Arbitrary Style Transfer

Computer Vision

D'ORAZIO ANTONIO, 1967788

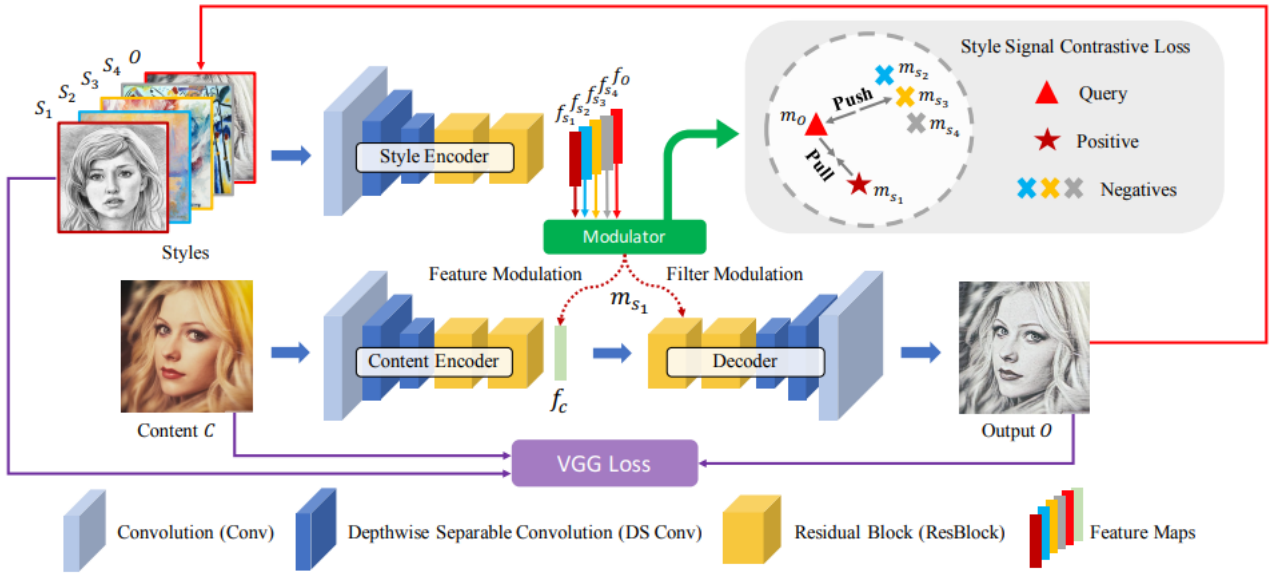
MINUT ROBERT ADRIAN, 1942806

ZARBA MELI GIACOMO, 1807439

Overview

- **Code refactoring with newer libraries**
  - PyTorch Lightning
  - Torchmetrics
  - Weights and Biases
- **Different experiments** to improve the performance from the baseline code
  - Architecture
  - Loss functions
- **Model fine-tuning** for two different tasks
  - Generic
  - Faces
- **Demo app** for testing the model

Reference Paper



MicroAST

Key points:

- Encode and decode content and style through lightweight components

  Two **micro-encoders,** for encoding the content and style features

  A **micro-decoder**, for applying the style on the content image

  A **dual-modulator** which helps to encode and decode styles

- Use the VGG only at training time to compute the loss
- At inference time, only use the learned components

