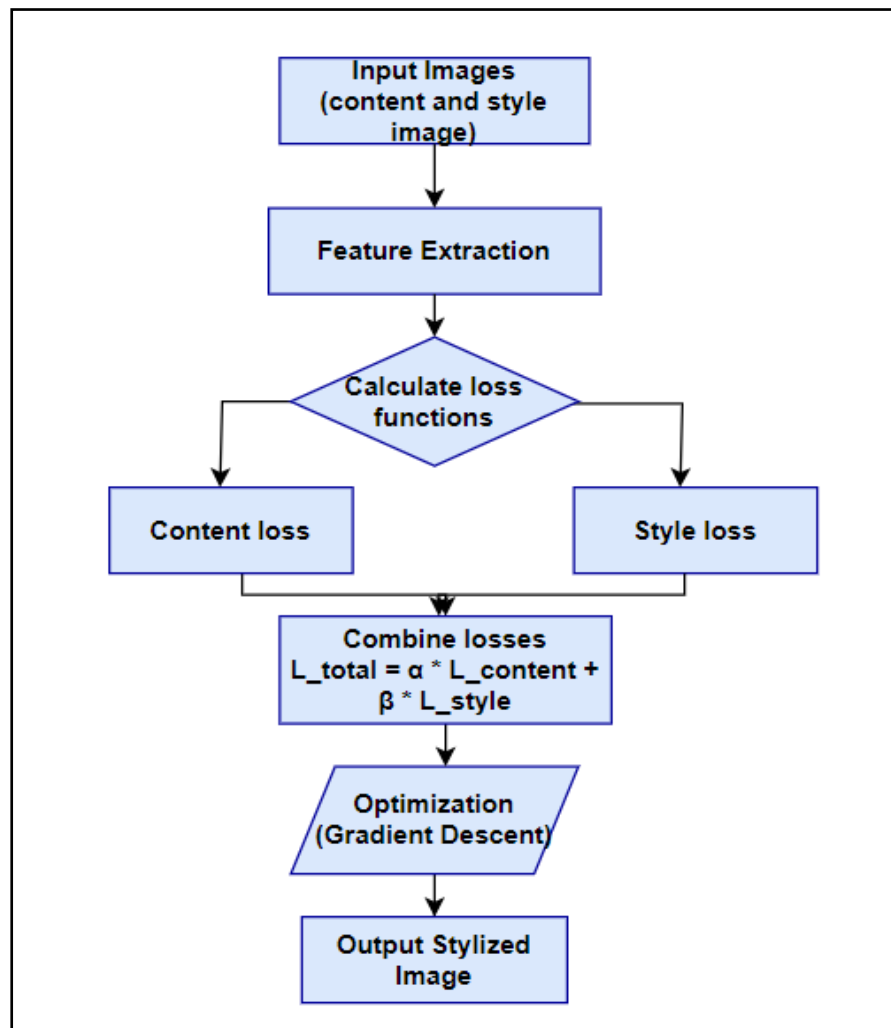


Fig 5.1 Flowchart of Gayts Method

ADAIN METHOD:-

The AdaIN[2] method facilitates neural style transfer by combining content and style from two input images to create a stylized output. It utilizes an encoder-decoder structure, where the encoder captures essential features, and the decoder generates the final stylized image.

ENCODER:

The encoder is a segment of the pre-trained VGG19 model, extracting features from both content and style images. The AdaIN layer aligns the mean and standard deviation of content and style feature maps without introducing additional parameters.

AdaIN Layer Formula:

$$\text{AdaIn}(p,q)=\sigma(q)((p-\mu(p))/\sigma(p))+\mu(q)$$

This formula adjusts the mean and standard deviation of the content feature map to match those of the style feature map

DECODER:

The decoder mirrors the encoder's architecture and is configured symmetrically. It avoids normalization layers within the decoder to maintain optimal performance.

LOSS FUNCTIONS:

The loss functions for neural style transfer involve content loss (L_c) and style loss (L_s), combined to form the total loss (L_t) with a weighting parameter (λ).

$$L_t = L_c + \lambda L_s$$

OPTIMIZATION:

During optimization, the goal is to find an image that minimizes the total loss (L_t), thereby achieving a balance between content fidelity and style transfer. This iterative process involves updating the pixel values of the generated image to progressively match the content features of the content image and the style features of the style image.

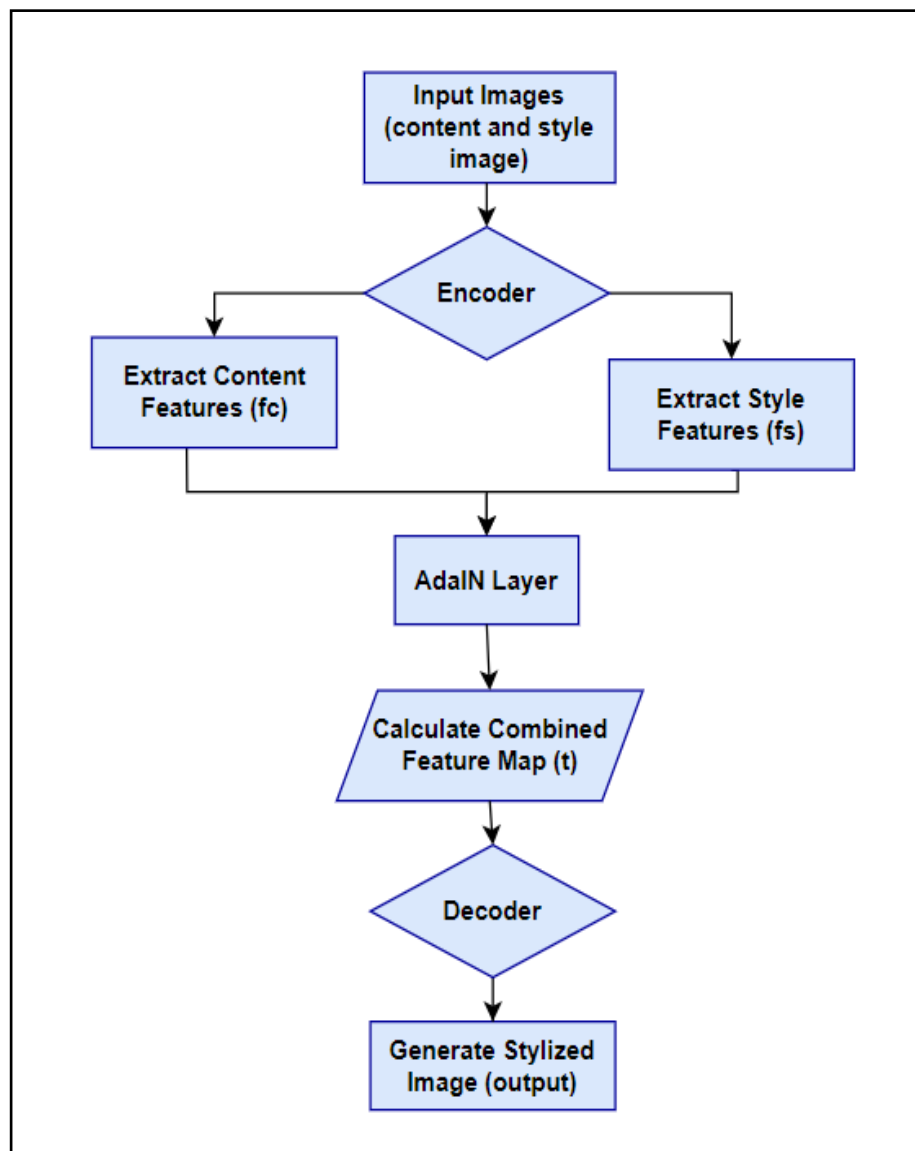


Fig 5.2 Flowchart of AdaIN Method

JOHNSON METHOD:-

The Johnson method [3] for neural style transfer is a powerful and versatile approach that combines an image transformation network with perceptual loss functions to achieve highquality stylized outputs. Here's a more detailed explanation:

Image Transformation Network (f_v):

The image transformation network, denoted as f_v , is responsible for generating the stylized output image. It takes the content image as input and applies transformations to it to match the style of the reference image.

This network typically consists of convolutional layers, which learn to extract and manipulate features from the input image to produce the desired stylized result.

Through the optimization process, the parameters of this network are adjusted to minimize the perceptual loss functions, effectively guiding the generation of the stylized image.

Pre-trained Classification Network (ϕ):

The pre-trained classification network, denoted as ϕ , serves as a feature extractor. It is typically a deep convolutional neural network trained on a large dataset for image classification tasks, such as VGG, ResNet, or Inception.

This network is used to compute feature representations of the content and style images, which are then used in the computation of the perceptual loss functions.

Perceptual Loss Functions:

The Johnson method employs three distinct perceptual loss functions to guide the optimization process:

a)- Feature Reconstruction Loss:

This loss function compares the activations of specific layers in the image transformation network (fv) with those of the content image. By evaluating high-level differences in feature representations, it captures semantic details and ensures that the content of the stylized image remains faithful to the original content image.

b)- Style Reconstruction Loss:

The style reconstruction loss measures the difference in stylistic features between the generated stylized image and the target style image. It does this by comparing the Gram matrices of feature maps extracted from intermediate layers of the image transformation network and the style image. This helps in preserving the artistic style and texture of the style image in the generated output.

c)- Pixel Loss:

This loss function quantifies the dissimilarity in pixel values between the generated stylized image and the target image. By promoting accurate content preservation at the pixel level, it ensures that the generated image maintains visual fidelity to both the content and style images.

Optimization:

The optimization process involves minimizing the combined perceptual loss, which is the sum of the feature reconstruction loss,

style reconstruction loss, and pixel loss, weighted by appropriate coefficients.

