Self-Photo Mosaic

Prithvishankar Srinivasan
Carnegie Mellon University
prithvis@andrew.cmu.edu

Akanksha Periwal
Carnegie Mellon University
aperiwal@andrew.cmu.edu

Abstract

Photo Mosaic is a popular way of representing the content of large images using equally titled smaller images. The mosaics are generated by matching the tile to the image in the library with the lowest loss. Hence, traditional photo mosaics rely upon a large library with various color schemes and backgrounds in order to effectively represent any given image. In this project, we tried to utilize the information present in the larger picture itself to represent various features in itself. This is done using by using a pre-trained network to extract the content and style of the network separately[1] and combine them in a well-defined ratio in order to retain the content of the style image as well as the content image.

1. Introduction

Photomosaic is a term used to describe images where tiles of the image have been replaced by small library images, which when processed and composed together give the appearance of a larger target image. The concept of stitching together a variety of images to produce a larger image when viewed at lower magnifications has been in existence since the early 90s. Since then, there have primarily been two forms in which photomosaics have been composed. One is to reduce each tile to a single colour and match it to the image in the library which has the same dominant colour. The other method involves taking every pixel in the tile and comparing it to the pixels in the set of library images. The overall loss is reduced in order to fit the best image. [2] The only factors limiting the performance of a photo mosaic is the absence of certain colors in the library images and the time taken to iterate through all the images in the library. Although, work has been done to address the time constraint [3], no work has been done to combat the former problem.

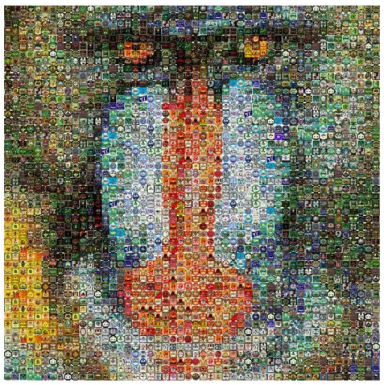

Fig 1: Photo mosaic generated by fast photo mosaic [3]

1.1. Neural Style transfer

Neural Style transfer [2] is a technique used to extract and consequently merge the content and style characteristics of two arbitrary images. The generated output image has style elements of one of the input images, while retaining the overall structure of the other.

As observed in the introductory work on neural style transfer [1], in a convolutional neural network trained for object recognition, as you go progressively deeper into the network layers, the feature maps obtained become more representative of the high level appearance of the image rather than low level pixel details. Thus, these outputs, when used to reconstruct an image, would result in a picture depicting the content of the original image fed into the network without retaining the lower level pixel intensity information. On the other hand, correlations between the feature responses in each layer of the network helps in understanding the texture information present in the image. Thus, if image reconstruction using this textural information from various layers would result in an output which captures the colours and texture of the input image, without too much emphasis on the overall positioning of global structures.

While these two aspects of an image are not completely independent of the other, the recombination of the two images aims at simultaneously minimizing the losses in both. The total loss function is a weighted combination of the style loss and the content loss. This weight ratio determines if the output image is more representative of the characteristics of the style image or that of the content image.

1.2. Self Mosaics

The idea behind a self mosaic is to replace the hundreds and thousands of library images, that are tiled up to create the target image, with resized versions of the target image itself. This requires the input image to be split into a grid of sub-images, each of which need to be replaced with a stylized version of the original image itself. These replaced patches need to resemble the colour pattern and texture (style) of the sub-image, but also need to reflect the structure (content) of the original target image. So, using the concept of style transfer, the style of the sub-image is merged with the content of the original image to create the output tiles. These tiles are then placed at their respective positions in the mosaic and then stitched together to generate the final output.