Inference pipeline

- **Encode the content image**

  Content Encoder: $f_c \coloneqq E_c(C)$

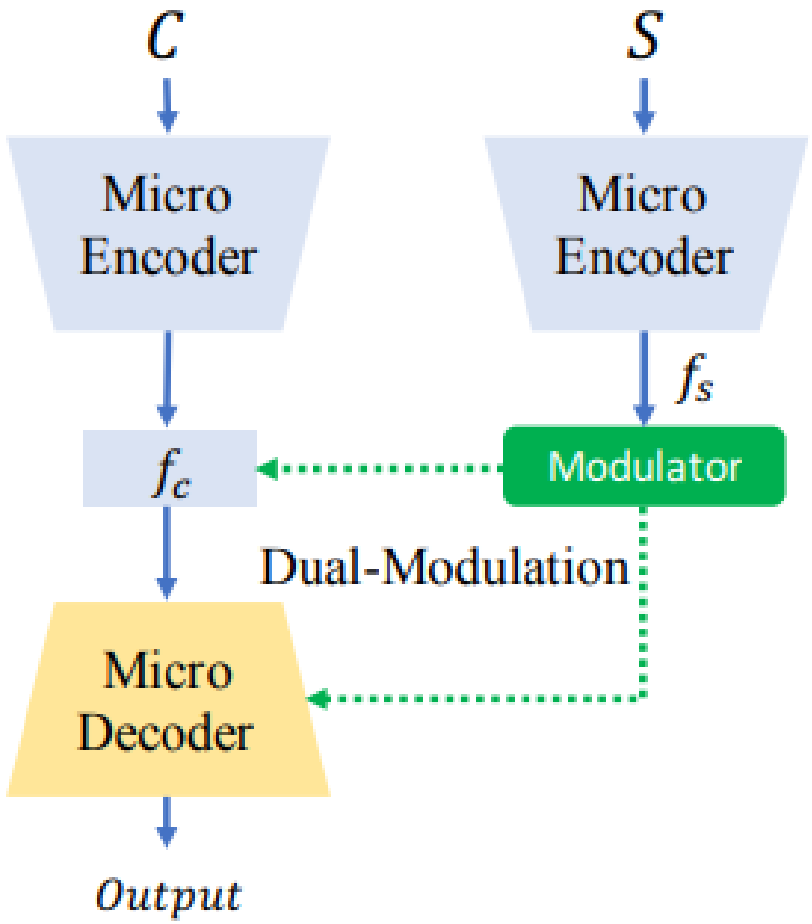
- **Encode the style image**

  Style Encoder: $f_s \coloneqq E_s(S)$

  Modulator: $m_s \coloneqq M_s(f_s)$

- **Apply the style and generate the output**

  Decoder: $O_s \coloneqq D(f_c, m_s)$



Encoders and decoder

Designed to be lightweight

- **Content Encoder** f_c, **Style Encoder** f_s

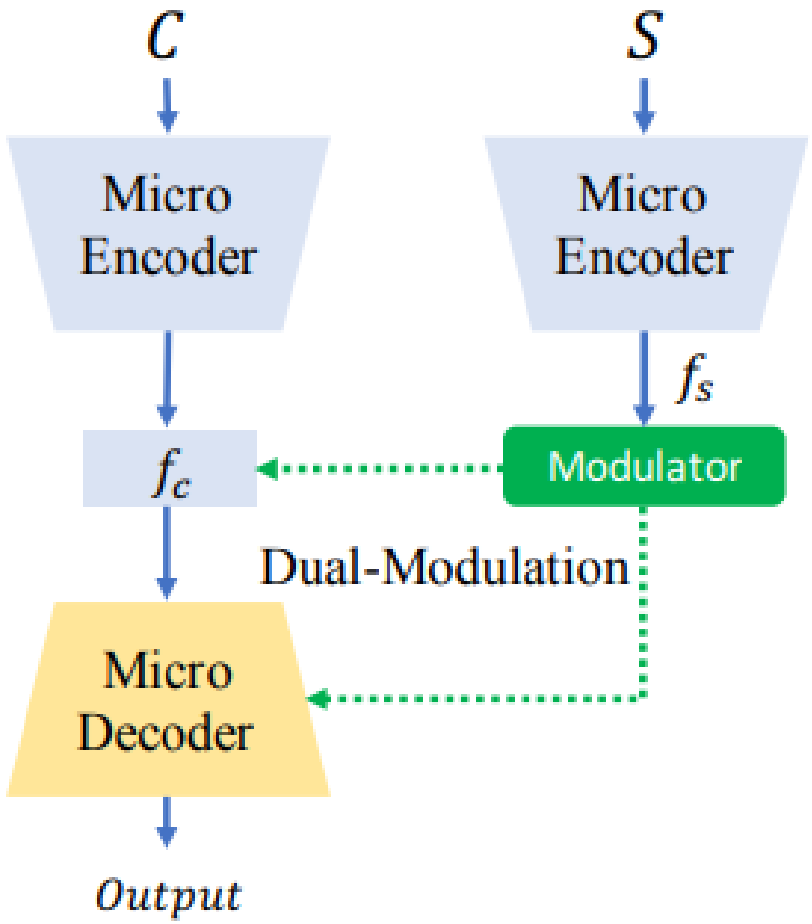
  One standard stride-1 convolutional (Conv) layer

