COMP6248 REPRODUCABILITY REPORT: A LEARNED REPRESENTATION FOR ARTISTIC STYLE

Thomas Darlison, Adam Melvin & Lawrence Gray
Team: Dicephalous Lampadephore
Electronics and Computer Science
University of Southampton

ABSTRACT

Style transfer is the transfer of the high level features of one image to another, often used to apply the artistic styles of pieces of art to normal images. This report aims to reproduce the findings of Dumoulin et al. (2016) – that multiple styles can be represented with a single neural network, that said network can have new styles added to it post training, and finally that this network can be used to arbitrarily mix different styles onto a single image. The report shows that all of these results are reproducable, however is not successful in reproducing them to the same level of quality as the original paper.

1 INTRODUCTION

Transfer of artistic style is a well studied area of both computer vision and machine learning, with newer implementations such as Babaeizadeh & Ghiasi (2018) even going as far as adjustable real-time style transfer. Dumoulin et al. (2016), the paper being reproduced in this report, contributed efficient multi-style networks to the field. This was an important development, as it allowed many styles to be trained into a single network with only small increases in the number of weights. This is achieved through the introduction of conditional instance normalisation (CIN) blocks – dependant on the style being transferred, different weights inside the CIN blocks are used, changing the behaviour without requiring changes in the rest of the network.

Further to simply training multi-style networks all at once, Dumoulin et al. (2016) also showed that new styles can be easily added to pretrained networks through training a new set of parameters within the CIN blocks without modifying the rest of the network. This was shown to take significantly fewer parameter updates to train than when training a network from scratch. The paper also showed that CIN blocks also allow for styles to be mixed to ratios set by the user without further training. This report aims to reproduce the findings that multi-style networks can effectively produce multiple styles, and also reproduce these further observed behaviours.

2 IMPLEMENTATION

The building of this network was completed by following the network description found in the appendix of Dumoulin et al. (2016). This describes the 16 convolution blocks that make up the network and all of the hyperparameters required to initialise the network. The network specifies that a CIN block follows every convolution, these blocks being the part of the network that enables the learning of multiple styles.

The initial idea of this project was to implement Dumoulin et al. (2016) using the torchbearer library but this was rejected when it came to light that it did not properly support the required loss function. This was due to the fact that the loss function required the content image, the style image and the generated pastiche to be processed through a forward pass of a VGG-16 model which had been modified to extract the outputs from specific layers. The outputs of the VGG-16 network are passed through a gram matrix (code obtained from pytorch[1]), then a weighted sum of the MSE loss is

[1] `pytorch.org/tutorials/advanced/neural_style_tutorial.html`

calculated. This is done using the content at the fourth convolution inside the VGG and the style at first, second, third and fourth convolutions. Because of these issues, a direct pytorch implementation was chosen to allow easy access into the layers of the VGG and along with writing a custom loss function.

CINs do not exist as part of pyTorch, therefore a custom implementation was required. A CIN is effectively a list of instance norms where an instance norm exists for each of the different styles being learned. When in training, the CIN must return the output of the instance norm for the style being trained. In inference, the CIN is able to return the output of an instance norm which is created from a mix of parameters from multiple instance norms (styles). To achieve this the instance norms from pyTorch had to be modified to be able to read and write parameters.

Adding styles to a pre-trained network was another required feature of this network, but simply instantiating the network to support more styles before loading the weights is not possible. The network supports the ability to append new instance norms to every CIN and initialise their weights with a Gaussian distribution. After freezing convolution blocks in the network a new style can be learned into the appended instance norms.

Although the paper suggests that its network can scale to process multiple images at a time, it was decided not to implement it in this way as, only supporting image at a time simplified being able to handle SGD and using the network for both training and inference. This decision was also made due to limitations in GPU memory that meant only one image could be trained on at a time.

3 Experimental Methodology

To train the network, a dataset was formed from the validation set from ImageNet's 2012 Challenge training data set[2]. This was chosen as it contained 50,000 images, and was split into 48,000 for training, 1,000 for validation and 1,000 for testing. Although a test data set was formed, a small subset of known set of images were chosen to test the network instead. This made comparison between runs easier to aid our training. One of these images was the same used to present the network in the paper, also allowing a direct comparison with their results.

To train the network, Google Colab[3] and also a personal machine were utilised. The personal machine was able to train faster than Colab but had restrictions in GPU memory as was limited to a GTX970 with 4GB RAM. For note, the other key components included an i7-4770k at 3.5GHz and 16GB GDR3 RAM. This restriction on GPU memory is why batches were not used within testing – to allow the model to fit onto the GPU. Although the code supports batches and could be utilised with batches on Colab, a choice was made to keep the code consistent across platforms.

The input image sizes were kept consistent with the paper at 256 pixels square for training and 512 for testing. While testing, the methodology of 40,000 parameter updates was kept consistent with the paper resulting in 16 images fed through the network 2,500 times.

