Computationally Efficient Approaches for Image Style Transfer

Ram Krishna Pandey[1], Samarjit Karmakar[2], A. G. Ramakrishnan[1]

[1]Indian Institute of Science, Bangalore
[2]National Institute of Technology, Warangal

December 17, 2018

Overview

1. Introduction
   - What is Artistic Style Tranfer?
   - Related Work
   - Fast style transfer
2. A Computationally Efficient Approach
   - Our contributions
   - Why depthwise separable convolutions?
   - Proposed architectures
   - Experimentation
3. Experimental Results
4. Conclusion

What is Artistic Style Transfer?

- In Artistic Style Transfer, we superimpose the texture or style of an artistic work (image) onto the content of a natural scene image.
- The task of artistic style transfer is interesting in the sense that the representations of the content and the style in convolutional neural network are separable.

An example






Original photo | Reference photo | Result

How do we get a separable representation?

- Higher level feature information (content) are captured more in deeper layers.
- The content of an image are the filter responses at the deeper layers of a VGG network trained for image classification task.

What is style of an image?

- The style of an image is the linear combination of the Gram matrices of the feature maps taken at different layers of the same network. Gram matrix captures the correlation of the feature maps at a layer.
- Let the feature maps of the I^{th} layer of the network be $F_I(S)$, where S is the style image. Then the Gram-based representation of this layer is given as:

$$G(F_I(S)) = [F_I(S)][F_I(S]^T \tag{1}$$

What is style of an image? (contd.)

- The weighted combination of the Gram-based representations of multiple layers of the network is considered for a representation of the style of an image. Assuming we take Gram-based representations for the layers $l = 1, 2, ...n$, the style representation (Sty) for the style image S can be given as,

$$Sty = \sum_{l=1}^{n} w_l \times G(F_l(S)) \tag{2}$$

- The style loss is given by

$$L_s = \sum_{l=1}^{n} w_l \times ||G(F_l(Y)) - G(F_l(S))||_F \tag{3}$$

Here, w_l is the weight assigned to the l^{th} layer, Y is the desired image and S is the style image.

What is content of an image?

- The content representation is simply the feature map of the k^{th} layer of the network. The deeper layers of the network are considered, since they capture more of the higher-level features of the image (which represent the content), than the shallow layers.
- The weighted combination of the content representations of multiple layers of the network is taken as the representation of the content of an image. Assuming we include content representations for the layers $k = 1, 2, ...n$, the content representation (Con) for the content image N is given by,

$$Con = \sum_{k=1}^{n} w_k \times F_k(N) \tag{4}$$

Here, w_k is the weight assigned to the k^{th} layer, Y is the desired image and N is the content image.

What is content of an image? (contd.)

- The content loss can be given as,

$$L_c = \sum_{k=1}^{n} w_k \times ||F_k(Y) - F_k(N)||_F \tag{5}$$

Objective function

- The objective is to find $\hat{Y}$, which satisfies the following:

$$\hat{Y} = \underset{Y}{\operatorname{argmin}}(\alpha L_c + \beta L_s) \tag{6}$$

where L_c and L_s are the content and style losses, respectively. This is a slow parametric method to obtain the desired image Y, since we need to iteratively optimize the objective function for each $\{N, S\}$ pair.

Fast style transfer

- We train an image transformation network to find a non-linear function that maps the content image to the desired output image for a given style image, on which the network has already been trained.
- We use a loss network pre-trained for image classification task to define perceptual loss functions that measure perceptual differences in content and style between the images. The loss network remains fixed during the training process.