  Two stride-2 depthwise separable convolutional (DS Conv) layers

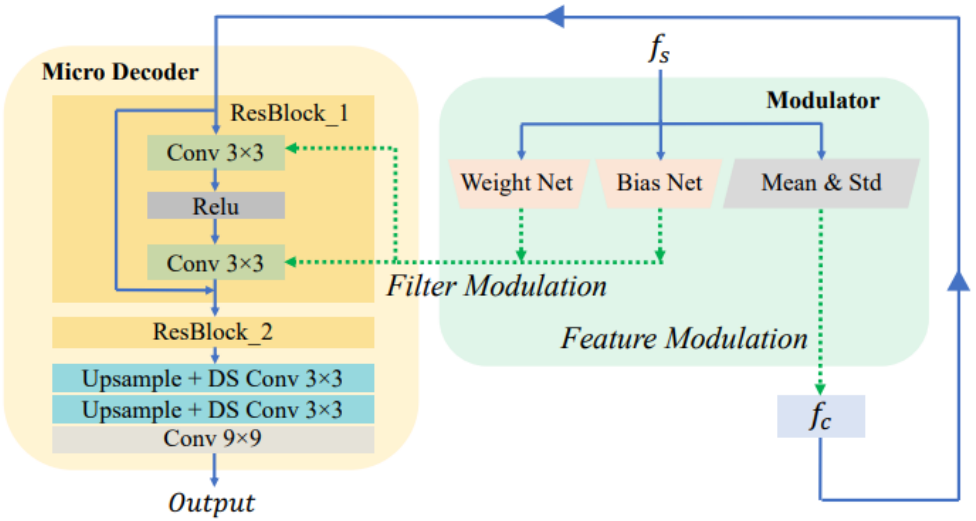
  Two stride-1 residual blocks (ResBlocks)

- **Decoder** D

  Symmetrical to the encoders



The Dual-Modulator



Captures the styles signal of the encoded features

- **DualMod:**

$$DualMod(D, f_c, m_s) \coloneqq FeatMod\big(f_c, (\mu_s, \sigma_s)\big) + FilterMod(D, (w_s, b_s))$$

- **FeatMod**: $FeatMod\big(f_c, (\boldsymbol{\mu_s}, \boldsymbol{\sigma_s})\big) \coloneqq \boldsymbol{\sigma_s}\left(\frac{f_c - \mu(f_c)}{\sigma(f_c)}\right) + \boldsymbol{\mu_s}$

  Captures the global attributes

  Uses the learned channel-wise mean and std of the style features to modulate the content features

- **FilterMod**: $FilterMod\big(D, (\boldsymbol{w_s}, \boldsymbol{b_s})\big) \coloneqq ResBlock\big(f_c, (\boldsymbol{w_s}, \boldsymbol{b_s})\big)$

  Captures the local textures (brushstrokes)

  The result is injected into the decoder D to modulate its Conv filters in the ResBlock

Where:

$$\boldsymbol{\mu_s} \coloneqq \mu(f_s),$$
$$\boldsymbol{\sigma_s} \coloneqq \sigma(f_s),$$
$$\xi = NeuralNetwork,$$
$$\boldsymbol{w_s} \coloneqq \xi_w(f_s),$$
$$\boldsymbol{b_s} \coloneqq \xi_b(f_s),$$
$$m_s \coloneqq (\boldsymbol{\mu_s}, \boldsymbol{\sigma_s}, \boldsymbol{w_s}, \boldsymbol{b_s})$$

Paper’s proposed loss

- **Content Loss** (i.e., perceptual loss)

  MSE of features at layer relu4_1 of the VGG-19

- **Style Loss**

  To match the Instance Normalization statistics, computed at layers {relu1_1, relu2_1, relu3_1, relu4_1}

- **Style Signal Contrastive Loss**

$$L_{contrastive} = \sum_{i=1}^{N} \frac{||m_{o_i} - m_{s_i}||_2}{\sum_{j \neq i}^{N} ||m_{o_i} - m_{s_j}||_2}$$

