One of the most exciting and rapidly-developing areas of artificial intelligence and deep learning is the recognition, generation and modification of digital images. Convolutional Neural Networks (CNNs), which are adaptable to a great variety of tasks, can create artistic images through neural style transfer processes (NST).

This system uses neural representations to separate and recombine the content and style of arbitrary images, providing the neural algorithm for the creation of artistic images. These algorithms generate a new image rendered in the style of one image but contain the content of a separate image. This is done by encoding the two source images using a CNN model and then utilizing a white noise image to minimize the loss between the generated image and the source content and style images.

Neural style transfer is an example of image stylization, a non-photorealistic rendering technique that has changed dramatically over the past two decades. Prior to NST, the transfer of image style was performed using machine learning techniques based on image analogy. Given a pair of images, a transformation could be learned and then applied to create a new artwork from a new photo, by analogy. The drawback of this method is that such a training pair rarely exists in practice. NST requires no such pairing; only a single example of neural artwork is needed for the algorithm to transfer its style.

Neural style transfer is an optimal and efficient tool for visual design applications like photo-stylization, video stylization, graphic design generation, custom fabric, typography and coloring book designs.

A similar style transfer process can be used for sound design. The style of one audio sample is extracted with CNNs and transferred to a target audio sample, combining the structural and tonal elements of the target data with the beat of the styling audio. The evolution of deep CNNs to extract high-level semantic information from audio signals has enabled the synthesis of textures for the audio signal.

To maximize the utility of these promising algorithms, developers must deploy them on a sophisticated technology platform that delivers uncompromising end-user experience, lower operating costs and excellent energy efficiency.

This white paper will demonstrate a neural style transfer that was trained and tested on the Ampere eMAG™ platform. Ampere’s Armv8 64-bit servers are purpose-built for large-scale public and private cloud environments. Ampere’s cloud solutions deliver such advantages through large number of cores, high-speed connectivity, memory throughput, and cost-effectiveness so that developers can deploy their solutions. The highly integrated, purpose-built Ampere solution delivers the highest performance, lowest total cost of ownership (TCO) for private and public clouds.

**Neural Style Transfer on Images**

**Model Selection:** We use a deep CNN model to extract the features of images and provide proper encoding. Effective models such as VGG-19 have a large number of convolutional layers to provide rich feature representations on a wide variety of images. Pre-trained models are used as they provide proper encoding. VGG-19, for example, has been trained on over a million images.
Figure 1 shows the output image obtained when content image is combined with the style image. Figure 2 shows image representations and reconstructions in CNN.

Loss Function: Let \( P \) be the content image, \( D \) be the style image, and \( X \) be the white noise image (i.e., the generated image) that will constitute the final image. Total loss is the sum of content loss and style loss.

\[
L_{\text{total}}(P, D, X) = \alpha L_{\text{content}}(P, X) + \beta L_{\text{style}}(D, X)
\]

Content Loss: The content loss is the mean squared error between the encoding of the white noise image and the content image. For a layer \( l \) and the input image \( \overline{X} \), let the number of filters be \( T_r \). The output (or encoded) image will have \( T_r \) feature maps, each of size \( S_r \), where \( S_r \) is the height times width. So, the encoded image of layer can be stored in a matrix \( F_{l, e} \in \mathbb{R}^{N_l \times M_l} \).

Where \( F_{ij} \) is the activation of \( i^{th} \) filter at position \( j \) in layer \( l \).

Style Loss: For capturing the style of an artist, a style representation is used. It computes the correlations between the different filter responses, where the expectation is taken over the spatial extent of the input image. These feature correlations are given by Gram Matrix \( G_{l,e} \in \mathbb{R}^{N_l \times N_l} \), where \( G_{ij} \) is the inner product between the feature maps \( i \) and \( j \) represented by vectors in layer \( l \) and \( N_l \) is the number of feature maps.

The style loss is the mean squared error between the gram matrices of style image and the white noise image. Let \( \overline{D} \) be the style image and \( \overline{X} \) be the white noise image. Let \( A' \) and \( X' \) be the style representations of style image and white noise image in layer \( l \). So, total style loss of a layer \( l \) is \( E_l \).