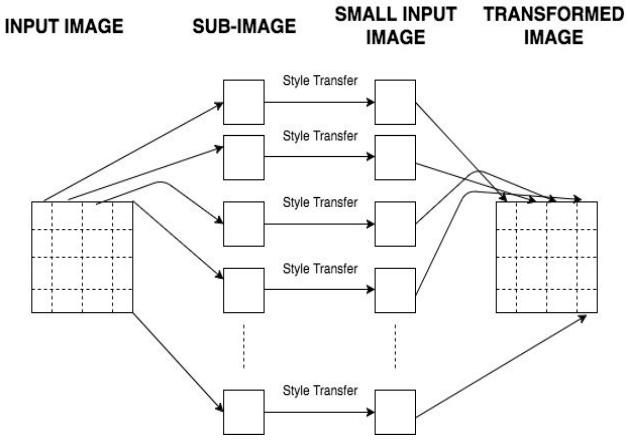

Fig 2: Flow diagram of subimages

2. Experiment

A pre-trained VGG-19 network, trained for common visual object recognition, was used as the network for the style transfer. After the initial tiling of the input image, each content-style image pair is passed through the feed forward VGG-19 network. An initial white noise image is then passed through the same network and gradient descent on the total loss function is used to modify the input pixels to the network till the outputs produced at selected network layers closely resemble those of the content and style images.

Since the deeper layers of the network are more representative of global structures, the 'relu4_2' layer was chosen to compute the content loss. The feature outputs of the relu1_1', 'relu2_1', 'relu3_1', 'relu4_1', 'relu5_1' were used for the purpose of calculating the style loss.

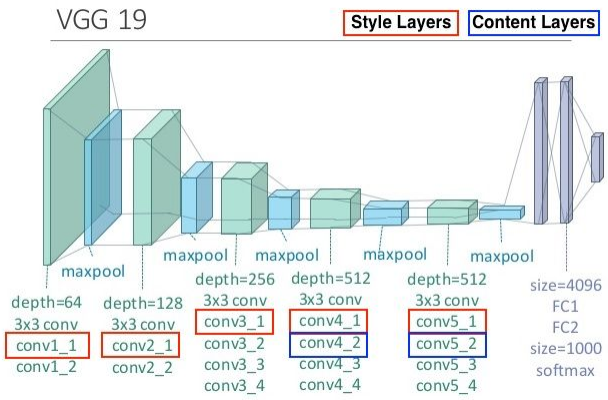

Fig 3: Architecture of VGG-19 network [4]

Content loss is given by the mean squared error between the layer 4 filter responses of the content image and the other initial white noise image being fed into the network. When the content is matched on a lower layer of the network the individual pixels of the content are still retained and it barely looks like a blend of the style over the content. While matching it with deeper layers makes the final image look like a proper transfer of style of the style image over to the content image. P and F represent features at layer 4 for original and generated images.

$$L_{content} = \frac{1}{2}\sum_{i,j}(F^4_{i,j} - P^4_{i,j})^2$$

The style loss on the other hand is obtained by generating the correlation between the different filter responses using a Gram Matrix [6]. To obtain an acute representation of the style, the pixels of the input image then needs to be modified so as to minimise the mean-squared distance between the two Gram Matrices. A and G represent the gram matrices of the original and generated image and N,M are the size of these maps.

$$L_{style} = \frac{1}{5}\sum_{l=1}^{5}\frac{1}{4N_l^2M_l^2}\sum_{i,j}(G^4_{i,j} - A^4_{i,j})^2$$

In our implementation, it was important for the tiled patches in the final mosaic to present a coherent appearance of the input image. Thus, the style loss weight in the loss calculation was an order of magnitude larger than the the content loss weight. Also, the different layers are weighted equally in contributing to the final style loss.

3. Results

The algorithm was run for a variety of images. The results shown below are for 1024x1024 images. After observing and experimenting with various intermediate sub-image sizes and results, the algorithm for each sub-image was run for 500 iterations with 400 nearly equal sized patches.