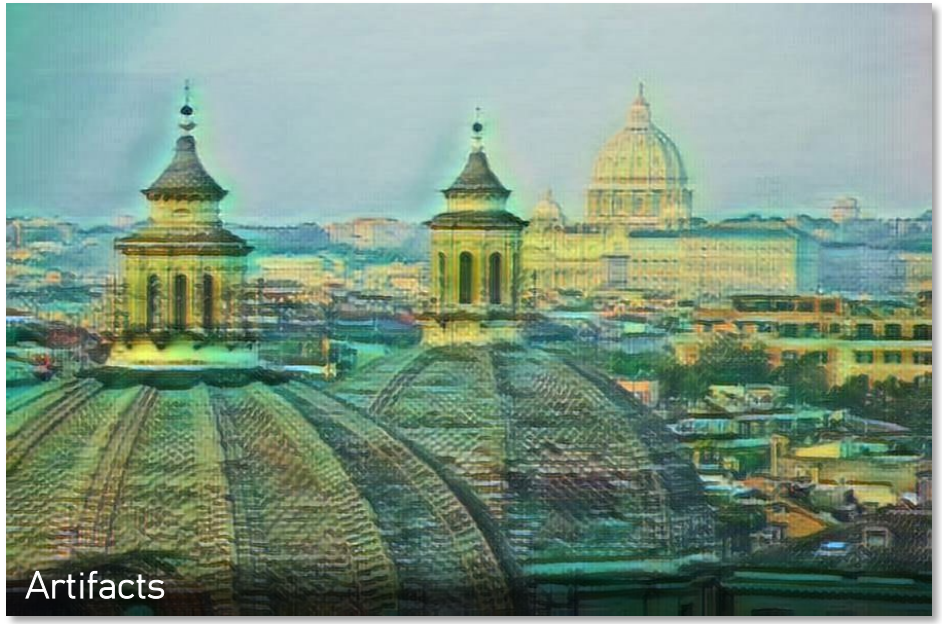
- **The final loss is**

$$L_{full} := \lambda_{content} L_{content} + \lambda_{style} L_{style} + \lambda_{contrastive} L_{contrastive}$$

The model

Challenges

- MicroAST produces artifacts by upsampling with bilinear interpolation
- The style may not be applied correctly
- Doesn't work well for portraits



Experiments

- **Decoder**
  - Transposed Convolution
  - 15x15 Convolution after bicubic upsampling
- **Loss**
  - L1
  - LPIPS (Learned Perceptual Image Patch Similarity)
  - Cosine distance
  - Regularization (Total Variation, Frechet distance)
- **Datasets**
  - LAION
  - FFHQ
  - Wikiart Portraits

Datasets

We used different datasets

- **Style**

  WikiArt (default): paintings in different artstyles

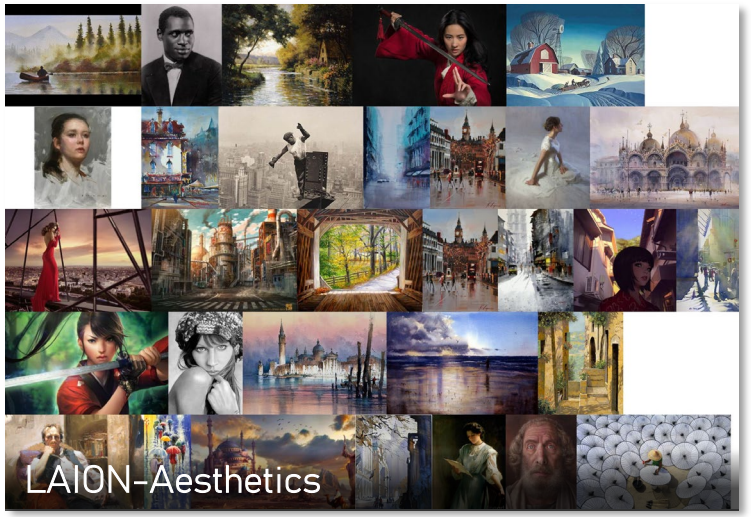
  LAION-Aesthetics: web images, including web artworks