To tune the model, each style image had an associated lambda value corresponding to a style-content ratio. This allowed the network to blend together the correct proportions of style and content within an image to produce a good representation. Upon multiple iterations, these could be changed and the outputs compared to see what lambda generated the best visual output and lowest loss during training.

To add styles to the pretrained network, the original network was frozen. This prevented the rest of the network from training further. When the new style image was added, the CIN modules added were the only layers able to be trained. To train, 5600 parameter updates were done for each new style, resulting in 350 training iterations for the addition of one style.

[2] `www.image-net.org`

[3] `colab.research.google.com`

4 RESULTS AND DISCUSSION

Attempting to train the multi-style network immediately presented an issue with the paper being replicated as, within the paper, they did not provide adequate information on the actual training methodology used – the training methodology was only explained in the open review comments[4].

The network recreated in this report is the first network Dumoulin et al. (2016) present – a 10-style Monet network. This network was able to achieve reasonable style transfer, including both colours and painting technique (Figure 1). When comparing the images produced in the original paper to our reproduced images, there were some clear differences, however. This became apparent where our trained network appeared to focus on small repeating patterns, compared to the much boarder strokes apparent in the paper. Another notable comparison is that out network tends to tint the whole image to the average colour of the style. This does not match the paper which can achieve large separate blocks of colour which match the style well.

Given that training took roughly two hours on colab, it was a very slow and difficult task to be able to tune the hyper-parameters for all ten images to produce a good result, and thus these differences could be a case of poor tuning. Furthermore, as the paper did not determine exactly the training method used to produce the results given, there is a possibility that a different training methodology was used – for example, in the open review comment explaining their training method, mini-batch was suggested as another possibility.

Figure 1: Output of trained network. Figure shows two images with five of the ten different styles applied. The top content image is taken from the original paper and the bottom content image from the ImageNet 2012 challenge dataset.

When adding styles to a pretrained network, the original paper states that 5000 parameter updates is sufficient to get a comparable output. This was tested with our network by adding in two new styles: Roy Lichtenstein's *Bicentennial Print* and Juan Gris' *Portrait of Pablo Picasso* (Figure 2). As suggested, adding styles proved to be a much quicker process than, and produced comparable results to, training the styles from scratch. One notable difference is that when the Lichtenstein is added to the pretrained network, the output images never manage to achieve as white a background as when it is trained into a network from scratch. Instead, a yellow-grey average colour is produced. Furthermore, the minimum loss value for the new style at the end of training is above the average loss for training the original network. This shows that the network may not be capable of fully learning the Lichtenstein, potentially due to the way the network was tuned for the Monet styles. Given that the original paper's experimentation in this area was adding Monet styles to a more complex network of various artists, producing better results, it is possible that more complex styles

[4] openreview.net/forum?id=BJO-BuT1g&noteId=HyD4_Yw7l

like the Lichtenstein cannot be effectively added to a network only trained on less complex styles, thus explaining these results.

Figure 2: Output of one of the original styles and the newly added styles after 5600 parameter updates of training each.

When mixing styles, the reproduced network was able to successfully arbitrarily mix any number of styles that the network had been trained on (Figure 3). It was found that the quality of the mixing varied wildly between different training attempts and depending on the choice of styles used. The issue primarily came when mixing styles with differing colour palettes, as it could sometimes go though some brighter (unexpected) colours when trying to mix the styles in equal quantities. Dumoulin et al. (2016) produce much smoother transitions without this effect, though this could be due to the general style transfer results achieved by the paper already being higher quality.

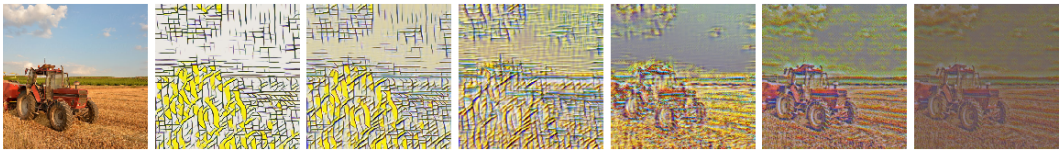

Figure 3: Style mixing a picture of a tractor between Roy Lichtenstein's *Bicentennial Print* and Claude Monet's *Grainstacks at Giverny; the Evening Sun*. The first image is the original image which the styles are being applied to.

5 Conclusion

When presented with the question of the reproducability of Dumoulin et al. (2016), the results achieved for this report are mostly positive. The network created can successfully transfer multiple styles, learn new styles in significantly fewer parameter updates than when training from scratch, and mix multiple styles. The overall quality of the style transfer was lower than intended, however it is likely that this is a case of insufficient tuning or slightly different training methodology. Improvements could therefore be made to these results given more time tuning or exploration of different training methodologies such as using mini-batches.

References

Mohammad Babaeizadeh and Golnaz Ghiasi. Adjustable real-time style transfer. *arXiv preprint arXiv:1811.08560*, 2018.

Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. 2016.