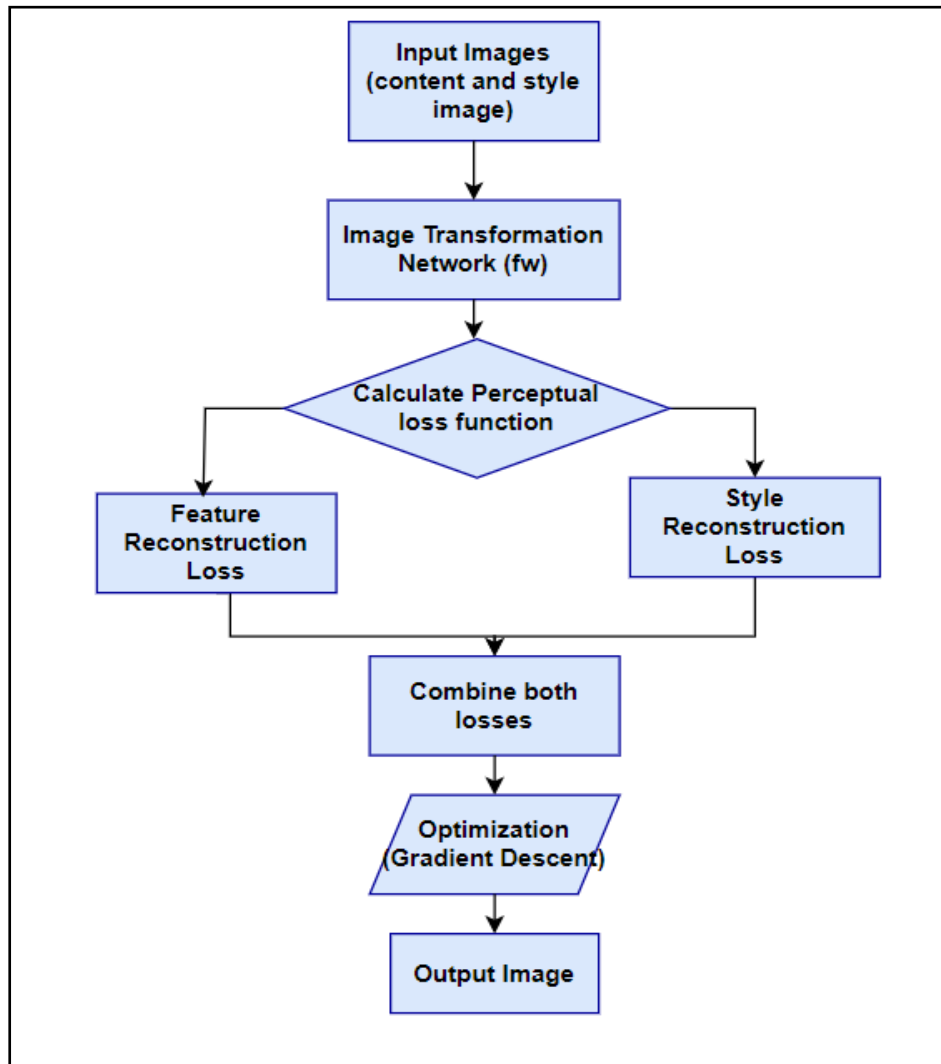


Fig. 5.3 Flowchart of Johnson method

5.2 ALGORITHM

Convolutional neural network (CNN) :

Convolutional Neural Networks (CNNs) are used in neural style transfer because of their ability to capture hierarchical features in

images. CNNs have proven to be very effective in computer vision tasks due to their ability to automatically learn and extract features from images. In neural style transfer, CNNs are used to extract both content and style features from the input images[4].

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning method that can take in an input image, give various elements and objects in the image importance (learnable weights and biases), and be able to distinguish between them. Comparatively speaking, a ConvNet requires substantially less pre-processing than other classification techniques. ConvNets have the capacity to learn these filters and properties, whereas in primitive techniques filters are hand-engineered. A ConvNet's architecture was influenced by how the Visual Cortex is organised and is similar to the connectivity network of neurons in the human brain. Only in a small portion of the visual field known as the Receptive Field do individual neurons react to inputs.

In other words, the network can be trained to understand the sophistication of the image better.

VGG19(Visual Geometric Group)

One of the top computer vision models to date is the CNN (Convolutional Neural Network) variant known as VGG19. VGG19[5] is an object identification and classification method that has a 92.7% accuracy rate when classifying 1000 photos into 1000 different categories. It is a well-liked technique for classifying images and is simple to employ with transfer learning.

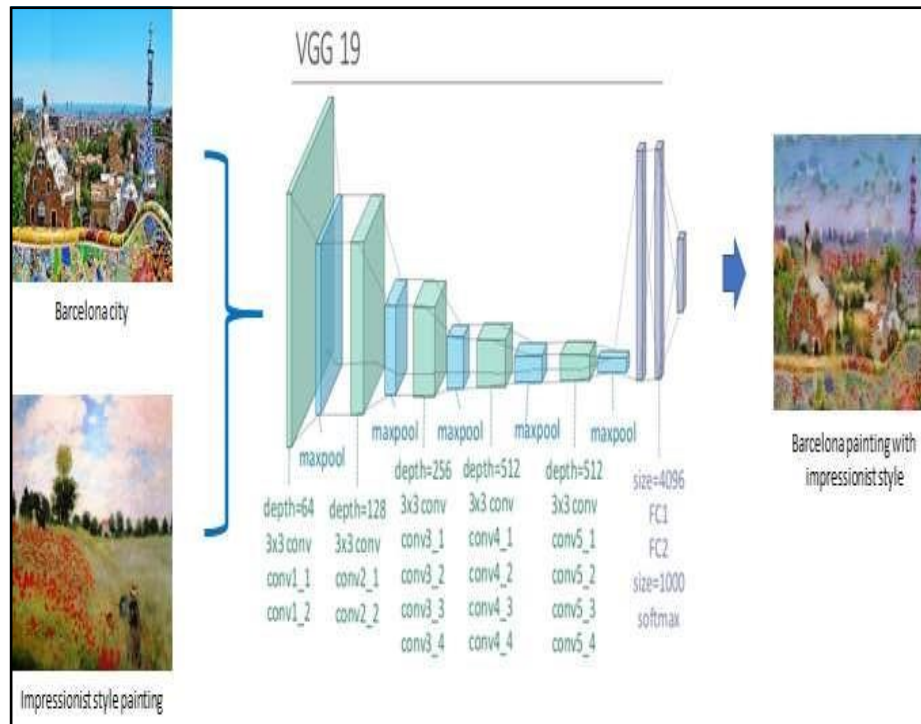


Fig. 5.4 VGG19 Representation

5.3 TECHNOLOGY USED

Deep Learning Frameworks:

- **TensorFlow:** Developed by Google, TensorFlow [9] is a popular open-source deep learning framework known for its flexibility and scalability. It provides high-level APIs for building neural networks and includes pre-trained models for tasks like image classification and style transfer.
- **PyTorch:** Developed by Facebook, PyTorch is another widely-used open-source deep learning framework known for its dynamic computational graph and intuitive interface. It is favored by researchers [10] and provides extensive support for neural style transfer implementations.

Convolutional Neural Networks (CNNs):

CNN architectures such as VGGNet, ResNet, and MobileNet are commonly used in neural style transfer projects. These networks are pre-trained on large image datasets like ImageNet and provide powerful feature extraction capabilities[4].

GPU Acceleration:

Graphics Processing Units (GPUs) are essential for accelerating the training and inference process in deep learning projects, including neural style transfer[1]. Technologies like CUDA (Compute Unified Device Architecture) from NVIDIA enable parallel computation on GPUs, significantly speeding up neural network training.

Python Programming Language:

Python is the most widely-used programming language in the field of machine learning and deep learning. It offers extensive libraries and frameworks for data manipulation, numerical computation, and deep learning model development.

Image Processing Libraries:

OpenCV[11] (Open Source Computer Vision Library) and PIL (Python Imaging Library) are commonly used for image loading, preprocessing, and manipulation tasks in neural style transfer projects.

Web Development Technologies :

Streamlit is utilized to build a user-friendly web interface for our style transfer application, allowing users to interactively upload images and apply artistic styles. Additionally, HTML, CSS, JavaScript, and web frameworks like Flask may be incorporated for web application development.

Containerization and Deployment:

Docker and Kubernetes are popular technologies for containerization and deployment of deep learning applications. They provide a consistent environment for running neural style transfer models across different platforms and environments.

Version Control and Collaboration:

Version control systems like Git and collaboration platforms like GitHub are essential for managing project code, tracking changes, and facilitating collaboration among team members.

5.4 DATASET DESCRIPTION

The ImageNet dataset is one of the most widely used large-scale datasets in the field of computer vision. It contains millions of labeled images across thousands of categories.

ImageNet Dataset:

- The ImageNet dataset consists of over 14 million images, covering more than 20,000 categories.
- Each image in the dataset is labeled with one or more categories, allowing for supervised learning tasks.
- The dataset includes a wide variety of images, encompassing diverse scenes, objects, and visual styles.
- ImageNet is perhaps best known for the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), where researchers compete to develop algorithms for object detection, image classification, and other computer vision tasks.

Utility in Neural Style Transfer:

- **Pre-Trained Models:** Many state-of-the-art neural style transfer models, including those based on VGGNet or ResNet architectures, are often pre-trained on the ImageNet dataset. These pre-trained models have learned to extract rich features from images, making them well-suited for tasks like style transfer [7].
- **Feature Extraction:** In neural style transfer, the pre-trained CNN is used to extract both content and style representations from the input images. The features learned by the CNN from the ImageNet dataset capture a wide range of visual concepts, which can be leveraged to represent both content and style in a meaningful way [8].
- **Transfer Learning:** Transfer learning, a technique where a model trained on one task is adapted for use on another task, is commonly applied in neural style transfer. By utilizing a pre-trained CNN trained on ImageNet, we can transfer the learned features to the style transfer task, allowing for faster convergence and better performance, especially when dealing with limited training data.
- **Generalization:** Since the ImageNet dataset covers a broad range of visual concepts and styles, models pre-trained on ImageNet tend to generalize well to various image manipulation tasks, including style transfer. This enables the neural style transfer algorithm to capture and transfer diverse artistic styles effectively.