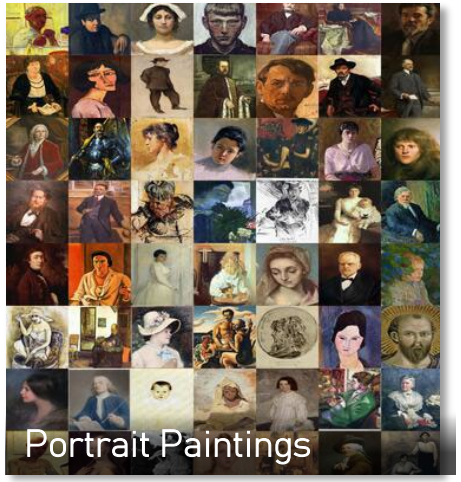
  Portrait Paintings: subset of WikiArt with paintings of portraits

- **Content**

  COCO 2017 (default): pictures of generic objects

  Flickr-Faces-HQ: pictures of faces





Loss functions and distance metrics

Metrics among the VGG layers

- **MSE** (default):

$$MSE(y_{pred}, y_{true}) = \frac{1}{N}\sum_{i=1}^{N}(y_{true_i} - y_{pred_i})^2$$

- **L1**

  More considerate of outliers.

$$\mathrm{L1}(y_{pred}, y_{true}) = \frac{1}{N}\sum_{i=1}^{N} |y_{true_i} - y_{pred_i}| .$$

- **Cosine similarity**:

  We got NaN loss during training.

$$\mathrm{S}_{\cos}(Y_{pred}, Y_{true}) = \frac{Y_{pred} \cdot Y_{true}}{||Y_{pred}||\ ||Y_{true}||}$$

Similarity metrics for the computed image

- **Frechet Inception Distance:** measures the quality of the generated images.

  It didn't suit our task.

Regularization

- **Total variation**

  Pushes down the amount of varying pixels in the image, acting as a denoiser

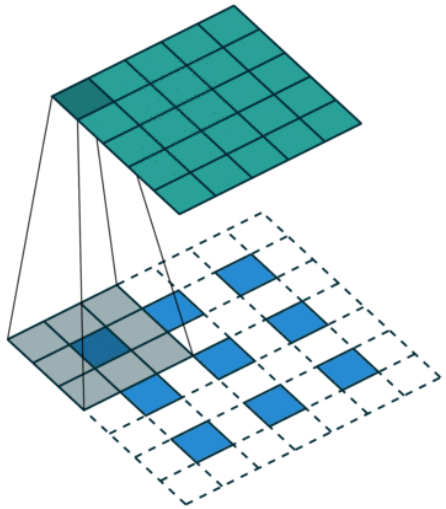
$$TV(y) = \sum_{i,j} |y_{i+1,j} - y_{i,j}| + |y_{i,j} - y_{i,j+1}|$$

Optimizations

- **LPIPS**

  use VGG weights fine-tuned for perceptual tasks

Transposed Convolution

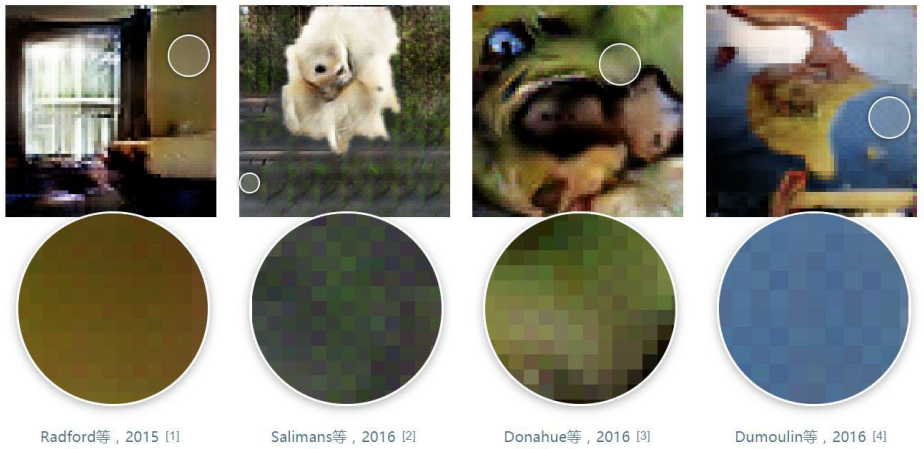


- K x K parametrized filter is used for upsampling the image

$$H_{out} = (H_{in}-1)\times\text{stride}[0]-2\times\text{padding}[0]+\text{dilation}[0]\times(\text{kernel\_size}[0]-1)+\text{output\_padding}[0]+1$$

$$W_{out} = (W_{in}-1)\times\text{stride}[1]-2\times\text{padding}[1]+\text{dilation}[1]\times(\text{kernel\_size}[1]-1)+\text{output\_padding}[1]+1$$