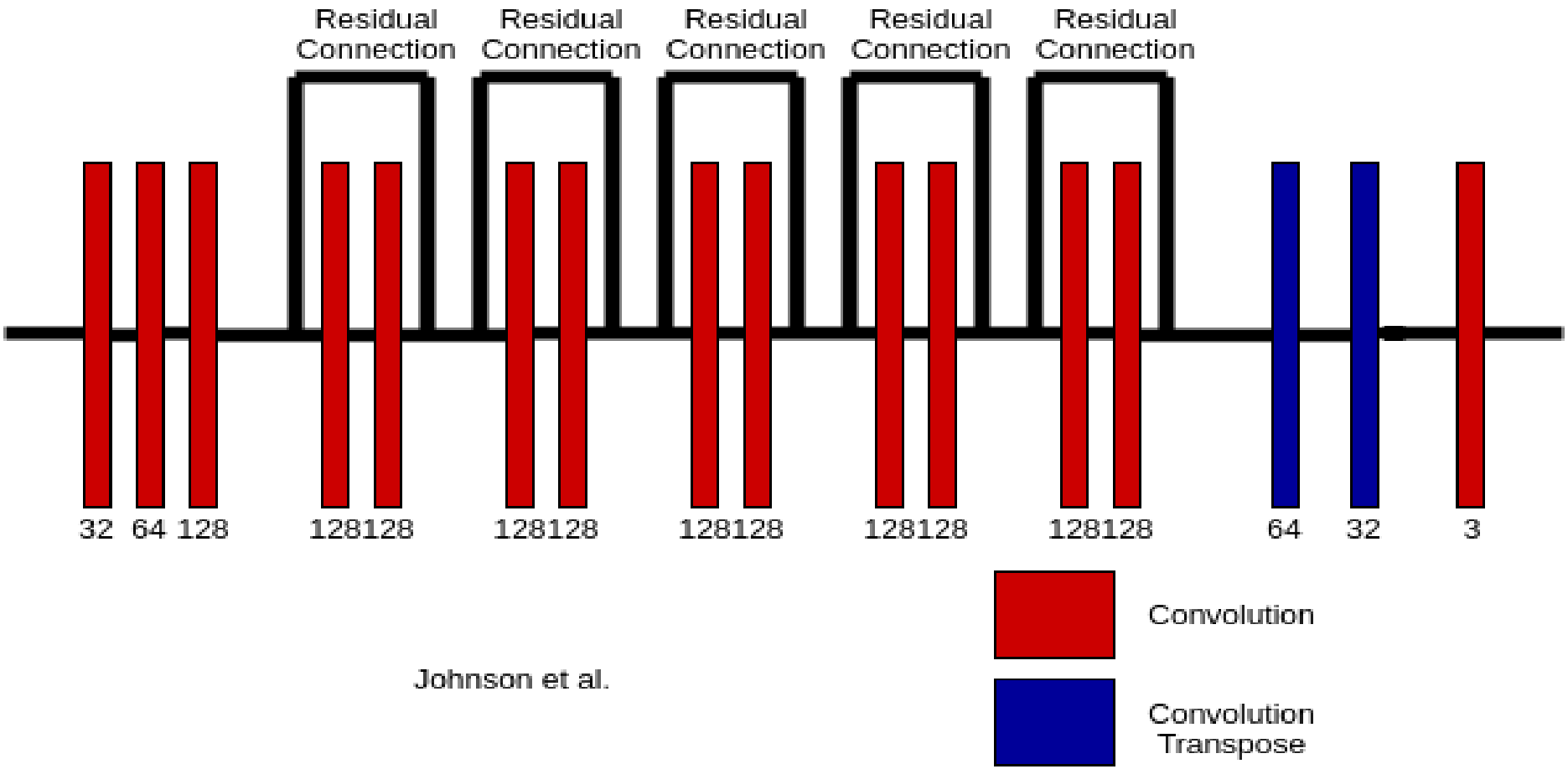
Fast style transfer (contd.)

- The image transformation network is a deep residual convolutional neural network with parameters λ.
- The loss network is a pre-trained VGG network on image classification task.
- The total loss, expressed as,

$$L_{total} = \alpha L_c + \beta L_s + \gamma L_{tv} \tag{7}$$

is minimised by backpropagating using stochastic gradient descent optimizer.

Johnson et al



Johnson et al.

Our contributions

- We have shown that using Charbonnier as the content loss function results in the preservation of more content in the output image.
- We have improved the computational efficiency of the method proposed by Johnson et al. by using depth-wise separable convolution layers instead of regular convolution layers in the image transformation network.
- We have replaced the transposed convolution layer with
  - Nearest neighbour upsampling with gradient flow
  - Concatenation of nearest neighbour and bilinear up-sampling

Why depthwise separable convolutions?