CHAPTER 6

TESTING AND MAINTENANCE

6.1 TESTING TECHNIQUES

- The testing to be performed is black box testing.
- The testing is performed by the developers team along with QA and Configuration Manager.
- In black box testing, the internal workings of the system are not known to the tester, who evaluates the system's functionality based on its inputs and outputs. This technique ensures that the neural style transfer application behaves as expected from the end-user's perspective, regardless of its internal implementation details. By involving multiple stakeholders in the testing process, we ensure comprehensive coverage of test cases and effective validation of the application's functionality, user interface, and performance.

6.2 TEST CASES USED

Test cases:-

Table 6.1 Test cases for style transfer

Test Case	Test Objective	Test Data	Expected Result	Actual Result	Pass/Fail
1	User Login	User Id and Password	Only Valid User login in the system	Unauthorized user can not login	Pass

2	Model Training	Training data with content and style images	Model converges, successfully learns style transfer	Model successfully learns style transfer	Pass
3	Inference	Test images (256x256 pixels) with varying content and styles	Generated images (256x256 pixels) exhibit desired style while retaining content	Generated images (256x256 pixels) exhibit desired style while retaining content	Pass
4	Content Preservation	Generated images	Content remains recognizable	Content recognizable	Pass
5	Style Transfer Quality	Generated images	Style closely matches style image	Style of the output closely matches style image	Pass
6	Validate content layer	Sample content images, pre-trained model weights	Content layer produces expected feature maps	Content layer produces expected feature maps	Pass

7	Validate style layer	Sample style images, pre-trained model weights	Style layer produces expected Gram matrices	Style layer produces expected Gram matrices	Pass
8	Verify style transfer accuracy	Input images (256x256 pixels) with known content and style	Output images (256x256 pixels) match the desired style and content	Output images (256x256 pixels) match the desired style and content	Pass
9	Verify data security	Testing with simulated security breaches	No unauthorized access or data breaches detected	Unauthorized user can not access	Pass
10	Assess the impact of updates	System before and after updates	Previously tested features still work as expected	Previously tested features work accurately	Pass
11	Image Size Verification	Test images (256x256 pixels) of various sizes	Generated images (256x256 pixels) maintain aspect ratio and content using a 3x3 kernel	Aspect ratio preserved, content recognizable using a 3x3 kernel	Pass

CHAPTER 7

RESULTS AND DISCUSSIONS

The key finding of the Neural style is that the representation of content and style of the Convolutional Neural Network are well separable so that we generate the images which is the mixture of the content and style representation from two different source image.

It Results on a variety of photographs demonstrate that the suggested method may transfer the style while maintaining the structure and color of the content image, reducing artifacts.

7.1 USER INTERFACE REPRESENTATION

Our neural style transfer website offers a user-friendly platform for creating stunning stylized images by blending the content of one image with the artistic style of another. With just a few clicks, users can transform their ordinary photos into captivating artworks inspired by famous paintings, iconic styles, or personal preferences.

7.2 KEY FINDINGS:

Easy Input: Users can upload two images - a content image containing the subject matter they want to stylize and a style image defining the artistic style they wish to apply.

Customization: Our platform provides options to adjust parameters such as style intensity, output resolution, and more, allowing users to fine-tune their stylized images to perfection.

Fast Processing: Utilizing cutting-edge neural network algorithms, our website delivers swift processing times, ensuring users can view their stylized images in moments.

Downloadable Results: Once the stylization process is complete, users can conveniently download their stylized images directly from the website, ready to be shared or used in various projects.

Inspiration Gallery: To spark creativity and provide guidance, we have curated a collection of example style transfers showcasing a diverse range of artistic styles and content combinations.

How It Works:

Upload: Users select their desired content and style images and upload them to the website. **Stylize:** Our neural network works its magic, seamlessly blending the content of the input images while preserving the style characteristics.

Preview: Users can preview the stylized image and make adjustments if necessary.

Download: With just a click, users can download their stylized masterpiece and use it however they please.

