IMAGE COMPOSITION USING DEEP PAINTERLY HARMONIZATION

[1]M. Sesha Sai, [2]K. Rohith, [3]G. S. Praneeth, [4]D. Nikhila, [5]N. Pavan Kalyan
[1]Student, [2]Student, [3]Student, [4]Student, [5]Student
Department of Computer Science and Engineering,
Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam, India.

Abstract: Superimposing a component from a photograph into a painting is a difficult assignment. Applying photograph compositing methods using existing painterly stylization computations, which are worldwide to outcome a photomontage, are performed ineffectively when applied locally. We address these issues with the appropriate rules that cautiously decides the neighbourhood measures need to be considered. We guarantee both spatial and in- scale measurable consistencies to show that the two properties are critical in producing quality outcomes. To adapt to the decent variety of deliberation levels and kinds of canvases, we present a procedure to change the parameters of the exchange contingent upon the artistic creation. We show that our calculation delivers essentially preferable outcomes over photograph compositing or worldwide stylization procedures and that it empowers imaginative painterly alters that would be in any case hard to accomplish.

Keywords **- Convolutional Neural Networks (CNN), Visual Geometry Groups (VGG), Gram Matrix.**

1. INTRODUCTION

Deep neural networks have outperformed human level execution in undertakings, for example, object recognition and detection. Creating better quality art utilizing AI procedures is basic for arriving at human-like capabilities, opens another range of potential outcomes. Also, with the progression of computer equipment just as the expansion of deep learning, it is presently being utilized to create artistic artefacts.

Image composition describes how different visual elements are arranged inside a frame and thereby allowing one to create new pieces with existing ones whose main goal is to make composting unnoticeable. Numerous devices have been produced for photographic compositing, but in any case, there is no proportionate for painting.

Through this paper, we propose an algorithm which enables one to impose an object in related paintings in such a way that the composition still looks like a genuine painting. Our scheme is to transfer relevant characteristics of the painting on the embedded object and further to improvise the output quality. For this approach we introduce a two-pass mechanism: First pass achieves the transfer of the desired characteristics (Robust Coarse Harmonization) and the Second is to enhance the resultant obtained from the previous pass.

1.1 Literature Survey

Eric Risser proposed a stable and controllable model to perform Neural Texture Synthesis and Style Transfer Using Histogram Losses [12]. Implementation of deep neural network models improved the overall photorealism of the resultant [11]. With regards to photographic exchange, Luan et al. [6] limit confounds utilizing scene analysis. Gatys et al. [2] fasten up the style transferring process by producing a large size, high quality stylisation using Guided Gram Matrixes and colour histogram matchings. Ongoing methodologies supplant the Gram lattice with coordinating different measurements of neural responses [3,5]. Liao et al. [8] further improve the nature of the outcomes by presenting bidirectional dense correspondence field matches.

1.2 Related work

Gatys Neural Algorithm for Artistic Style Late work on Neural Style transfer [1] has indicated noteworthy outcomes on transferring the style of an artwork by considering the statistical measurements of layer reactions of a profound neural system.

N.A.A.S. utilizes two images to generate an output piece: the content image, and the style image. As is inferred by their marks, the style of the style picture is woven onto the content of the content picture. This is accomplished by utilizing a shrewd system which permits the computer to recognize the content and style of an image, using a convolutional neural system (conv net or CNN). The output picture (at first an irregular arrangement of pixels) is contrasted with the predetermined content and style, and is gradually changed until it matches the required outcome. The methods transfer arbitrary styles starting with one image then onto the next by coordinating the connections between component enactments extricated by a pretrained deep neural system for picture classification whose model parameters do not need to be updated during the training (i.e., Visual Geometry Group [10]). The reconstruction procedure depends on an iterative improvement system that limits the content and style losses figured from the VGG neural system. The algorithm proceeds in three stages:

1. The input picture I and style S are handled with the VGG arrange [10] to deliver a set of activation values esteemed as feature portrayals F[I] and F[S]. Instinctively, these catch the statistics that speak to the style of each picture.