- Our main objective is parameter reduction.
- In Xception model, Francois Chollet shows the working of depthwise separable convolution in deep learning architectures. It results in a huge reduction in the number of parameters, while the model performance is similar.

DepSep architecture

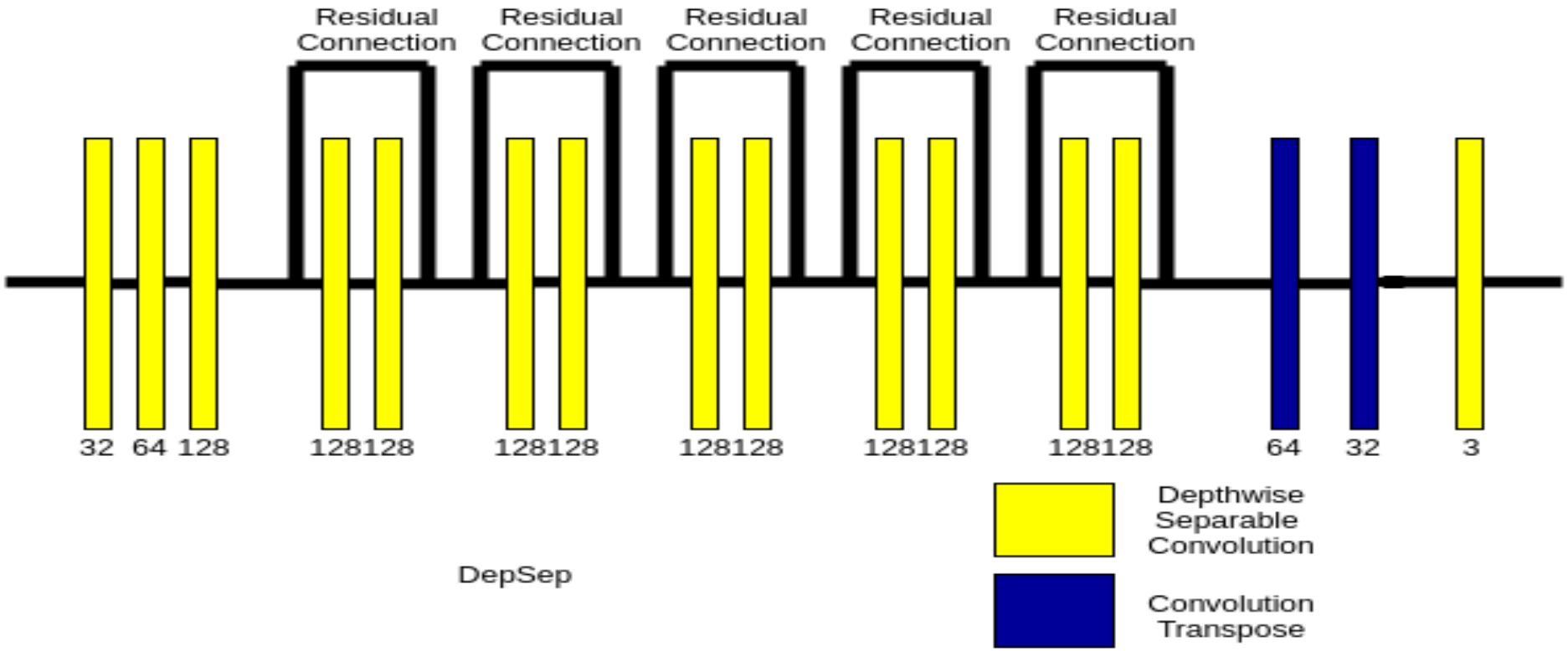


Figure: 'DepSep' architecture: an improvement over the image transformation network of Johnson et al. The convolutional layers have been replaced with depth-wise separable convolutional layers, with channel multiplier = 4.