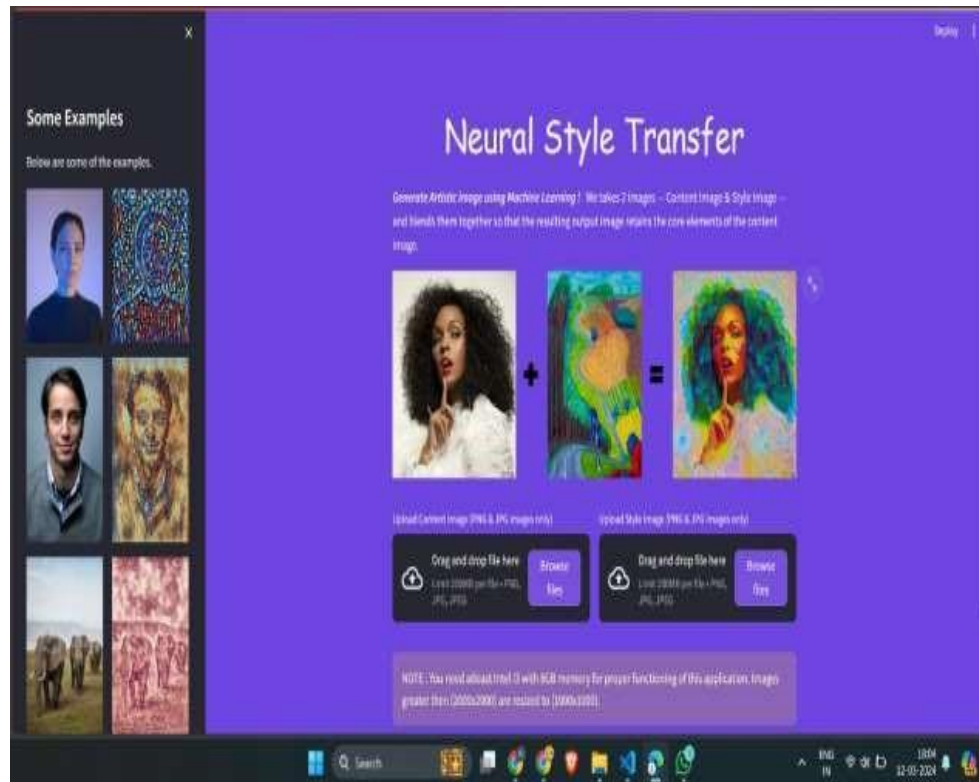


Fig 7.1 Interface of our website

7.3 SNAPSHOTS OF INPUT IMAGE



Fig 7.2 Content Image 1



Fig 7.3 Style Image 1



Fig 7.4 Content Image 2

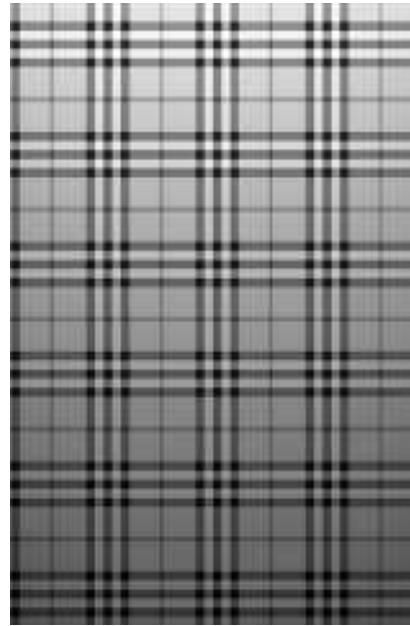


Fig 7.5 Style Image 2



Fig 7.6 Content Image 3



Fig 7.7 Style Image 3

7.4 SNAPSHOTS OF OUTPUT IMAGE

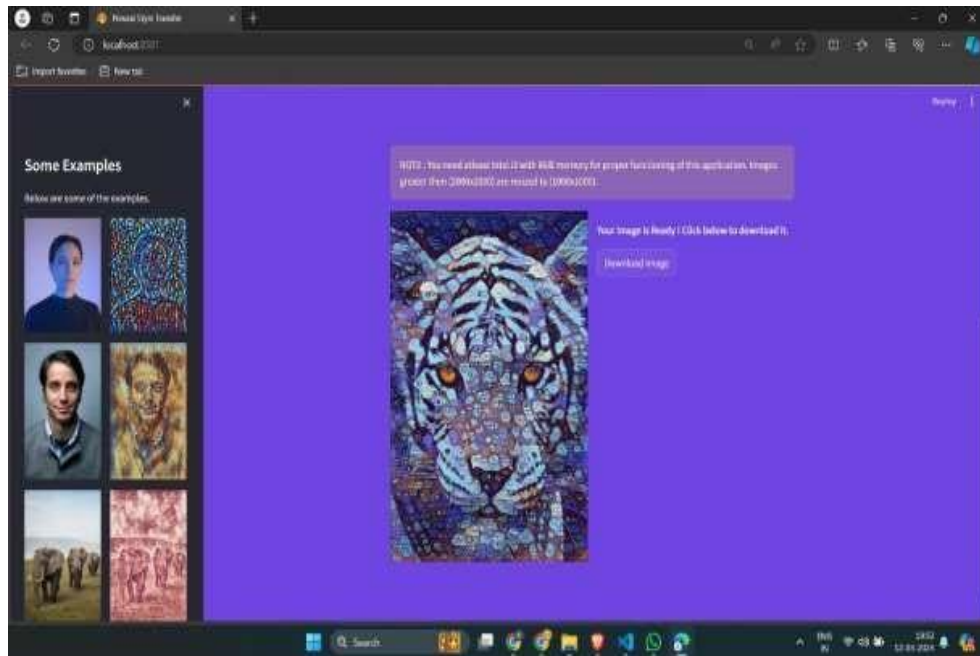


Fig 7.8 Output Image 1

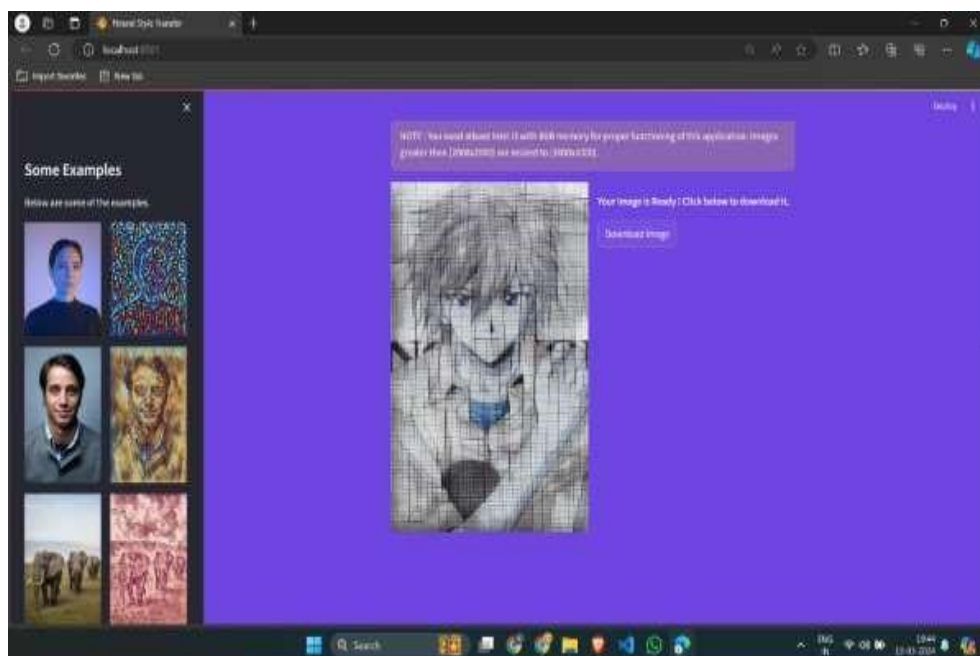


Fig 7.9 Output Image 2

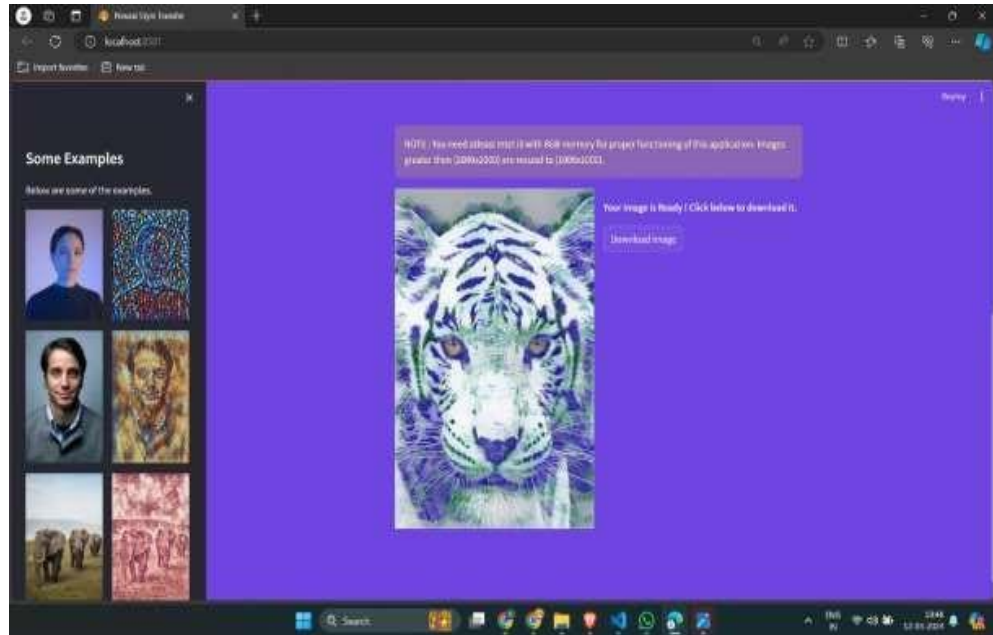


Fig 7.10 Output Image 3

7.5 PRESENTATION OF RESULTS

Table 7.1 Comparative Analysis of Neural Style Transfer Approaches

Approach	Efficiency	Real-time Performance	Quality
Gatys	Moderate (50-70%)	no	High (80-90%)
AdaIN	High (70-90%)	yes	Moderate (70-80%)
Johnson	High (70-90%)	yes	Moderate (60-70%)
Our method (hybrid model-vgg 19)	High (92.7%)	yes	Moderate (80%)

The comparison of different approaches to neural style transfer reveals distinct trade-offs in terms of efficiency, real-time performance, and

quality. The Gatys approach[1] demonstrates moderate efficiency, ranging between 50-70%, but excels in producing high-quality stylized images, scoring between 80-90%. In contrast, the AdaIN[2] and Johnson[3] approaches prioritize efficiency, ranging from 70-90%, enabling real-time performance, yet sacrificing some quality compared to Gatys, scoring between 70-80% and 60-70%, respectively. Our method, employing a hybrid model based on VGG 19, emerges as highly efficient, with an efficiency score of 92.7%. Remarkably, it achieves real-time performance while maintaining a competitive quality of 80%. This balance positions our method as a promising choice for applications requiring both efficiency and high-quality stylized image outputs.

7.6 PERFORMANCE EVALUATION

The performance evaluation of different approaches to neural style transfer highlights significant variations in efficiency, real-time performance, and quality. The Gatys approach[1], while excelling in producing high-quality stylized images, suffers from moderate efficiency and is not suitable for real-time applications due to its computational demands. In contrast, the AdaIN[2] and Johnson[3] approaches prioritize efficiency, enabling real-time performance, but at the cost of slightly lower quality compared to Gatys. Our method, utilizing a hybrid model based on VGG 19, demonstrates exceptional performance, with high efficiency, real-time capability, and competitive quality. Achieving an efficiency score of 92.7%, our method stands out as highly efficient, while maintaining a quality score of 80%, comparable to the Gatys approach. The combination of efficiency, real-time performance, and quality positions our method as a promising choice for applications requiring rapid and high-quality neural style transfer.

CHAPTER 8

CONCLUSION AND FUTURE WORK

8.1 CONCLUSION

- This technique helps to recreate the content image in the style of the reference image. It uses Neural Networks[14] to apply the artistic style from one image to another due to which it shows the result of high perceptual quality. Neural style transfer opens up endless possibilities in design, content generation, and the development of creative tools.
- Our study illustrates the advantages of combining international and local losses in the transfer of image style. Fusion architecture refers to designed. A style loss function based on a local method specified in numerous layers is employed on the one hand to maintain the detailed design

8.2 FUTURE WORK:

Potential future work in neural style transfer and image transformation encompasses various avenues aimed at enhancing speed, flexibility, and overall quality. Researchers are continually striving to push the boundaries of existing methodologies while addressing practical limitations for real-time applications.

Speed Optimization:

The optimization of methods like Gatys' neural transfer[14] for real-time applications remains a significant focus. Techniques such as parallelization, algorithmic optimizations, and hardware acceleration

(e.g., GPUs, TPUs) are being explored to expedite the style transfer process.

Research efforts are directed towards developing efficient algorithms that can achieve comparable results to Gatys' method but with significantly reduced computational overhead.

Hybrid Approaches:

One promising avenue involves exploring hybrid approaches that combine the artistic quality of Gatys' method with the efficiency of other techniques such as Adaptive Instance Normalization (AdaIN)[2]. By leveraging the strengths of different methods, researchers aim to create novel frameworks capable of producing high-quality stylized images efficiently.

Real-time Style Control:

There is growing interest in enabling users to dynamically control the style of images in realtime, even during live video streams[13].

Research in this area involves developing algorithms and interfaces that allow users to interactively adjust style parameters and observe immediate changes in the stylized output.

Data Efficiency:

Addressing the data requirements of existing style transfer methods is crucial. Researchers are investigating techniques to reduce the dependency on large training datasets.

Few-shot or even zero-shot style transfer methods[15] are being explored, where the model can adapt to a new style with limited or no training data.

User Studies and Evaluation Metrics:

Conducting comprehensive user studies and defining appropriate evaluation metrics are essential for understanding user preferences and assessing the quality of stylized outputs.

Metrics beyond traditional perceptual measures, such as content preservation, style fidelity, and temporal coherence[14] (for videos), are being developed to provide a more nuanced evaluation.

Dynamic Style Adaptation:

Future research endeavors include exploring dynamic style adaptation mechanisms that can adjust stylization parameters based on changing input conditions or user preferences.

Adaptive algorithms[2] that can dynamically modify the stylization process in response to environmental cues or user feedback hold promise for enhancing user experience and applicability in diverse scenarios.

Robustness and Generalization:

Enhancing the robustness and generalization capabilities of style transfer models across different domains and input modalities is another important research direction.

Techniques for domain adaptation, style interpolation[11], and cross-modal style transfer[12] are being investigated to extend the applicability of style transfer methods to a wide range of tasks and scenarios. By delving into these avenues, researchers aim to propel the field of neural style transfer and image transformation towards greater efficiency, flexibility, and usability, ultimately unlocking new possibilities for creative expression and practical applications.

8.3 REFERENCES

- [1] Gatys, L. A., Ecker, A. S., Bethge, M. (2016). "Image Style Transfer using CNNs." In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- [2] Huang, X., Belongie, S. (2017). "Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization." In IEEE International Conference on Computer Vision (ICCV), 2017.
- [3] Johnson, J., Alahi, A., Fei-Fei, L. (2016). "Perceptual Losses for Real-Time Style Transfer and Super-Resolution." In European Conference on Computer Vision, 2016.
- [4] Wei, L.-Y., Levoy, M. (2000). "Fast Texture Synthesis using Tree Structured Vector Quantization." In Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques, 2000.
- [5] Elad, M., Milanfar, P. (2017). "Style Transfer via Texture Synthesis." In IEEE Transactions on Image Processing, 2017.
- [6] Simonyan, K., Zisserman, A. (2014). "Very Deep Convolutional Networks for LargeScale Image Recognition." In International Conference on Learning Representation, 2014.
- [7] Dosovitskiy, A., Brox, T. (2016). "Generating Images with Perceptual Similarity Metrics." In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- [8] Khatter, H., Arif, S., Singh, U., Mathur, S., Jain, S. (2021). "Product Recommendation System for E-Commerce using Collaborative Filtering and Textual Clustering." In Third International Conference on Inventive Research in Computing Applications (ICIRCA), 2021.

