Title: Creation of artistic images by applying deep learning algorithm

Main Idea:

Getting the idea from Prisma, which is a photo editor app that creates amazing photo effects transforming images into paintings. So, I want to do some optimizations on existing deep learning methods that can achieve this function. The optimization includes the relationship between certain deep learning modal works well on certain painting categories, like portrait, landscape and so on, and CNN depths, learning rate, loss function impact on how well the transformed painting looks like. And a better image processing method could also lead to a better transformed painting. Like Otsu method may only work well on processing landscape images. If we can detect which category the image belongs to before we process it, we can use different image processing methods to deal with it, then leads to a better result.

Software(tools or packages imported):

Skimage, scipy, caffe, numpy, pillow, torch7, loadcaffe, pretrained vgg network.We will write optimization code, compare which algorithm has the best effect. Detailed function illustrated in basic method.

Basic Method：

Input: Image we want to change p; Style image a, output image x

Preprocessing: Using guassian filter reducing unimportant noisy and emphasize content

$$d(x,y,k,l) = \exp(-\frac{(x-k)^2+(y-l)^2}{2\sigma_d^2})$$

Or other filter based on performance

step1： Construct our neural network based on the structure of VGG(many layers of Convd_layer+pooling)， download from the link：

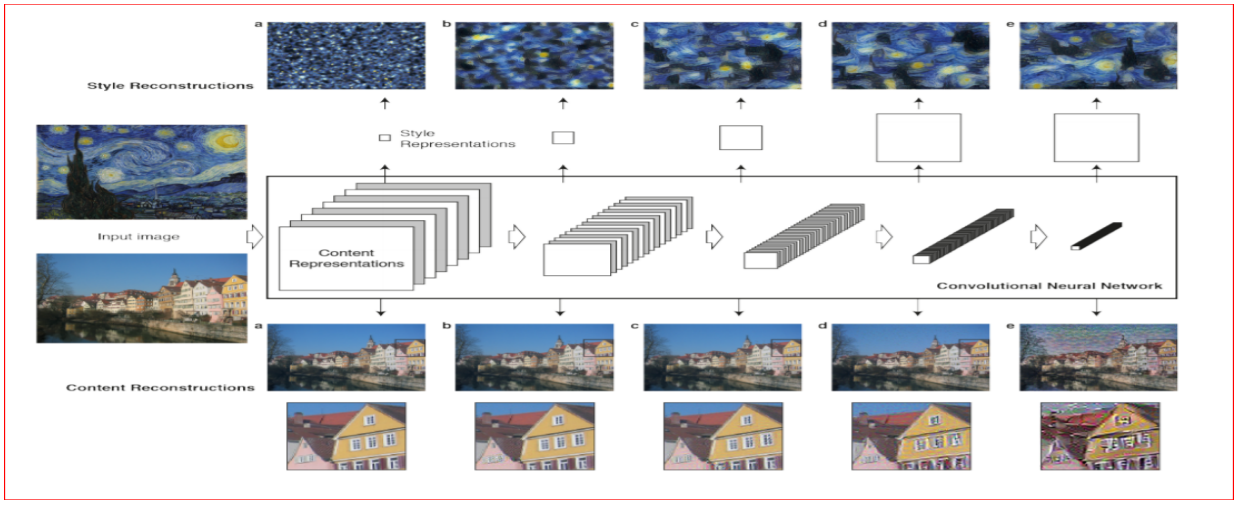
https://gist.github.com/ksimonyan/3785162f95cd2d5fee77#file-readme-md

Step 2： Capture content from p， style from a

Method option： 1. Using OTSU, extract content by extent of grey.

$$\sigma_w^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t)$$

2. Using corner_peaks in skimage package to get content/ style
3. Based on the paper's idea, reconstruct arbitrary convolutional layer to extract content or style.



Loss function：

$$\mathcal{L}_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{content}(\vec{p}, \vec{x}) + \beta \mathcal{L}_{style}(\vec{a}, \vec{x})$$

L_content: difference between input picture p and output picture x.

L_style: difference between input style picture a and output picture x.

Alpha and Beta represent how important the original picture is:

For example: increasing alpha means we wish the output image has more p's content.

Minimize this loss function means to minimize the difference between output picture and input pictures so that the output picture should alike the two input images as much as possible.

Goal:

1. Completing a style transition from one picture to another and output picture is visually beautiful.
2. Optimize existing codes by
    1. Changing or adding a new style/content capture method or do preprocessing.
    2. Experiment different learning rate or other parameters
    3. Comparing with other existing codes and experiment to get the best one

Example inputs and output:

Paper Reference：

Leon A. Gatys, Alexander S. Ecker, Matthias Bethge 《a neural Algorithm of Artistic Style》

Code reference:

https://github.com/bbfamily/prisma_abu/blob/master/README.md

https://github.com/jcjohnson/neural-style(implementation of the paper, unknown tools)

https://github.com/anishathalye/neural-style(implementation in tensorflow)

Teammate: Muyi Zhu, Yuanzhao Li

Progress Milestones: reading related materials and finding feasible optimization ways for the code by April 8. Doing experiments on one certain painting category, like landscape painting, gets the best deep learning model by April 14. Doing experiments on other painting categories, like portrait, by April 30. Finish the report and code by May 12.