Laser Lemon

**Team members**

Ben Papp

Edward Simendinger

Nathan Watts

**Mission Statement**

We want to make deep learning algorithms accessible to artists and others who might not have the technical expertise to make use of them otherwise.

**Problem**

In recent years there has been a dearth of deep learning algorithms and techniques, including many that can be used for artistic purposes, such as Deepdream, neural style transfer, and more. However, many of these still require a significant amount of technical skill to use, making them inaccessible to the vast majority of artists who would potentially use them

**Context**

Since there have been deep vision networks, it’s been a continuous struggle to visualize what artifacts of vision were being learned. Out of natural progression from this came networks which ‘hallucinate,’ that is, when artifacts or semantic objects are detected, the activation within the network is manipulated such that the representation of objects or features from inside the deep convolutional architecture are projected out onto the image. While the results of this process are visually interesting and thought-provoking, the ability to create such art currently lies exclusively at the hands of the software engineer that can write code at the level of inside the deep architecture. We want to use tools that digital artists would be familiar with, such as masks and gui menus, to allow artists to utilize the generative power of neural networks like they would use any other artistic tool; like a paintbrush that paints with the thoughts of a learning algorithm.

**Customer**

The end user of our product would be a digital or graphics artist looking to incorporate AI-influenced enhancements into their art. They will value a tool that is versatile, with expressive results, which manages to be accessible to individuals with no coding experience. It is important that resulting effect is able to follow from the intent of the painter; the product will not be valuable if it only makes a mushy blur on your image 2/3rds of the time.

**Challenges**

One potential challenge that we will have to deal with is the possibility that Deepdream simply isn’t fast enough for artists to incorporate into their workflow. The algorithm is fairly well understood and optimized at this point, but it will likely require GPU acceleration to have acceptable performance, especially if we’d like to include some kind of live preview before the effect is applied.

**Expertise**

We will need an understanding of the Deepdream algorithm-- the algorithm is not exceptionally difficult to implement and has many implementations available as open-source projects. In addition, the original paper and blog post from Google Deepmind provides both a high- and low-level explanation of the algorithm.

We will also need some understanding of the workflow of our target demographic-- how can we make the tool easy to integrate into their existing workflow? For this, we could communicate with members of the SMFA community.

**Risks**

Deep Learning networks are finicky things. While it may not be a requirement of our implementation to train a network itself, in making a project with an integrated network, if at some point the hallucination algorithm for example does not have the desired effect on the image, it could require significant error to debug and find a solution. While we’d like to had to the user a precise interface of parameterized control, should we find such pitfalls we may need to resort to simpler, better documented hallucination effects.

**Ethical Questions**

This project exists within a space which is somewhat controversial among artists. One potential worry is that it might take too much of the creative power away from the artists by automating the creative process, in a way. We could mitigate this issue by making the interface very transparent and making sure it has enough parameters exposed for the artist to have creative control over the result.

**Goal**

Upon completion, an artist should be able to integrate our tool into their workflow and produce a variety of works of art (to demonstrate creative control over the process).

**Documents**

To explain our product, we will advertise it as an artists’ tool. We will largely use example images of art that the product has produced as the main selling point. We will show the results of various different parameters that affect the final result, and the specific ways in which they alter the image. Additionally, we will have some figures that display how simple it is to use the interface itself.

For the more technical audience, we can advertise it with a focus on the machine learning aspects of the problem. There are some people who might not be as interested in the results as they are in the process behind it.

**Use Case/Design**

The use case we envision for this product is something that an artist can incoroprate into their existing workflow. Many digital artists use or are already proficient with digital art software such as Photoshop or Gimp, and as such, we will integrate our project into those tools using their plug-in interfaces. Both tools have a well-documented interface for creating plug-ins and extensions using python, which makes them compatible with existing machine learning frameworks such as Tensorflow and Pytorch.

The user will launch the tool from within their image editing software as they would launch any other image effect. They will be presented with an interface that allows them to select from a set of predefined presets and a few parameters that can be adjusted, chosen to generally give good results throughout their ranges. Parameters include the network, the activations to adjust, and the learning rate of the effect. The user can look at a small preview window to get an idea of what their selection will do to their image.

Power users, and those more experienced with the tool, can open up the “advanced” settings window, which will allow them to adjust a larger number of parameters, and adjust parameters outside of “safe” ranges. This will generally be less likely to give good results, but gives more creative control. Once users have set the values, they can choose to save them as a new preset to re-use later.

Additionally, if possible, we would like to establish a system for adding more networks to the system. This could be made possible using a standardized system for annotating the network structure and checkpoints, as is commonly used by frameworks such as Tensorflow and Pytorch.