- [9] Sharma, S., Gupta, S., Gupta, D., Ju neja, S., Singal, G., Dhiman, G., Kautish, S. (2022). "Recognition of Gurmukhi Handwritten City Names Using Deep Learning." In Hindawi Limited, 2022.
- [10] Mahendran, A., Vedaldi, A. (2014). "Understanding Deep Image Representations by Inverting." CVPR, 2014.
- [11] Liu, F., Shen, C., Lin, G. (2015). "Deep Convolutional Neural Fields for Depth Estimation from a Single Image." In CVPR, 2015.
- [12] Li, C., Wand, M. (2016). "Precomputed Real-time Texture Synthesis with Markovian Generative Adversarial Networks." In European Conference on Computer Vision, 2016.
- [13] Yang, C.-Y., Ma, C., Yang, M.-H. (2014). "Single-Image Super-Resolution: A Benchmark." In European Conference on Computer Vision, 2014.
- [14] Jing, Y., Yang, Y., Feng, Z., Ye, J., Yu, Y., Song, M. (2020). "Neural Style Transfer: A Review." In IEEE Transactions on Visualization and Computer Graphics, 26(11), 3365-3385, 2020.
- [15] Hou, Q., Cheng, M.-M., Hu, X., Borji, A., Tu, Z., Torr, P. (2017). "Deeply Supervised Salient Object Detection with Short Connections." In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

GitHub Link - <https://github.com/KIET-Github/CS-2024-B/tree/main/PCS24-53-KalashJain>

RESEARCH PAPER ACCEPTANCE PROOF



REG : ICICC 2024: Paper Notification 276

2 messages

ICICC 2024 <icicc.ui@gmail.com>

To: Kalashjain <kalashjain171@gmail.com>

Mon, 20 Nov, 2023 at 11:11 am

International Conference on Innovative Computing and Communication (ICICC-2024) – A Flagship Conference

Dear Author(s),

Greetings from ICICC 2024!

We congratulate you that your paper with submission ID 276 and Paper Title '**Review of various Neural Style Transfer Methods: A Comparative Study**' has been accepted for publication in the Springer LNNS series [indexing: SCOPUS, INSPEC, WTI Frankfurt eG, zbMATH, SCImago; All books published in the series are submitted for consideration in Web of Science]. This acceptance means your paper is among the top 20% of the papers received/reviewed. **Our registration process has started, and we have left with limited registrations. Kindly submit your registration fees as early as possible.**

You are requested to register as soon as possible and submit the following documents to icicc.ui@gmail.com at the earliest.

1. Final CRC as per the springer LNNS format. (See <https://icicc-conf.com/downloads>)
2. Copy of e-receipt of registration fees. (For Registration, see <https://icicc-conf.com/registrations>)
3. The final revised copy of your paper should also be uploaded via Microsoft CMT.

The reviewer's comments are given at the bottom of this letter, please improve your paper as per the reviewer's comments. While preparing the final CRC manuscript, kindly check the following Google link of proceedings of the previous International Conference on Innovative Computing and Communication:

<https://scholar.google.com/citations?hl=en&user=ffvhHUA AAAAJ>

and it is suggested to cite the relevant latest papers matching the area of your current research paper.

The paper prior to submission should be checked for plagiarism from licensed plagiarism softwares like Turnitin/iAuthenticate etc. The similarity content should not exceed 15%.

Pay registration fees via Bank Transfer to the following account:

Account Name: UI-EDUCON
Account No: 510909010229212
Bank: City Union Bank
Branch: Sector-8, Rohini, Delhi
IFSC Code: CIUB0000244

Pay via online portal (Indian Authors):

https://pages.razorpay.com/pl_IOTjy0iZ8kxq43/view

For International Authors (Outside India), Please use Paypal with extra 5% service charges:

https://www.paypal.com/paypalme/ICICCConference?locale.x=en_GB

With Regards

CERTIFICATES :-





Review of various Neural Style Transfer Methods: A Comparative Study

¹Akash Goel, ²Palak Singh, ³Ragini Rani, ⁴Kalash Jain

^{1,2,3,4}Department of Computer Science, KIET Group of Institutions, Delhi-NCR, Ghaziabad, India

Abstract— NST, or Neural Style Transfer has revolutionized the field of image processing by allowing the amalgamation of artistic styles to photographs. First introduced by Gatys et al., NST relies on a slow and iterative optimization process. However, recent advances have introduced faster and more efficient approaches, such as Adaptive Instance Normalization (AdaIN) and Johnson's method. Gatys's method, which laid the foundation for NST, uses a CNN (Convolutional Neural Network) to extract information about the content of an image and artistic features or style of an image. Based on reducing the heterogeneity between features of content images and style images, this procedure although ultramodern, is tedious. By reconceptualize model normalization, AdaIN initiated a innovative technique for rapid amalgamation of content and style features from random images. It terminates the requirement of time-consuming optimization by focusing on real-time artistic shift with excellent mouldability. Johnson's technique involves a unique learning experience using sensory loss and previously learned interactions to demonstrate high-contrast tasks This allows for better preparation through the use of localized activities and in imagery on

instantaneous analysis, resulting in a greater mixture of the two parts. This research studies the comprehensive insight into NST techniques and their evolution, highlighting potential approaches for image processing and the resolution method.

Keywords— Pioneering, Fusion, Artistic Imprint, Visual Stimuli, Iterative Optimization, Feature Statistics, Real-Time Alterations, Instance Normalization, Style Conversion, Network Functionalities, Style Adaptation

I. INTRODUCTION

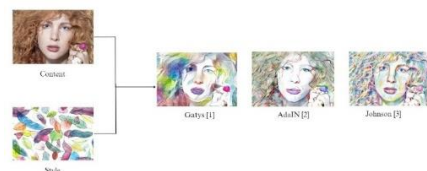
Explained in the Gatys et al.'s "Neural algorithms of art form", Neural Style Transfer (NST) involves decomposition that integrates the artistic object's characteristics with another feature. Convolutional Neural Networks (CNNs), that combines artistic style with images and its mobility. Gatys and his colleagues have illustrated CNN's ability to extract both material and process information from visual sources through application. The CNN piece claims to represent the content and content of the image statistics for styles. Their approach, which resulted in custom images that frequently optimize for desired matches. CNN-featured

distribution, got the field out of constraints and imposed by specific means or results from ground truth, it opened the door to the development of neural transfer. The launch of the NST has generated a lot of excitement ever since contributing to other efforts in business as well as in education seeking to innovate and expand fundamental algorithms. Moreover, it has been successfully used in a variety of industries and systems, consisting of as Prisma and Deep Forger. However, the most enormous drawback of this technique is the prolonged iterative optimization technique. In this paper, we're going to speak roughly some other crucial neural fashion shift method known as Adaptive Instance Normalization (AdaIN), a unique neural fashion transfer mechanism [2]. AdaIN combines the excellent features of two preceding strategies, supplying more desirable effectiveness and greater versatility of optimization-based totally methods. This indicates that AdaIN can be used for actual-time fashion changes, making it highly adaptable. AdaIN is a useful and powerful tool for creative style conversion on the grounds that, in evaluation to Gatys' preceding approach, it enables fast style modification and application without the want for complicated optimizations. AdaIN gives a singular viewpoint at the characteristic of example normalization (IN) in normalizing function facts that seize the fashion statistics of an photo. This technique involves editing the statistical traits of the enter content material to align with the traits of the input style, resulting in a combination of content and fashion. It's like seamlessly integrating the intrinsic features of one image with the melodious ingenious characteristics of any other.

This approach is remarkably faster than the previous method by Gatys et al., with no loss of flexibility in transferring to newstyles. It also provides runtime user control, making it a convenient and effective tool for neuromorphic transfer without the need to modify the training procedure but they were often limited to a single style.

Many problems are involved in converting images, such as sharpening a noisy image, turning a low-quality image into a high-quality image, or adding color to a grayscale image. In computer vision, tasks include recognizing objects in images or estimating object depth.

Another way to solve these image transformation problems is given by Johnson [3] and his colleagues which train a neural network to generate an output image by using the input image as a reference. However, instead of comparing each pixel of the predicted image with the actual image, which can be very detailed and slow, this method uses advanced features derived from a pretrained network to assess the resemblance between images. During training, this method, called perceptual loss, evaluates how closely the generated image resembles the real image more effectively than pixel-by-pixel comparison. When it's time to use the network trained on new images, it operates in real time, which is very convenient. Essentially, it combines the best of both worlds: accurate training using high-level features and real-time effectiveness.



In this, we compare style transfer outputs achieved with the methods of

Gatys et al. [1], AdaIN [2], and Johnson [3]

II. RELATED WORK

1. Image processing and filtering -

The process of creating artistic images involves simplifying and abstracting the original images. Therefore, it makes sense to explore and combine relevant image processing filters for the enhancement of a given photograph. For eg, in their work [4], Winnemoller and colleagues were the first to use bilateral filters and other than Gaussian filters to automatically generate active shapes image. Among the various image-based artistic rendering techniques, image filtering-based methods are normally simple to practice and effective. However, one limitation is their limited range of art styles.

2. Visual Texture Modelling - The field of visual texture modelling has long been central to structural synthesis [5]. Over time, two main approaches have emerged for visual texture modelling: parametric texture modelling involving a abstract statistics and non-parametric texture modelling using Markov Random Fields (MRF).