DepSep-NN architecture

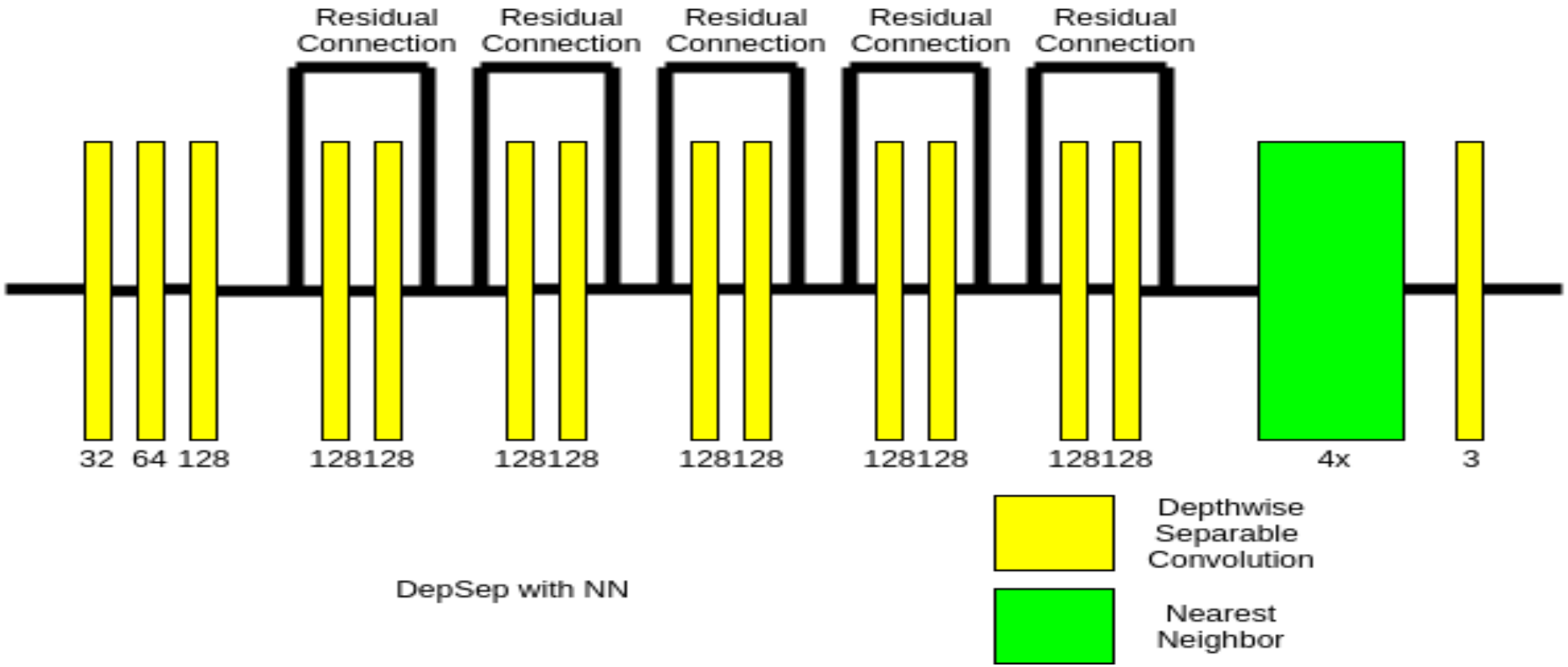


Figure: 'DepSep-NN' architecture, an improvement over DepSep, by the replacement of transposed convolutional layers with nearest neighbor up-sampling.