2. The style initiations are mapped to the input ones. In the first methodology as proposed by Gatys et al., the whole arrangement of style enactments is utilized. Different alternatives have been later proposed, e.g., utilizing closest neighbours' neural patches [7].

3. The output picture O is reproduced through a streamlining process that tries to protect the substance of the input picture while simultaneously coordinate the visual appearance of the style picture.

1.2.1 Reconstruction Losses

1.2.1.1 Content and Style Losses

Content are the high-level features that describes the object and their arrangement in the image. Each successive mapping filters of the CNN architecture out more of the characteristics of more importance where lower levels focus on the individual pixel values. The content loss function measures how much the feature map of the generated image differs from the feature map of the source image.[14]

For style loss, we capture the textural information of the image. Style loss is defined as the difference of correlation present between the feature maps computed by the generated image and the style image using **Gram Matrix**. The Gram matrix is essentially just a matrix of dot-products for the vectors of the feature activations of a style-layer. The intuition behind using gram matrix is that we're trying to capture the statistics of the lower layers.[14]

$$\mathcal{L}_{Gatys} = \mathcal{L}_c + w_s \mathcal{L}_s \qquad (1)$$

$$\mathcal{L}_c = \sum_{\ell=1}^{L} \frac{\alpha_\ell}{2N_\ell D_\ell} \sum_{i=1}^{N_\ell} \sum_{p=1}^{D_\ell} (F_\ell[O] - F_\ell[I])_{ip}^2 \qquad (2)$$

$$\mathcal{L}_s = \sum_{\ell=1}^{L} \frac{\beta_\ell}{2N_\ell^2} \sum_{i=1}^{N_\ell} \sum_{j=1}^{N_\ell} (G_\ell[O] - G_s[S])_{ij}^2 \qquad (3)$$

where L is the total number of convolutional layers, N_ℓ the number of filters in the ℓ^{th} layer, and D_ℓ the number of activation values in the filters of the ℓ^{th} layer. $F_\ell`[\cdot] \in \mathbb{R}^{N\ell \times D\ell}$ is a matrix where the (i,p) coefficient is the p^{th} activation of the i^{th} filter of the ℓ^{th} layer and $G_\ell[\cdot] = F_\ell[\cdot]F_\ell[\cdot]T \in \mathbb{R}^{N\ell \times D\ell}$ is the corresponding Gram matrix. α_ℓ and β_ℓ are weights controlling the influence of each layer and ws controls the trade-off between the content and the style.

1.2.1.2 Histogram Loss

Histogram matching is a technique that is often used to modify a certain photograph with the luminosity or shadows of another. Pierre Wilmot et al. [12,13] found that applying the same technique to define another loss could help preserve the textures of the style picture.

$$\mathcal{L}_{hist} = \sum_{\ell=1}^{L} \gamma_\ell \sum_{i=1}^{N_\ell} \sum_{p=1}^{D_\ell} (F_\ell[O] - R_\ell[O])_{ip}^2 \qquad (4)$$

where, $R_\ell = histmatch(F_\ell[O], F_\ell[S])$

1.2.1.3 Total Variation Loss

Johnson et al. [4] showed that the total variation loss introduced by Mahendran and Vedaldi [9] improves the style transfer process by enhancing the visual quality of the output.

$$\mathcal{L}_{tv} = \sum_{x,y} \left(O_{x,y} - O_{x,y-1}\right)^2 + \left(O_{x,y} - O_{x-1,y}\right)^2 \qquad (5)$$

2. PAINTERLY HARMONIZATION ALGORITHM

The primary pass creates a midway result that is near the required style. We can design a robust algorithm that can adapt to inconceivably various styles. This pass accomplishes coarse harmonization by first performing a rough match of the colour and texture properties of the pasted region to those of semantically comparative areas in the painting. We find nearest-neighbour neural patches independently on each network layer to coordinate the responses of the pasted region and of the background. This generates an intermediate result that is a better starting point for the second pass.

Later, in the subsequent pass, we start from this intermediate result and focus on visual quality. Intuitively, since the intermediate image and the style image are visually close, we consider a single reference layer that captures the neighbourhood properties of the image which produces a correspondence map that we process to expel spatial exceptions. We then up sample to the finer levels of the system, in this manner guaranteeing that at each output location, the neural responses at all scales originate from the similar location in the painting that prompts progressively coherent textures and better-looking outcomes.