2.1 Parametric Texture modelling with summary statistics- Texture modelling can take the approach of gathering image statistics from a texture sample and use these abstract statistical properties to represent the texture. This concept was initially introduced by Julesz, who considered texture as pixel-based statistics of order N . Afterward, work in explored the use of set feedback, filtering for texture analysis, instead of direct pixel-based measurement. Based on this, Portilla and Simoncelli developed a texture model based on the response of a multi-scale directional filter and used a

gradient descent method to enhance the synthesis outcome.

In a most latest parametric texture modelling method, as presented by Gatys et al. [6], was the first to leverage abstract statistics within the context of a Convolutional Neural Network (CNN). They introduced a new way to represent Gram-based to model textures, which involves examining the correlations between filter responses in different layers of a pre-trained classification network, such as the VGG network [7]. Specifically, this representation of Gram-based encodes the second-order statistics of the CNN filter response set.

2.2 Non-parametric Texture Modelling with MRFs - Another way to understand structural modelling is by using non-parametric resampling. Non-parametric methods like Markov Random Field (MRF) models suggest that style of each pixel in a texture image depends on nearby pixels. Based on this hypothesis, Efros and Leung introduced a method to synthesize individual pixels. This is achieved by exploring for similar areas in the source texture image and allocating pixels accordingly. Their work is one of the first nonparametric methods using MRF and it's like finding pieces in one picture and putting them together to make a new image.

Expanding on these advancements, Wei and Levoy took steps to enhance the efficiency of neighbourhood matching process by systematically using a fixed neighbourhood. This technique continues to align with the principles of non -parametric texture modelling using MRFs, highlighting the importance of pixel context in texture synthesis.

3. Image Reconstruction

In the field of computer vision, an important step involves filter an abstract interpretation from the input image.

Image reconstruction, on the other hand, does this process in reverse; it aims to recreate the entire original image from an abstract representation. The goal here is to understand and extract the information contained in this abstract representation. Their main focus is on image reconstruction algorithms based on convolutional neural network (CNN) representations, which fall into two categories: Image-Optimisation-Based Online Image Reconstruction and Model-Optimisation-Based Offline Image Reconstruction [16].

3.1 Image-Optimisation-Based Online Image Reconstruction:

The CNN representation inversion method was originally introduced by Mahendran and Vedaldi [10]. When tasked with inverting a CNN representation, their algorithm goes through an iterative optimization process, typically starting with random noise. This process continues until the image matches the desired CNN image. Since it depends on gradient descent in image space, this method can be time consuming, especially for larger reconstructed images.

3.2 Model-Optimisation-Based Offline Image Reconstruction:

To tackle the efficiency challenges posed by [10], Dosovitskiy and Brox [8] proposed a different strategy. They suggest to pre - train a feed forward network, which offloads the computation to the training phase. During testing, the reverse process became easy with a simple network switch. This method substantially accelerates the image reconstruction procedure. In their later study [8], they improved the results by incorporating a Generative Adversarial Network (GAN) [9].

4. Neural Style Transfer was introduced by Gatys et al. [1], initially relied on

slow optimization. Johnson et al. [3] accelerated the process using perceptual losses. Ulyanov et al. proposed a faster style transfer with a pre-trained style-specific feed-forward network. Later, they improved it by introducing conditional instance normalization (CIN) [10]. Gatys et al. [11] extended CNN to handle arbitrary styles using an encoder. Cheng et al. proposed a patch-based style swapping, while Huang et al. proposed Adaptive Instance Normalization (AdaIN) [2]. Further extensions include whitening and coloring by Li et al., the decorated module follows the style of Sheng et al. and meta-networks for style transfer [3].

5. Feed-Forward Image Transformation

Various image processing methods, such as depth estimation, semantic segmentation and surface normal prediction, uses a fully-convolutional neural networks to generate detailed scene labels. They train these networks using the per-pixel classification or regression loss functions. Recent advances in surface and depth normal estimation using feed-forward convolutional networks, where training involves per-pixel regression or classification losses. Some technique surpass the per-pixel loss by incorporating techniques such as penalizing image gradients, recurrent CRF inference layers [13], or CRF loss layers to elevate the quality and overall consistency of the generated output.

5.1 Perceptual Optimization.

Recent research explores perceptual optimization, where images are generated from the optimization processes driven by high-level features extracted from convolutional networks. Objectives include maximizing class prediction scores [12] and dissecting individual features to gain a deeper

understanding of network functionality. This technique also creates convincing "fooling" images with high confidence. Mahendran and Vedaldi has set the stage by introducing feature inversion, a process driven by minimizing the loss of feature reconstruction. This innovative approach reveals information stored on different network layers. In parallel, Dosovitskiy and Brox have made significant contributions by training a feed-forward network designed to invert convolutional features, providing a faster alternative to Mahendran and Vedaldi's optimization process. However, it is important to highlight a key difference: while Dosovitskiy and Brox's network depends on per-pixel reconstruction loss, their network directly optimizes object reconstruction loss [10].

5.2 Style Transfer

Artistic style transfer introduced by Gatys et al. [1], a technique that combines image content and style through feature reconstruction and style reconstruction losses [10]. A same technique had been previously used for texture synthesis. This is computationally intensive. To mitigate this, they prepare a feed forward network to rapidly approximate style transfer solutions. At the same time, [14] also proposed a feedforward method for achieving speedy style transfer.

5.3 Image Super-Resolution.

Image super-resolution has been extensively explored with various techniques. Yang et al. [15] conducted a comprehensive evaluation of these methods, categorizing them into prediction-based, edge-based, statistical methods, patch-based and sparse dictionary approaches. Before the rise of convolutional neural networks, these

methods included techniques like bilinear, bicubic, Lanczos, and more. Recent advancements include the work by [1], impressive results for enhancing the resolution of individual images. This is achieved through the utilization of a three-layer convolutional neural network combined with a per-pixel Euclidean loss, highlighting the effectiveness of this approach.

6. In the context of our project on Neural Style Transfer, recommender systems play an important role in recommending art styles that match the user's preferences and content. While our project focuses on image transformation using neural networks, the e-commerce recommendation system aims to improve product recommendations for users [18]. As part of our Neural Style Transfer project, this research highlights the power of deep learning techniques, especially CNNs, in solving complex recognition tasks. While Neural Transfer focuses on transforming visual content, this research shows the broader impact of deep learning in various applications, including handwritten character recognition for automation Postal [19].

III. METHODOLOGY

Gayts Method

The research presented in the main text uses VGG-Network, Certainly, they used a 19-layer convolutional neural network known for its outstanding performance in visual object recognition. In their method, they make use of all 16 convolutional layers and 5 pooling layers in this network, except for the fully connected layers. They use the publicly available VGG model within the caffe framework. To synthesize images, they chose average pooling instead of maximum pooling,

which improves the gradient flow and leads to more pleasing results.

In a neural network, each layer acts as a complex set of filters, and these filters become more complex as you move through the network. As an input image (\vec{a}) is passed through the network, each layer processes it and generates what is known as a feature map. The quantity of feature maps in a layer corresponds to the count of distinct filters present within that layer. Each feature map is characterized by its size, which is determined by the multiplication of its height and width. An experiment is carried which involves a random noisy image to gain insights into the information gathered by each layer of the network. Gradient descent method utilizes in this experiment to adjust the image iteratively until it matches the response characteristics of a specific image of interest.

Throughout this process a comparison is made at each layer between the original image (\vec{b}) feature maps and generated image (\vec{a}) feature map. Then the squared error loss is calculated to evaluate the alignment of these feature representations. At each layer this loss serves as a metric for measuring the dissimilarity between the original image and the generated image. The researchers are able to visualize and understand the nature of information encoded at various levels of the neural network by using this technique. Basically, this approach is valuable for reconstructing the content of an image that accurately captures the features represented in the network layer.

$$L_{content}(\vec{b}, \vec{a}, l) = \frac{1}{2} \sum_{m,n} (F_{mn}^l - P_{mn}^l)^2$$

The gradient of this loss with respect to the activations in layer l is equivalent.

$$\frac{\partial L_{content}}{\partial F_{m,n}^l} = \begin{cases} (F^l - P^l)_{mn} & \text{if } F_{mn}^l > 0 \\ 0 & \text{if } F_{mn}^l < 0 \end{cases}$$

from there the gradient for image \vec{a} can be calculated using back propagation of the standard error. Through the process of modifying the initial random image (\vec{a}), it is transformed until it generates a matching response in a specified layer of the CNN, mirroring that of the original image (\vec{b}).

At every layer within the neural network, the style representation is constructed using the CNN's response. This type of representation calculates the correlation between different filter responses, with the spatial extent of the input image taken into account through expectations. The correlations between these features are signified by the Gram matrix G^l , which has dimensions $N^l \times N^l$. In this matrix, each G_{mn}^l value indicates the scalar product of vectorized feature maps m and n in layer l .

$$G_{mn}^l = \sum_k F_{mk}^l \cdot F_{nk}^l$$

To create textures that simulate the image's style, the gradient descent method is employed. Starting with the white noise image, adjusting it several times to replicate the original image's stylistic appearance. This adaptation requires minimizing the mean squared difference between the elements of the Gram matrix for the original image and the Gram matrix for the generated image. Therefore, assuming that \vec{b} stands for the original image and \vec{a} represents the generated image and B^l and G^l denote their corresponding

representations in layer l . The contribution of each layer to the overall loss is then evaluated by-

$$S_l = \frac{1}{4N_l^2 M_l^2} \sum_{m,n} (G_{mn}^l - B_{mn}^l)^2$$

And the complete loss is

$$L_{style}(\vec{b}, \vec{a}) = \sum_{l=0}^L v_l S_l$$

The total loss contribution of each layer is determined by the coefficient v_l , and the specific values used will be detailed in the following results section. The analytical computation of derivative of S_l concerning the activations in layer l can be performed.