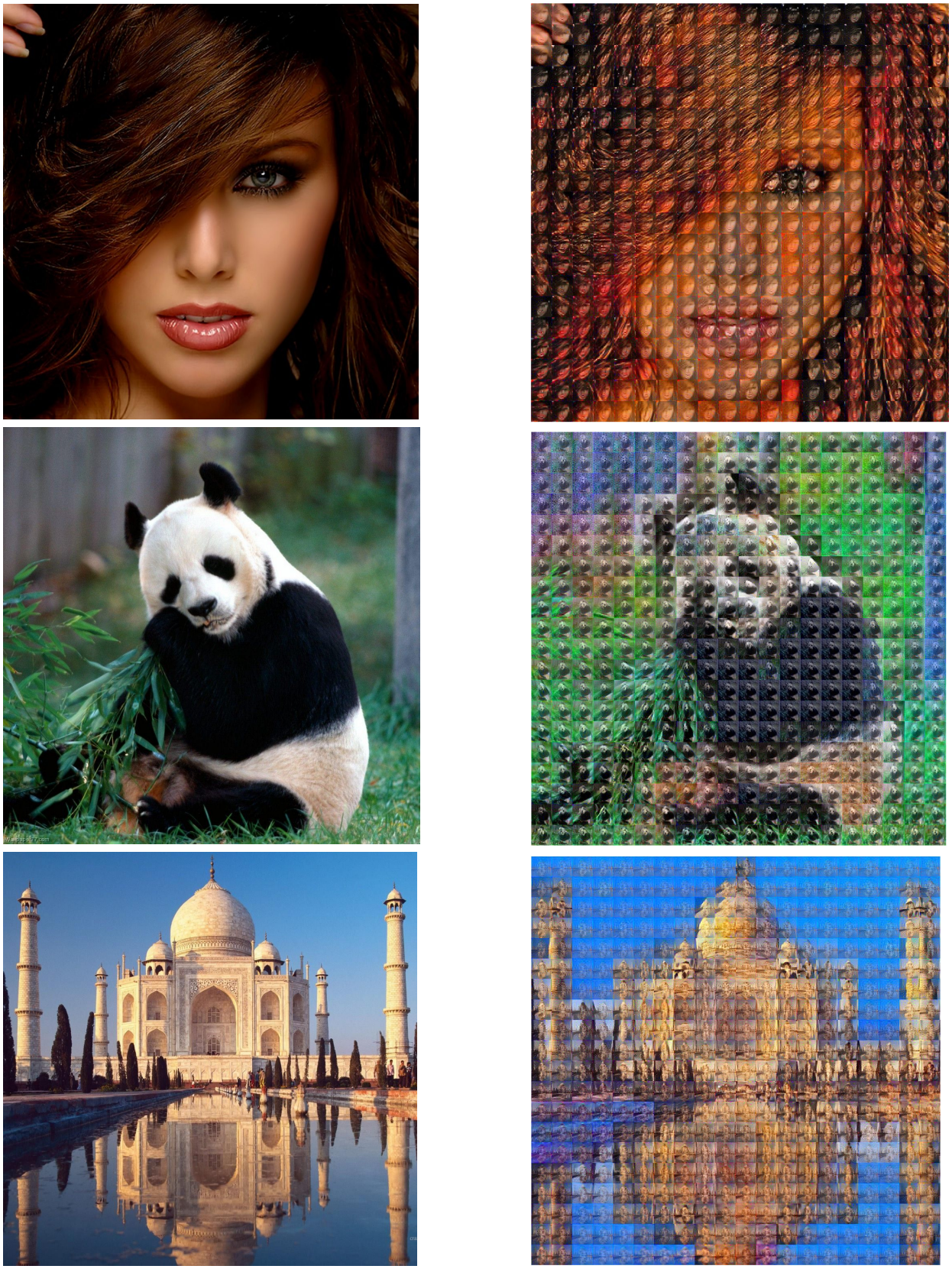

Fig 4. Input images and output self mosaics

4. Analysis

Figure 5 shows a more detailed view of the style transfer results of different patches using the same master image. The output sub-images show that the input features are captured quite effectively, while retaining the overview of the original input image.