(a)

(b)

Fig 1 represents inputs- style image (a), and the content image (b)

2.1 First Pass Harmonization

This pass is very much like the technique used by Gatys et al. [1] except that we use the principle the nearest neighbour approach of Li and Wand [7]. The instinct is that by doling out the closest style patch to each of the input patch, it chooses style insights that are progressively significant to the pasted component.

Fig 2 represents masking the required image contents

Upon embedding the object in the required position in the painting as shown Fig 1 we mask all the parts of the image that have nothing to do with it when we compute our loss as shown in Fig 2 The content loss can be computed as the mean-squared error between the masked features of our content image and the masked features of our input. Followed by we compute style loss using Gram matrices, but the problem of the mask is that it might hide some useful characteristics regarding the style that is the reason why before applying the mask to the style features, we will make some kind of mapping to reorganize them.[13]

Mapping for each layer of results computed from the model, look at each 3 by 3 patch of the content features and find the 3 by 3 patch in the style features which are nearer to it, and match them.

Once that mapping is done, transform the style features so that the centres of each 3 by 3 patch in the content features is aligned with its match in the style features. Then apply the resized mask on the input features and the style features, compute the Gram matrices of both and then take the mean-squared error to give the style loss.

We call this system independent mapping because the task is computed independently for each layer.

2.2 Second Pass Harmonization


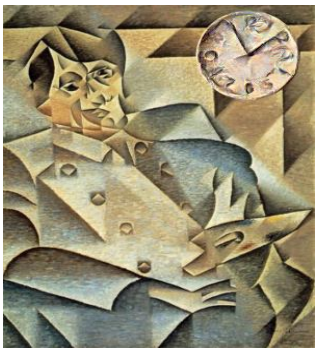

Fig 3 examples of low visual quality

From Fig 3 we can infer that the results generated from the first pass matched the desired style but does not satisfy the following two criteria: first the higher importance characteristics are not present and the other is to focus on the visual quality.

To remedy to those two things, we perform a second pass to refine the first result. This time, the **mapping** between the content and the style is with respective toa reference layer first(a single layer mapping), and the results will are reflected to the others in such a way that they refine it by trying to ensure that adjacent vectors in the style features remain adjacent through the mapping which also produces a higher quality output.[13]

3. IMPLEMENTATION DETAILS

We implemented GUI using **Kivy** which is a free and open source Python library used for developing versatile applications and other multitouch application programming with a characteristic UI that gives a broad info backing to mouse, console, TUIO, and OS-explicit multitouch occasions and **GNU Octave** primarily intended for numerical computations which provides a very powerful environment for processing and analysing images. All the techniques described in the paper are implemented in Python.

We employed a pre-trained CNN Architecture VGG-19.For the first pass harmonization we considered conv4_2 as the content extractor ($\alpha_l = 10$ for this layer and $\alpha_l = 0$ for all other layers), and conv3_1, conv4_1 and conv5_1 as the style extractor ($\beta_l = 1$ for those layers and $\beta_l = 0$ for all other layers) and for the second pass harmonization we considered conv4_2 as the content extractor ,conv1_1, conv2_1, conv3_1 and conv4_1 as the style extractor. We also implemented the histogram loss and for that used conv1_1, and conv4_1 as the histogram representation. We chose conv4_1 as the reference layer for the nearest-neighbour search in the second pass.

At the end, those four losses are summed with some weights to give the final loss of the second stage:

$$\mathcal{L} = \mathcal{L}_c + \omega_s \mathcal{L}_s + \omega_h \mathcal{L}_{hist} + \omega_{tv} \mathcal{L}_{tv} \quad (6)$$

We name τ as the output floating value and the set parameters-$\omega_s = \tau$, $\omega_h = \tau$,$\omega_{tv} = \tau * sigmoid(median\_tv(S))$,where $sigmoid(x) = 10/(1 + \exp(10000x - 25))$ and median_tv(S) is the median total variation of painting S

4. RESULTS

(a) cut and paste (b) Indpendent Mapping -1st phase (c) Consistent Mapping – 2nd Phase

Fig 4 Represents few of our results

Study 1 Comparision we showed the users 10 paintings,each of them have been edited using the four algorithms and we then asked them to pick the result which best captured the context and the style and the quantitative results are tabulated. From Table 1 we can infer that our algorithm is most prefered over others.