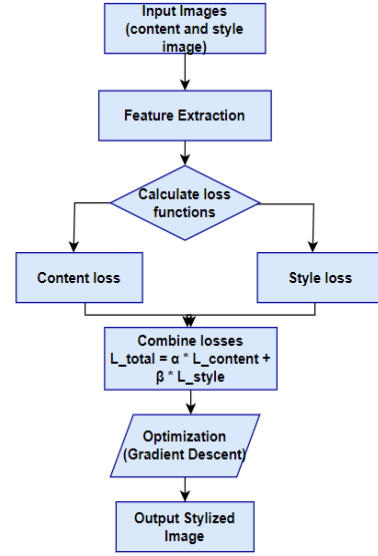
$$\frac{\partial S_l}{\partial F_{mn}^l} = \begin{cases} \frac{1}{N_l^2 M_l^2} \left((F^l)^T (G^l - B^l) \right)_{ji} & \text{if } F_{mn}^l > 0 \\ 0 & \text{if } F_{mn}^l < 0 \end{cases}$$

The gradient of S^l associated with the activations easily calculated in the network's lower layers using the standard error backpropagation technique.

The objective is to generate images that combine a photograph's content with the style of a painting by reducing the dissimilarity between a white noise image and the manner in which the photograph's content is represented in a specific layer of the network. Additionally, here the aim is to minimize the differences between drawing style representations across multiple CNN layers, so if we have \vec{o} as a photo and \vec{q} as a work of art, then our objective is to minimize the loss function which is:

$$L_{total}(\vec{o}, \vec{q}, \vec{z}) = \alpha L_{content}(\vec{o}, \vec{z}) + \beta L_{style}(\vec{q}, \vec{z})$$

Here, α and β represent the weighting coefficients for the recreation of content and style, respectively [1].



Flowchart of Gayts Method

ADaIN Method

The style transfer network does something quite interesting. It accepts two images as input: one for the content image and another for the style image. It then creates an output image that merges the content from one with the artistic style from the other. To do this, the AdaIN method offers a simple approach. It uses an encoder-decoder structure, like a translator, to achieve this magic. The encoder decodes the input images and captures their essence, while the decoder takes this encoded information and mixes it to produce the final image with the desired style.

The content material photograph (c) and fashion picture (s) skip via the encoding community, generating function maps. These function maps are then forwarded to the AdaIN layer wherein the combined characteristic map is computed. This combined characteristic map is then surpassed to the decoder network, which is initialized randomly, and the decoder feature because the

generator for the photograph converted thru neural fashion switch.

$$t = AdaIn(f_c, f_s)$$

$$T = g(t)$$

In AdaIn layer, the style function map (fs) and the content material function map (fc) play vital function inside the introduction of a combined characteristic map (t). This blended feature map is then used by the deciphering community which is represented by the function g, to generate the final stylized photograph.

Encoder

The encoder is an integral part of the pre-trained VGG19 For example. This model was initially trained on the ImageNet dataset and this section was later removed for convenience process of stylized image creation.

In AdaIN layer, features of both the content and style images are processed. The following equation is used to define this layer:

AdaIn formula

$$AdaIn(p, q) = \sigma(q) \frac{p - \mu(p)}{\sigma(p)} + \mu(q)$$

In this equation, " σ " signifies the standard deviation, and " μ " is used to represent the mean of the relevant variable. Notably, the mean and variance of the content feature map (f_c) are adjusted to align with the mean and variance of the style feature maps (f_s).

It's worth emphasizing that the AdaIN layer, as introduced by the authors, does not incorporate any additional parameters beyond mean and variance. This layer is not equipped with trainable parameters. This is why it's implemented as a Python function rather than being integrated as a Keras layer. The function processes both style and content feature maps, determines the mean and standard deviation of the

image, and then generates an adaptive feature map normalized by the instance.

Decoder

The authors clearly state that the decoder network should mirror the architecture of the encoder network. To achieve this, they inverted the encoder configuration symmetrically. They incorporated Layers of UpSampling2D to improve the feature's map of spatial resolution.

It is important to emphasize that the authors recommend avoiding the use of any type of normalization layer within the decoder network. In fact, they demonstrate that including batch or version normalization has a negative impact on overall network performance.

Loss Functions

In constructing the loss functions which is utilized in the neural style transfer model, we apply the method proposed by the author[2]. They recommend utilizing the pre-trained VGG-19 model for computing the network loss function. It's worth emphasizing that this loss function only applies when training a decoder network.

The total loss (L_t) is a composite of two elements: content loss (L_c) and style loss (L_s). The lambda parameter (λ) is utilized to manage the extent of style transfer.

$$L_t = L_c + \lambda L_s$$

Content loss

The content loss is computed as the Euclidean distance between the features of the content image and the features of the following neural-style transferred image.

$$L_c = ||f(g(t)) - t||_2$$

When content is lost, the authors recommend using the AdaIN t layers output as the content target, instead of

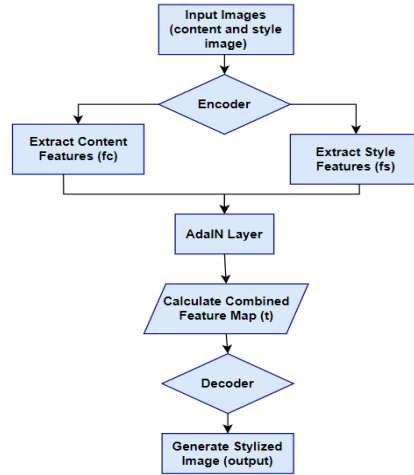
employing the original image's features as the target. This adjustment is introduced to accelerate the convergence process (ϕ_i).

Style loss

In contrast to the more commonly employed Gram Matrix, the authors introduce a novel approach for computing style loss. This involves calculating the disparity in statistical properties, particularly mean and variance, offering a more conceptually straightforward metric. This is expressed as follows:

$$L_s = \sum_{i=1}^L \|\mu(\phi_i(g(t))) - \mu(\phi_i(s))\|_2 + \sum_{i=1}^L \|\sigma(\phi_i(g(t))) - \sigma(\phi_i(s))\|_2$$

In this equation, " ϕ " represents the VGG-19 layers used to calculate the loss [2].



**Flowchart of AdaIN method
Johnson Method**

The system comprises of two main features which are: an image transformation network denoted as fv and a loss network represented as ϕ that is used to denote several loss function l_1, \dots, l_n . Weight v ; is parameter of deep residual convolutional network which is

image transformation network, it convert input images denoted as " p " and transform them into output image \hat{q} through mapping $\hat{q} = fv(p)$. Each loss function computes a scalar value $l_i(\hat{q}, q_i)$ that measures the dissimilarity between the output image \hat{q} and the target image q_i . These loss functions are important for training the image transformation network, which uses gradient descent to minimize a weighted combination of the various loss functions:

$$V^* = \arg \min_v E_{p, \{q_i\}} [\sum_{i=1} \lambda_i l_i(fv(p), q_i)]$$

To overcome the limitations associated with per-pixel loss and improve the ability to capture perceptual and semantic distinctions between images, we adopt a unique approach. These methods are based on the idea that convolutional neural networks, initially trained for image classification, have already internalized the perceptual and semantic details that we seek to evaluate with our loss functions. Therefore, we employ a pre-trained ϕ network to classify images as a constant loss network to determine our loss functions. We employ a loss network, denoted as ϕ to determine the object reconstruction loss ℓ_{feat}^ϕ and the style reconstruction loss ℓ_{style}^ϕ to calculate dissimilarity between the style and content images. With each input image represented as p , we set our sights on content target q_s and a style target \hat{q} . To change the style, the target of the content q_c , align with the input image p and the output image \hat{q} , endeavors to fuse the content of $p=q_c$ with the desired style represented as q_s . The network is trained according to these style objectives. In the context of ultra-high resolution, where input p is a

low-resolution input, content target q_c is a high-resolution ground-truth image, and style reconstruction loss does not play a prominent role in the training process of the network according to the super-resolution coefficient.

Image Transformation Networks

This use a design that bypasses traditional pooling layers, opting for split and split structures.

For super resolution, we use residual blocks and specific convolution layers based on the up sampling factor " f ", rather than relying on bicubic interpolation. This approach is computationally efficient and increases the effective size of receptive field.

Perceptual Loss Functions

We establish two separate perceptual loss functions that allow us to evaluate high-level perceptual and semantic disparities between images. These loss functions are derived from a pre-trained classification network known as ϕ , which itself takes the form of a deep convolutional neural network. In our experimental setups, we systematically utilize a 16-layer VGG network pre-trained on ImageNet to serve as the ϕ loss network.

Feature Reconstruction Loss

Instead of emphasizing a precise pixel-to-pixel match between the output image $\hat{q}=fv(p)$ and the target image q . We achieve this by examining the activation of the j th layer in network ϕ during processing of image p , denoted by ϕ_j . If j represents a convolutional layer, then $\phi_j(p)$ will be a feature map of size $C_j \times H_j \times V_j$.

$$\ell_{feat}^{\phi,j}(\hat{q}, q) = \frac{1}{C_j H_j V_j} \|\phi_j(\hat{q}) - \phi_j(q)\|_2^2$$

Style Reconstruction Loss

The loss of feature reconstruction loss functions act as a penalty on the output image \hat{q} if it diverges from the target q in terms of content. We utilize the activations $\phi_j(p)$ at the j th layer of the network ϕ for the input p . These activations materialize as a feature map of size $C_j \times H_j \times V_j$. Now, the Gram matrix $G_j^\phi(p)$ to be the $C_j \times C_j$ matrix constructed by calculating the relationship between these features are given by

$$G_j^\phi(p)_{c,c'} = \frac{1}{C_j H_j V_j} \sum_{h=1}^{H_j} \sum_{v=1}^{V_j} \phi_j(p)_{h,v,c} \phi_j(p)_{h,v,c'}$$

The Gram matrix $G_j^\phi(p)$ is a representation of how features in $\phi_j(p)$ co-activate. This can be seen as a measure of the uncentered covariance between these features, with treating each grid location as a distinct data point. To compute $G_j^\phi(p)$ efficiently, reshape $\phi_j(p)$ into a matrix ψ and calculate it as $\psi\psi^T$, which is then normalized by $C_j \times H_j V_j$.

