SOMNIA: Self-Organizing Maps as Neural Interactive Art

Byron V. Galbraith
Talla Inc.
Boston, MA 02116
byron@talla.com

Abstract

SOMNIA is a software tool for creating and displaying continually evolving abstract textures in real time. Based on the Self-Organizing Map neural network, SOMNIA enables both the visualization of the state of the network as it trains as well as the interactive manipulation of the network's hyperparamters.

1 Introduction

A challenge that neural network-based tools used for generative art and music have is the problem of intentionality. The intentionality of the director of an adaptive system like neural networks is often in the form of hyperparameter selection followed by manual curation of model outputs after training. Unfortunately, it may not be apparent until after a training run has completed that the hyperparameter choices made produced acceptable outputs. It is also often not clear how to achieve desired changes or improvements in the model output results — instead the creator must conduct hyperparameter sweeps, then continually evaluate the output until the model produces something acceptable.

SOMNIA (Self-Organizing Maps as Neural Interactive Art) is a real-time generative texture method based on the Self-Organizing Map (SOM) (Kohonen (1998); Du and Swamy (2013)) neural network model.

2 Self-Organizing Maps

First developed by Kohonen, the SOM is an unsupervised neural network-based clustering method inspired by the topological ordering of neuron responses in the brain (Kohonen (1998)). In SOMs, neurons are defined in an n-dimensional grid, where each neuron n not only contains an internal weight w_n, it also has a fixed location in the grid c_n.

The network is trained using a competitive (winner-take-all) approach as follows. First, a sample input x is presented to the network. The Euclidean distance between x and each neuron's weight in the SOM is computed, with the neuron closest to the input declared the best matching unit, bmu.

$$bmu = \arg\min_k \|x - w_k\|.$$

The weights of every neuron in the SOM layer are then updated according to the Kohonen learning rule:

$$w_i(t+1) = w_i(t) + \alpha(t)h(t, \|c_{bmu} - c_i\|),$$

where α is the learning rate and h is the neighborhood function. The neighborhood function produces an activation value dependent both on the current step t of the training process and the Euclidean distance between a a particular neuron and the best matching unit within the SOM grid. Choices for

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

$h()$ can be rectangular windows, Gaussians or difference of Gaussian for an on-center, off-surround effect. Typically a radius is also defined in which an neuron falling outside the radius of effect will have no impact from the training step.

3 SOMNIA

Figure 1: The state of a SOM layer at a snapshot in time.

The structural nature of SOMs has made them well-suited for visualization purposes. Using a mapping function to translate the weights of each neuron to an RGB color value enables a ready visual representation of where the adapted clusters occur within the SOM layer. In addition to displaying the final trained network, it is also possible to visualize the adaption of the network as it trains (Galbraith (2010)). By choosing large SOM layer sizes and no longer constraining hyperparameters to the realm of dimensionality reduction optimization, this visualization capability can be adapted to perform iterative generative art.

SOMNIA is a software implementation of this concept. First, a large texture is defined, where each pixel is assigned a uniformly random RGB value. Each pixel is treated as the weight of a neuron in the 2D grid. The grid is treated as toroidal, so neighborhood functions are able to wrap around both axes. A source image is provided as the data source, and each pixel from the source image is presented one at a time to the SOM network. After the training update, the texture is re-rendered to the screen to display the new state of the network. The current implementation of SOMNIA is in Python and uses OpenGL as the visualization runtime. Figure 1 shows the state of a 480x854 SOM layer captured after several training steps that used a difference of Gaussians neighborhood function with $r = 32$ and $\alpha = 32$. A photograph was used as the source data. Note how the SOM layer effectively learns to extract the base palette of the source image.

SOMNIA also supports the real-time, manual adjustment of hyperparameters through a physical interface that is controllable by the director. By exposing control of values such as the learning rate, neighborhood function properties (e.g. radius, shape, etc), and training behavior, the director is able to imbue more intentionality into the evolution of the SOM texture. This control is exposed through integration with MIDI controllers, allowing adjustments to training process using interfaces such as knobs, pads, and keys. In this way, SOMNIA enables the "performance" of a visual texture in a similar fashion to performing music.

References

Teuvo Kohonen. The self-organizing map. *Neurocomputing*, 21(1):1 – 6, 1998. ISSN 0925-2312. doi: https://doi.org/10.1016/S0925-2312(98)00030-7. URL `http://www.sciencedirect.com/science/article/pii/S0925231298000307`.

Ke-Lin Du and Madisetti NS Swamy. *Neural networks and statistical learning*. Springer Science & Business Media, 2013.

Byron V. Galbraith. Computational modeling of biological neural networks on gpus: Strategies and performance. Master's thesis, Marquette University, Milwaukee, Wisconsin, 2010.