Table 1 represents the results of "Comparision" user study

Algorithm used	Number of votes per image									
	1	2	3	4	5	6	7	8	9	10
ours	50	40	75	30	33	40	40	40	35	37
CNNMRF	5	9	0	15	13	3	10	7	10	4
Deep Analogy	10	6	0	10	7	15	3	3	3	15
Multiscale Harmonization	0	0	0	2	3	0	0	3	5	0

5. CONCLUSION

We have introduced a flavour of python technology to embed an element of an image to another related painting. We presented a two-pass algorithm which first transfers the general style and context of the painting on the embedded object so that

the composite still looks like a genuine painting and to achieve consistency among the neural layers we map the neural responses of a dedicated layer to all the other layers. To adapt to various work of art styles, we considered a pre-trained architecture of CNN namely VGG to alter the hyper-parameters as a function of the style and content of the style painting.

We believe that our work opens additional opportunities for inventively altering and joining pictures and expectation that it will inspire craftsmen. Different roads of future work incorporate fast feed-forward system approximations of our specified structure as an augmentation to painterly video compositing.

6. ACKNOWLEDGMENT

The project team members would like to express our thanks to our guide Dr. R. Sivaranjani Head of Dept. of Computer Science and Engineering, ANITS for her valuable suggestions and guidance in completing our research work. We also thank the other teaching staff in the department for their great efforts and for giving us the advice that has helped us to achieve our project's goals.

REFERENCES

[1] Gatys L. A., Ecker A. S., Bethge M.: "Image style transfer using convolutional neural networks." In the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2016).

[2] Gatys L. A., Ecker A. S., Bethge M., Hertzmann A., Shechtman E.: "Controlling perceptual factors in neural style transfer.", In the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017).

[3] Huang X., Belongie S.: "Arbitrary style transfer in real-time with adaptive instance normalization.", In the IEEE International Conference on Computer Vision (ICCV) (Oct 2017).

[4] Johnson J., Alahi A., Fei-Fei L.: "Perceptual losses for real-time style transfer and super-resolution." In Proceedings of European Conference on Computer Vision (ECCV) (2016), Springer, pp. 694–711.

[5] Li Y., Fang C., Yang J., Wang Z., Lu X., Yang M.-H.: "Universal style transfer via feature transforms. In Advances in Neural Information Processing Systems", in the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), **2017**.

[6] Luan F., Paris S., Shechtman E., Bala K.: "Deep photo style transfer.", In the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017).

[7] Li C., Wand M.:" Combining Markov Random Fields and Convolutional Neural Networks for image synthesis." In the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2016).

[8] Liao J., Yao Y., Yuan L., Hua G., Kang S. B.: "Visual attribute transfer through deep image analogy.", ACM Trans. Graph. 36,4 (2017).

[9] Mahendran A., Vedaldi A.: "Understanding deep image representations by inverting them." In the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015), pp. 5188–5196.

[10] Simonyan K., Zisserman A.:" Very deep convolutional networks for large-scale image recognition. "arXiv preprintarXiv:1409.1556 [cs.CV] 4 Sep 2014.

[11] Tsai Y.-H., Shen X., Lin Z., Sunkavalli K., Lu X., Yang M.-H.: "Deep image harmonization.", In the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017).

[12] Wilmot P., Risser E., Barnes C.:" Stable and controllable neural texture synthesis and style transfer using histogram losses." arXiv:1701.08893 [cs.GR] 1 Feb 2017.

[13] Sylvain Gugger:" Deep Painterly Harmonization" [online Article]https://sgugger.github.io/deep-painterly-harmonization.html

[14] Matthew Stewart:" Neural Style Transfer and Visualization of Convolutional Networks" [online article] https://towardsdatascience.com/neural-style-transfer-and-visualization-of-convolutional-networks-7362f6cf4b9b