Project Progress Report : Texture Synthesis

Rohan Chandra Sachin Grover Kyungjun Lee Moustafa Meshry Ahmed Taha
{rohan,kjlee,mmeshry,ahmdtaha}@cs.umd.edu saching@umd.edu

We propose using the Deep Recurrent Attentive Writer (DRAW) network [1] for texture synthesis. We managed to train and deploy DRAW tensorflow implementation. Simple texture synthesis experiments were evaluated to verify the idea validity. Figure 1 shows a sample result for a tilted chess board texture. The generated tiles not only capture the correct texture but also aligns smoothly with the center input tile. Figure 2 shows more examples. This progress aligns with the project proposed timeline.

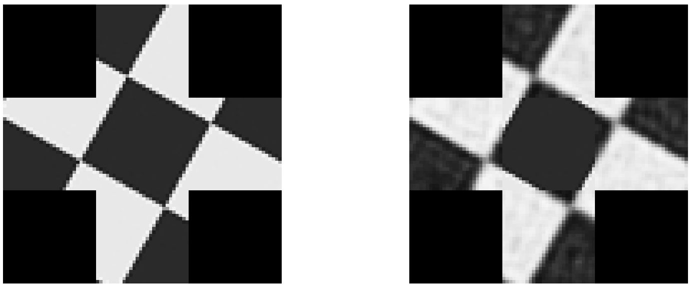


Figure 1: Left: Original texture. Right: Synthesised texture using center tile

The current network uses pixel-wise difference as the loss function. This naive loss fails for complex texture synthesis. Thus, a texture evaluation approach using filter banks is being integrated into the Tensorflow network. Varma et. al [2] well-known approach for texture description is employed due to its efficiency. Our prior knowledge with its details will save time to explore other project aspects within the same timeline.

DRAW is a variational network, so the learned probability model focuses on common input structure and discards fine details. Thats why the current texture results are blurry even for a chess-board texture. To reduce the fuzzy effect, we plan to investigate multiple approaches. Sharpening the result as a post processing step will be considered. We will explore DRAW structure and increase its complexity, so that it can better comprehend texture details. Danihelka, one of the DRAW authors, suggested we use adverserial networks like PixelCNN to eliminate this effect. If time permits, this approach will be investigated.