DepSep-NN-Bil architecture

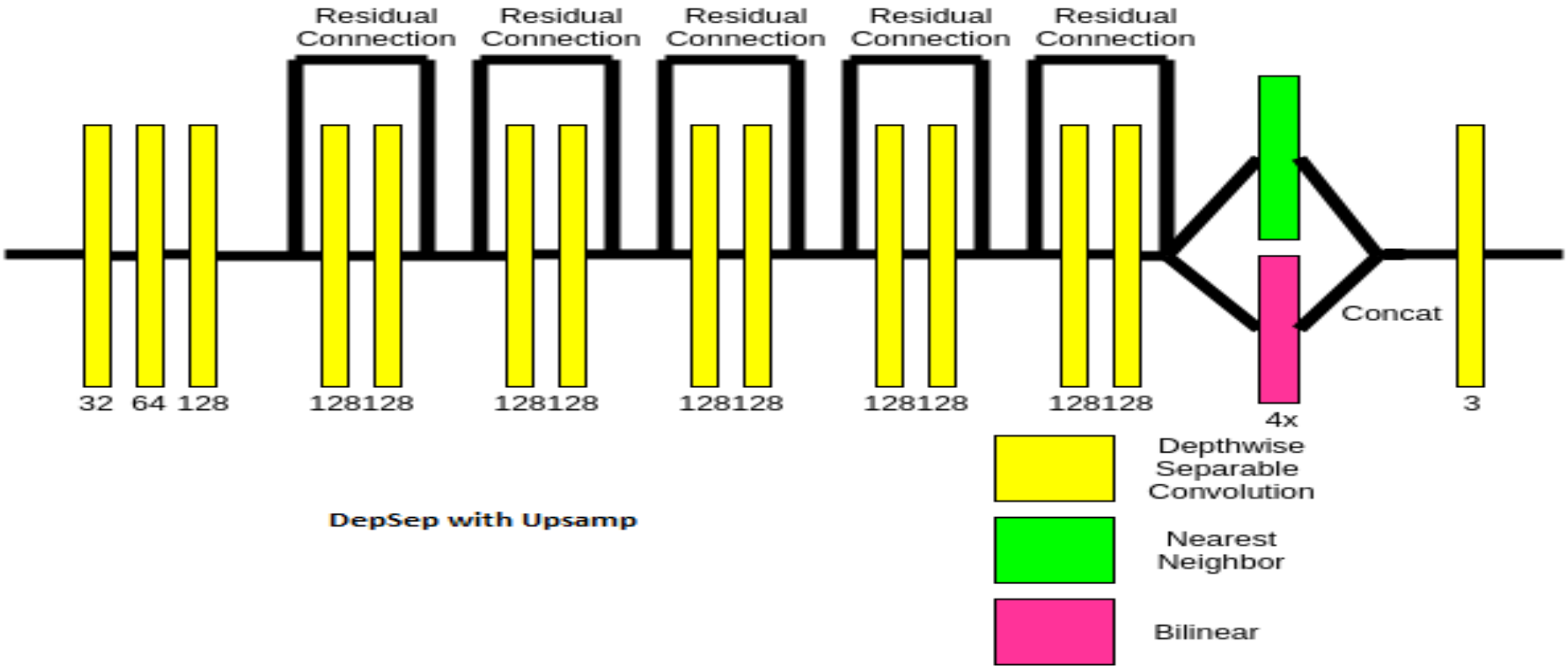


Figure: 'DepSep-NN-Bil' architecture, an improvement over DepSep, by replacing transposed convolutional layers with the concatenation of nearest neighbor and bilinear up-sampling.

Experimentation

- We have implemented the proposed architectures as above and trained the networks on the Microsoft COCO dataset.
- We have used Adam optimizer to minimise the loss, with learning rate as 0.001, β_1 as 0.999 and β_2 as 0.99.
- Instance normalization has been applied after every depthwise separable convolutional layer, as suggested by Ulyanov et al.