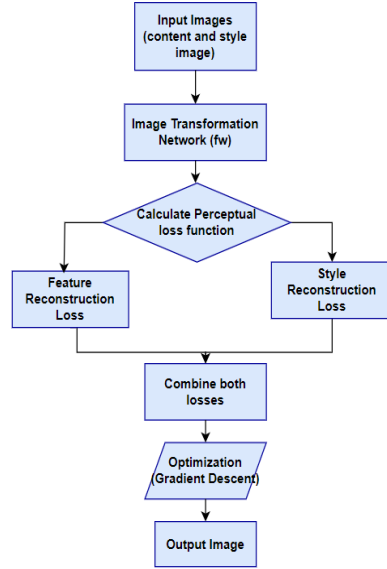
$$G_j^\phi(p) = \psi\psi^T / C_j H_j V_j.$$

The style reconstruction loss can be calculated using the squared Frobenius norm, which represents the difference between the Gram matrix of the generated image and the target image.

$$\ell_{style}^{\phi,j}(\hat{q}, q) = \|G_j^\phi(\hat{q}) - G_j^\phi(q)\|_F^2.$$

Pixel loss, which quantifies the dissimilarity between the output image \hat{q} and the target q , is determined by calculating the squared Euclidean distance normalized by their size $C \times H \times V$, defined as $\ell_{pixel}(\hat{q}, q) = \|\hat{q}, q\|_2^2 / CHV$. It's applicable when a

ground-truth target q is available for the network to match [3].



Flowchart of Johnson's method

IV. COMPARISON AND ANALYSIS

Approach	Efficiency	Real-time Performance	Quality
Gatys	Moderate (50-70%)	no	High (80-90%)
AdaIN	High (70-90%)	yes	Moderate (70-80%)
Johnson	High (70-90%)	yes	Moderate (60-70%)

Comparative Analysis of Neural Style Transfer Approaches

In our study, we conducted a comprehensive comparative analysis of three important neural transfer methods, namely Gatys neural transfer, AdaIN (real-time arbitrary transfer), and Justin Johnson's Fast Neural Style Transfer. This analysis focuses on three important aspects: efficiency, real-time performance, and quality of the type conversion results. Neural style transfer from Gatys, known for quality art style transfer, has relatively lower efficiency due to its optimization-based approach, which often requires significant computational resources. While it delivers exceptional artistic results, Gatys method may not be suitable for the application that requires real time style transfer. On the other hand, AdaIN method known for its efficiency, it gives the perfect solution for real-time type applications. This method provides a perfect balance of speed and quality and also ensures satisfactory results of a wide range of practical applications. Johnson method is highly effective because it provides a harmonious compromise between speed and quality. Whereas Johnson's results may not reach the same artistic quality levels as Gatys' method but they still offer a good quality result that would be reasonable for various tasks. When we have to choose between AdaIN and Johnson then the factors like artistic quality, efficiency, and real-time performance should be used for the decision-making process.

V. CONCLUSION

In this research paper we have discuss the three methods of neural style transfer that involves transforming images by combining artistic styles with photographs. In this firstly we explore the spearheading work of Gatys [1] and colleagues, who presented the concept

of combining aesthetic styles with photos utilizing convolutional neural networks. Their method is robust in nature but this method can be slow and iterative, hindering the efficiency of the style transfer.

Secondly, we discuss the Adaptive Instance Normalization (AdaIn) [2] method, this method overcome the limitations of Gatys method. AdaIn method balances the flexibility and speed by utilizing instance normalization to effectively blend content and style. One of the key advantages of AdaIn method is that it allows real-time style switching with various styles, giving users control during runtime without the need to modify the training process. This innovative approach has transformed the way style transfer is conducted. It offers more efficient and flexible solution.

And the third notable method discussed in this paper is the Johnson [3] method, which utilizes perceptual loss functions and loss networks to measure perceptual and semantic disparities in images. Johnson method basically enhances the quality of image transformation and super-resolution tasks by using pre-trained image classification networks. This helps in creating more accurate and high-quality image transformations, ultimately improving the overall efficiency of the process. The paper also investigates picture change systems and also focus on their effectiveness and design principles

VI. ACKNOWLEDGMENT

This paper introduces an advance approach of neural style transfer, firstly it discusses the Gatys method of neural style transfer using CNN. To overcome the limitation of Gatys we executed

AdaIn method for real-time style transfer. AdaIn method offers a seamless integration of styles onto images.

Additionally, this paper also discusses the Johnson's perceptual loss methodology which combine both speed and precision in the style transfer process. In this, the related areas of style transfer such as image reconstruction, image processing and texture modelling are discussed. Moreover, we express our gratitude towards the spearheading work of Gatys, AdaIn and Johnson, whose commitments have essentially progressed the field of neural style transfer and visual processing.

VII. REFERENCES


- [1] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks" in IEEE Conference on Computer Vision and Pattern Recognition, 2016
- [2] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization" in IEEE International Conference on Computer Vision (ICCV), 2017
- [3] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in European Conference on Computer Vision, 2016
- [4] L.-Y. Wei and M. Levoy, "Fast texture synthesis using tree structured vector quantization," in Proceedings of the 27th annual conference on Computer graphics and interactive techniques, 2000
- [5] Michael Elad and Peyman Milanfar, "Style transfer via texture synthesis", in IEEE Transactions on Image Processing, 2017.
- [6] Leon A. Gatys, Alexander S. Ecker and Matthias Bethge, "Texture synthesis using convolutional neural networks,"

- in part of Advances in Neural Information Processing Systems 28, 2015
- [7] Karen Simonyan and Andrew Zisserman, “Very deep convolutional networks for large-scale image recognition” published in International Conference on Learning Representation, 2014.
- [8] Alexey Dosovitskiy; Thomas Brox, “Inverting visual representations with convolutional networks,” in IEEE Conference on Computer Vision and Pattern Recognition, 2016
- [9] Alexey Dosovitskiy and Thomas Brox, “Generating images with perceptual similarity metrics based on deep networks,” in Advances in Neural Information, 2016
- [10] Aravindh Mahendran and Andrea Vedaldi, “Understanding deep image representations by inverting them” in CVPR, 2014
- [11] Yosinski, J., Clune, J., Nguyen, A., Fuchs, T., Lipson, H “Understanding neural networks through deep visualization” in ICML Deep Learning Workshop, 2015
- [12] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge, “A neural algorithm of artistic style” in Computer Vision and Pattern Recognition, 2015
- [13] Fayao Liu, Chunhua Shen, Guosheng Lin, “Deep convolutional neural fields for depth estimation from a single image” in CVPR, 2015
- [14] Chuan Li & Michael Wand, “Precomputed real-time texture synthesis with markovian generative adversarial networks” in European Conference on Computer Vision, 2016
- [15] Chih-Yuan Yang, Chao Ma & Ming-Hsuan Yang, “Single-image super-resolution: a benchmark” in European Conference on Computer Vision, 2014
- [16] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu and Mingli Song, “Neural Style Transfer: A Review”, published in IEEE Transactions on Visualization and Computer Graphics 26(11):3365-3385, 2020
- [17] Qibin Hou, Ming-Ming Cheng, Xiaowei Hu, Ali Borji, Zhuowen Tu, Philip Torr “Deeply supervised salient object detection with short connections”, in IEEE Conference on Computer Vision and Pattern Recognition, 2017.
- [18] Harsh Khatter, Shifa Arif, Utsav Singh, Sarthak Mathur and Satvik Jain, “Product Recommendation System for E-Commerce using Collaborative Filtering and Textual Clustering”, in Third International Conference on Inventive Research in Computing Applications (ICIRCA), 2021
- [19] Sandhya Sharma, Sheifali Gupta, Deepali Gupta, Sapna Juneja, Gaurav Singal, Gaurav Dhiman, and Sandeep Kautish, “Recognition of Gurmukhi Handwritten City Names Using Deep Learning and Cloud Computing” in Hindawi Limited, 2022

PROOF OF PATENT PUBLICATION


2/5/24, 11:50 AM

Intellectual Property India



Office of the Controller General of Patents, Designs & Trade Marks
Department for Promotion of Industry and Internal Trade
Ministry of Commerce & Industry,
Government of India

(<http://ipindia.nic.in/index.htm>)



INTELLECTUAL
PROPERTY INDIA
PATENTS DESIGNS TRADE MARKS
GEOGRAPHICAL INDICATIONS

(<http://ipindia.nic.in/index.htm>)

Application Details	
APPLICATION NUMBER	202411002095
APPLICATION TYPE	ORDINARY APPLICATION
DATE OF FILING	11/01/2024
APPLICANT NAME	1 . Harsh khatter 2 . Akash Goel 3 . Palak Singh 4 . Ragini Rani 5 . Kalash Jain
TITLE OF INVENTION	SYSTEM AND METHOD FOR NEURAL STYLE TRANSFER FOR 3D MODEL
FIELD OF INVENTION	COMPUTER SCIENCE
E-MAIL (As Per Record)	harsh.khatter@kiet.edu
ADDITIONAL-EMAIL (As Per Record)	
E-MAIL (UPDATED Online)	
PRIORITY DATE	
REQUEST FOR EXAMINATION DATE	--
PUBLICATION DATE (U/S 11A)	02/02/2024

Application Status

APPLICATION STATUS	Awaiting Request for Examination
--------------------	----------------------------------

View Documents

<https://ipsearch.ipindia.gov.in/PatentSearch/PatentSearch/ViewApplicationStatus>

1/2

2/5/24, 11:50 AM

Intellectual Property India

➡

Filed➡Published➡RQ Filed➡Under Examination

➡Disposed

In case of any discrepancy in status, kindly contact ipo-helpdesk@nic.in