Motivation
Image style transfer is a long-standing problem that seeks to transfer the style of a reference style image onto another input picture. Our project has implemented two recent proposed image style transfer algorithms based on convolutional neural networks.

Model
Neural algorithm:

1. Content Loss
   \[ L_c(y_c, y) = \frac{1}{2} \sum_{ij} (F^t_{ij}(y_c) - F^t_{ij}(y))^2 \]

2. Style Loss
   \[ G_{ij}^l = \sum_k F^l_{ik} F^l_{jk} \]
   \[ E_l(y, y_s) = \frac{1}{4N_l^2 M_l^2} \sum_{ij} (G^l_{ij}(y) - G^l_{ij}(y_s))^2 \]
   \[ L_s(y, y_s) = \sum_l w_l E_l \]

3. Joint Loss Function
   \[ L_j(y_c, y_s, y) = \alpha L_c(y_c, y) + \beta L_s(y, y_s) \]

4. Output:
   \[ \hat{y} = \arg \min_y L_j(y_c, y_s, y) \]

Realtime style transfer: Train an Image Transform Network

Training Details
Two epochs over the 80k Microsoft COCO dataset with batch size 4 (resize to 256x256). We use a pre-trained VGG16 network for the loss calculation and an image transformation network with 3 convolution layers, 5 residual layers and 3 deconvolution layers.

Results
Neural algorithm results with different style weights.

Comparison of two methods:

Style
Baseline
Real-time

Discussion
Important details for final results:
1. Input images need preprocessing (subtract mean value and normalize) to make loss network work properly.
2. Projecting the values of generated images to range [0, 255] to get rid of some noisy artifacts.
3. During the training process, allowing the image transformation network to work in range [0, 255] generates much better results than than [0, 1].

Future Plans
1. Add photorealistic loss and semantic segmentation to reduce local distortion in original neural algorithm
2. Implement realtime algorithm for photo style transfer
3. Train a style extraction network to realize arbitrary image style transfer

References