Content and Style image

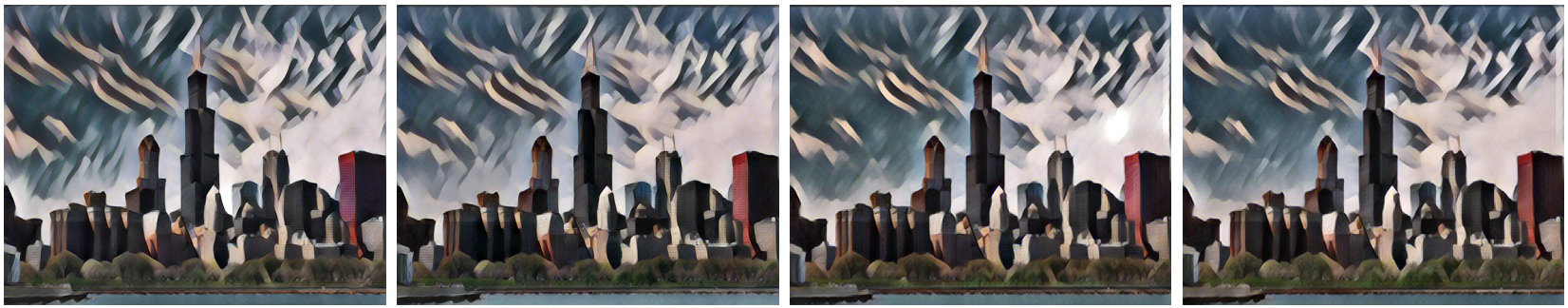
(a) Chicago skyline

(b) Udnie

(c) The Starry Night

Figure: Each model has been tuned to a specific style image. For each style image, we train a model on a dataset of images. The content image is fed into the model while testing. (a) A natural scene image - Chicago skyline. (b) "Udnie" image by Francis Picabia, 1913. (c) "The Starry Night" image by Vincent van Gogh, 1889.

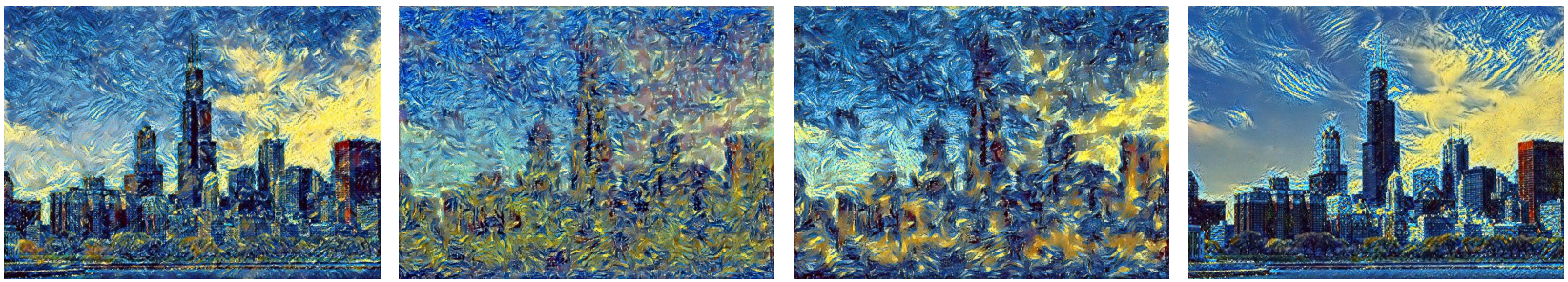
Qualitative Evaluation



(a) Johnson et al (b) DepSep (c) DepSep-NN-Bil (d) DepSep-NN

Figure: The results obtained from the various architectures proposed, for Chicago skyline as the content image and Udnie as the style image. (a) The stylized output of proposed by Johnson et al. (b) Output of our DepSep model shown in Fig. 2. (c) Output of our DepSep-NN-Bil model shown in Fig. 4. (d) Output of our DepSep-NN model shown in Fig. 3.

Qualitative Evaluation (contd.)



(a) Content - MSE Style - MSE (b) Content - Char Style - Char (c) Content - MSE Style - Char (d) Content - Char Style - MSE

Figure: A qualitative analysis of the effect of different loss functions on the stylized output of the method proposed by Gatys et al. Chicago skyline is the content image used, and Starry night, the style image. We have varied the content and style losses with MSE and Charbonnier (Char) loss functions. The details are given in the subcaptions above.

Testing time

Table: Comparison of the testing time taken by the original model by Johnson et al. with those of the modified architectures proposed by us.

Architecture	**Testing time in sec.**	**% decrease in time**
Johnson et al.	1.33	-
Modified 1: DepSep	0.97	26.1
Modified 2: DepSep-NN-Bil	0.81	39.1
Modified 3: DepSep-NN	0.57	57.1

Conclusion

- We have explored various models by changing the loss function, such as the mean square, MSCE and Charbonnier in the original implementation proposed by Gatys et al. and have shown their results.
- We have further improved the already fast architecture proposed by Johnson et al. using depth-wise separable convolution, nearest neighbour interpolation and the combination of nearest neighbour and bilinear interpolation and provided three computationally efficient architectures.
- Our proposed architectures show significant improvements in testing times of 26.1%, 39.1%, and 57.1%, respectively in comparison to the original model.

Thank You