The total style is:

\[
L_{\text{style}}(\overline{D}, \overline{X}) = \sum_{l} w_l E_l
\]

where \( W_l \) is the weighting factor of each layer.
Hyperparameter Tuning: To calculate the style and content loss, we use standard error back-propagation. To calculate $L_{total}$, $L_{content}$ is weighted by $\alpha$ and $L_{style}$ is weighted by $\beta$. The ratio of $\beta$ is generally kept $10^{-3}$ or $10^{-4}$, to prevent the style from dominating and therefore preventing the loss of content. We use the Adam optimizer algorithm, and find that running it for 500-2000 iterations seems to produce nice results.

Technology Overview:

Hardware: This neural style transfer was performed on Ampere’s eMAG platform with a TensorFlow 1.0 backend.

Architecture: Arm®v8 64-bit server

CPU op-mode: 64-bit

Model name: Arm®v8 64-bit server operating up to 3.3GHz

Processor Subsystem
- 32 Arm v8 64-bit CPU cores up to 3.3 GHz with Turbo
- 32 KB L1 I-cache, 32 KB L1 D-cache per core
- Shared 256 KB L2 cache per 2 cores

Memory
- 32 MB globally shared L3 cache
- 8x 72-bit DDR4-2667 channels
- Advanced ECC and DDR4 RAS features
- Up to 16 DIMMs, 1 TB/socket

System Resources
- Full interrupt virtualization
- I/O virtualization
- Enterprise server-class RAS – End-to-end data poisoning – Error containment and isolation – Background L3 and DRAM scrubbing.

Connectivity
- 42 lanes of PCIe Gen 3, with 8 controllers – x16 or two x8/x4 – x16 or two x8/x4 – x8 or two x4 – Two x1
- 4 x SATA Gen 3 ports
- 2 x USB 2.0 ports

Technology & Functionality
- TSMC 16nm FinFET+
- Arm v8.0-A, SBSA Level 3 – EL3, secure memory and secure boot support
- Advanced power management

Power
- TDP: 125 W

Software stack: The following dependencies must be installed: TensorFlow, Numpy, OpenCV, Python3 and pretrained VGG network data file.

Usage: Copy your content image to the default image content directory ./image_input. Copy one or more style images to the default style directory ./styles. Run the command:

```
bash stylize_image.sh <path_to_content_image> <path_to_style_image>
```

Note: Supported image formats include: .png, .jpg, .ppm, .pgm
Summary

This white paper illustrates how a Convolutional Neural Network can mix two images to create a new image using neural style transfer. The new image is created by applying the style of one image to the content of another image with the neural style transfer algorithm. This gives better insight into how the algorithm can be applied to different images.

The Ampere Arm 64-bit platform with optimized software produces excellent throughput with lower latency. The eMAG platform supports programmability to deliver performant power across emerging AI workloads with the flexibility to adopt the latest AI technologies.

eMAG delivers better scalability while reducing platform complexity and improving TCO. The powerful ISA architecture provides efficient memory block loads to quickly load data tiles for optimized convolution or optimized generalized matrix multiplications. All devices on the edge are moving towards implementing some form of AI, increasingly performed locally due to cost latency and privacy concerns. Arm-64 bit provides a good solution to accelerate deep learning workloads.

Ampere helps developers innovate new solutions through a broad portfolio of hardware and software offerings to enable the continuum of AI workloads from the edge to the cloud. The eMAG platform supports many pretrained models, computer vision and deep learning algorithms, enabling developers to accelerate computer vision and deep learning algorithms.

This model is inferenced using Arm®v8 64-bit operating at 3GHZ with 32 CPU cores. The software ecosystem and necessary libraries already exist to run the inference application and this demo does not require any additional software or modification to the model. The existing system includes math libraries, supports model optimization and commonly used frameworks, and improves performance on inference.

References

https://modelzoo.co/model/neuralart
https://github.com/cysmith/neural-style-tf
https://www.researchgate.net/publication/320463069_Style_Transfer_for_Audio_using_Convolutional_Neural_Networks/download
https://en.wikipedia.org/wiki/Neural_Style_Transfer