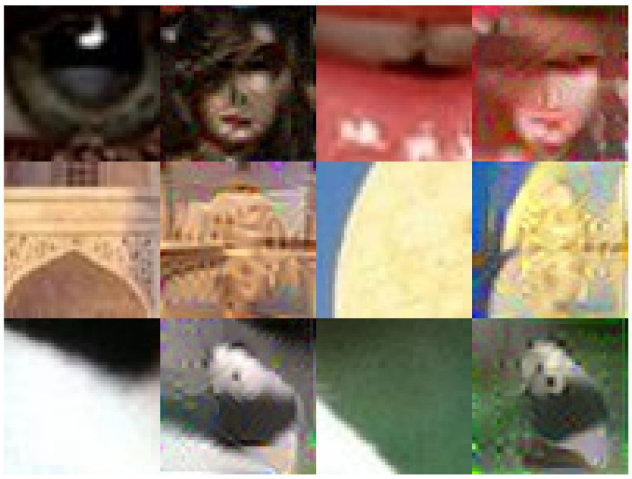

Fig 5. Generated images for certain patches

The mosaic creation is best when there are a limited number of colours and textures in the sub-images. If the sub-images are overly detailed, it is difficult to capture all the information as a part of the image style. Also, as can be seen in figure 4, the hand of the panda has a bluish hue instead of the original black. This is also noticed in other parts of the image (the out of focus background areas), where due to the lack of any textural features or colour gradients in the entire sub-patch, the style features of the image are harder to discern and thus would take many more iterations to give a more accurate representation.

5. Conclusion & Future work

This self-photo mosaic creation works best when the images have a limited set of colours and textures. Overly-detailed sub-patches are difficult to capture as the style image has too many elements in those cases.

The algorithm however can be improved for better clarity and performance. While the content sub-images in this case are not intended to have the clarity of the ones in a traditional photomosaic, some work can still be done to improve their appearance by tuning the parameters and algorithm for the same.

Also, the average runtime of the algorithm was pretty high. Since the content image remains the same over the patches, this may provide a means of minimizing the computation required for each of the sub-images and can be optimized for better performance.

References

[1] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image style transfer using convolutional neural networks,” in Proc. CVPR, Jun. 2016, pp. 2414–2423.

[2] https://en.wikipedia.org/wiki/Photographic_mosaic

[3] G. Di Blasi and M. Petralia, “Fast Photomosaic,” Proc. of International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, 2005.

[4] https://www.slideshare.net/ckmarkohchang/applied-deep-learning-1103-convolutional-neural-networks

[5] https://github.com/anishathalye/neural-style

[6] L. A. Gatys, A. S. Ecker, and M. Bethge. Texture Synthesis Using Convolutional Neural Networks. In Advances in Neural Information Processing Systems 28, 2015

APPENDIX A- IMPLEMENTATION

The project was coded using Python and Matlab. A detailed description is given below.

Image preprocessing: This part is executed using Matlab. the image is first resized while maintaining aspect ratio. It is then split into 400 equal sized parts. A smaller version of this image is then made to match the size of the parts which serves as the content image for the style transfers.

Neural style transfer: This part of the code was implemented using Python [5]. PIL library is used to handle image functions. Tensorflow library was used to handle the pre-trained VGG network and compute style, content losses.

Image postprocessing: This part is executed using Matlab. The transformed subimages are placed into the correct positions to create the mosaic of the original image.

Individual patches with high variance were first tested. Then the hyper parameters were tuned based on visual appeal, since there is no method to quantify the results. Content weight had to be relatively lowered compared to style weight.

After the parameters were tuned, a small section of the original image was divided into patches and tested. After considerable results were achieved, the algorithm was run on the images to achieve the entire result.

APPENDIX B- DIVISION OF WORK

Prithvishankar Srinivasan
50% Literature Survey
70% Neural style transfer implementation
30% Hyper parameter tuning
100% Image postprocessing
70% Project poster work
40% Final report work

Akanksha Periwal
50% Literature Survey
100% Image preprocessing
30% Neural style transfer implementation
70% Hyper parameter tuning
30% Project poster work
60% Final report work