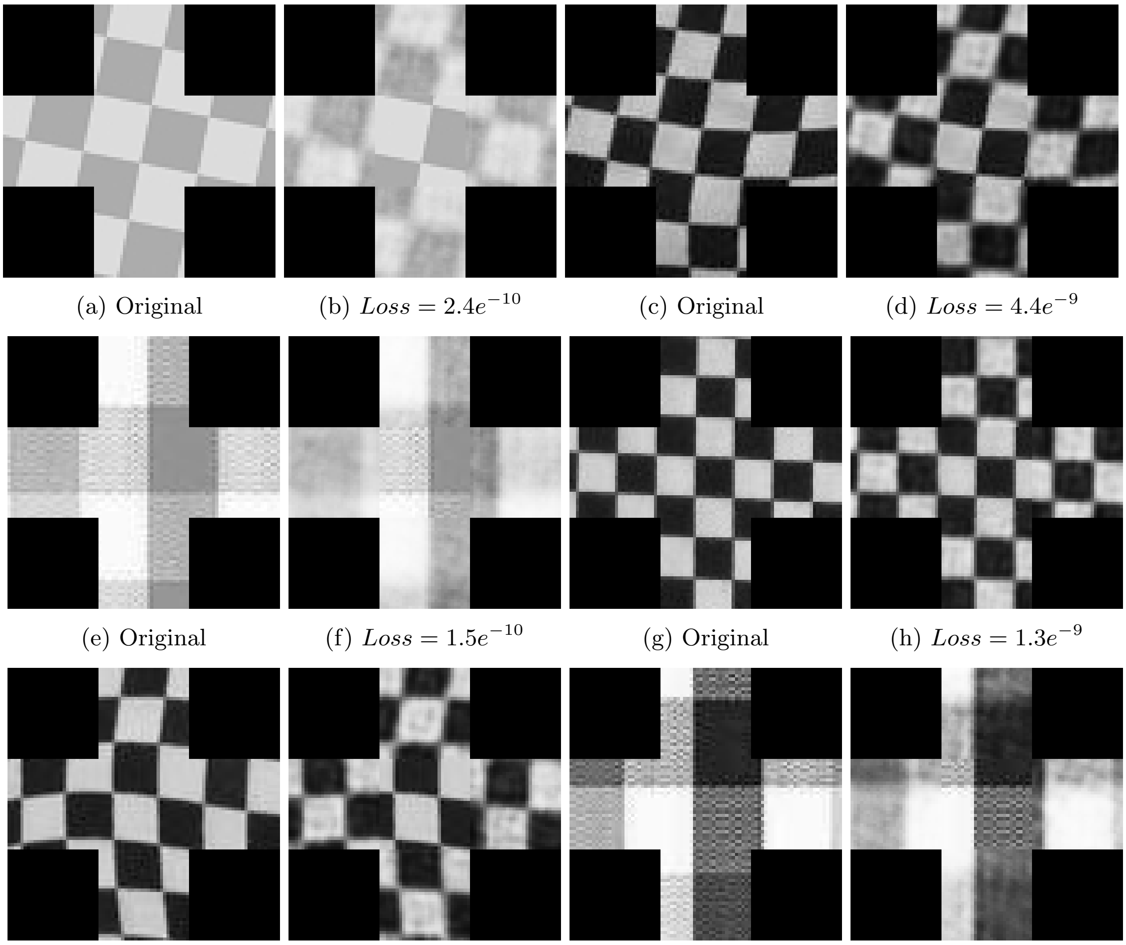
Once our network is tuned to generate complex texture with higher quality, we will quantitatively evaluate the constructed texture using the VGG network [4]. This approach is proposed by Gatys et al. [3] for texture generation and adapted later for image style transfer. Basically, feature maps generated at different VGG-network layers by similar textures have high correlation. Thus, the correlation between the feature maps of input texture x and the synthesized image $\hat{x}$ is a quantitative evaluation metric. To elaborate, x and $\hat{x}$ are fed into the VGG-network independently, each producing their own set of feature maps, $F^l \in \mathbb{R}^{N_l \mathrm{x} M_l}$ and $\hat{F}^l \in \mathbb{R}^{N_l \mathrm{x} M_l}$ where each layer l has N_l distinct feature maps each of size M_l when vectorised. F^l_{jk} is the activation of the j^{th} filter at position k in layer l. The correlation between these feature maps are stored in a matrix G and $\hat{G}$ respectively. $G^l \in \mathbb{R}^{N_l \mathrm{x} N_l}$ is defined as:

$$G^l_{ij} = \sum_k F^l_{ik} F^l_{jk}$$

The distance, between two textures, is the weighted sum of layer-wise distance.

$$D_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G^l_{ij} - \hat{G}^l_{ij})^2$$

$$\mathcal{D}^{total}(x, \hat{x}) = \sum_{l=0}^{L} w_l D_l$$



(a) Original (b) $Loss = 2.4e^{-10}$ (c) Original (d) $Loss = 4.4e^{-9}$

(e) Original (f) $Loss = 1.5e^{-10}$ (g) Original (h) $Loss = 1.3e^{-9}$

(i) Original (j) $Loss = 2.9e^{-9}$ (k) Original (l) $Loss = 1.09e^{-9}$

Figure 2: The first and third column show the original texture. The second and fourth column show the corresponding synthesised texture.

References

[1] Gregor, Karol and Danihelka, Ivo and Graves, Alex and Rezende, Danilo Jimenez and Wierstra, Daan *DRAW: A recurrent neural network for image generation* arXiv preprint arXiv:1502.04623, 2015.

[2] Varma, Manik and Zisserman, Andrew *A statistical approach to texture classification from single images* International Journal of Computer Vision 2005

[3] Gatys, Leon A., Ecker, Alexander S., Bethge, Matthias *Texture Synthesis Using Convolutional Neural Networks* arXiv preprint arXiv:1505.07376, 2015.

[4] Simonyan, Karen and Zisserman, Andrew *Very deep convolutional networks for large-scale image recognition*, arXiv preprint arXiv:1409.1556 2014