- Problem: checkerboard artifacts
(many convolutional layers required to smooth it out)



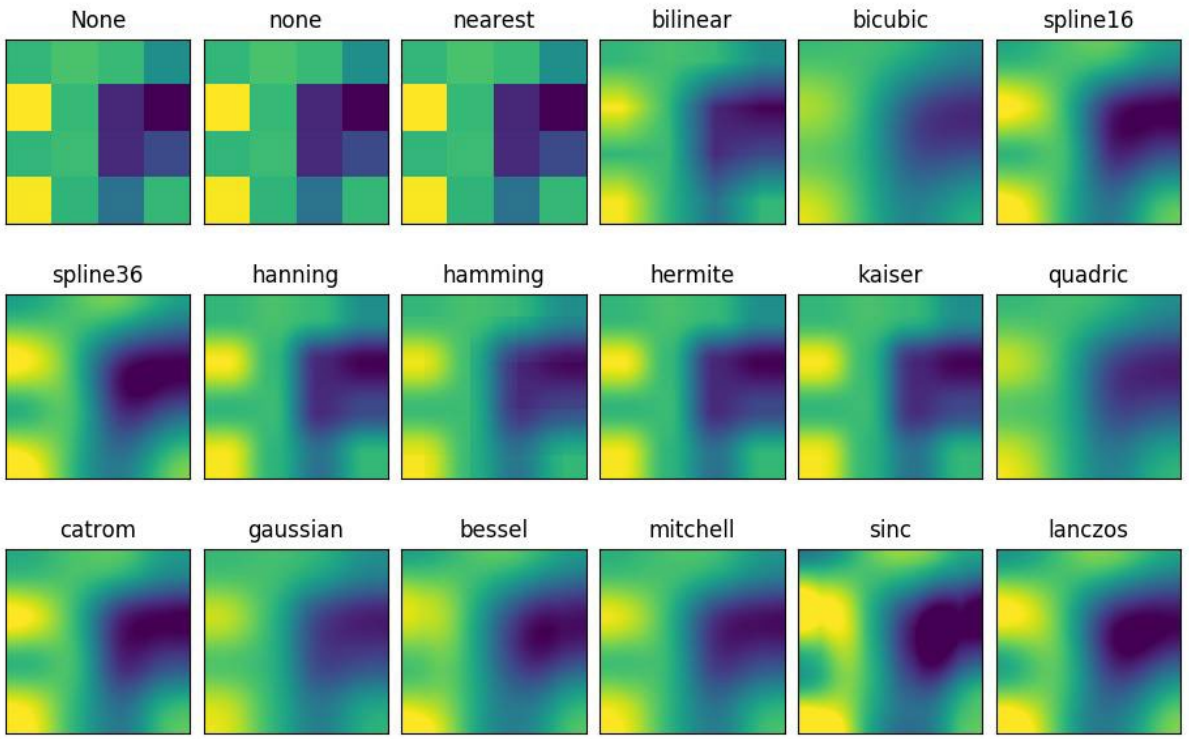
Alternative

Replace the transpose convolution with:

- **Upscaling** with bicubic interpolation
- **Convolution** with a big filter (15 x 15)

  To enhance the edges

  Big filter size to support high-res images



Hyperparameters Tuning

- **Content weight**: $\{0.5, 0.75, 1.0\}$

  To give the model more freedom to edit the content

- **Style weight**: $\{3.0, 6.0, 9.0\}$

  To apply the style more

- **Total Variation weight**: $\{1e^{-5}, 4e^{-5}, \}$

  To denoise the image

- **Style Signal Contrastive weight**: $\{3.0, 6.0\}$

  To avoid collapse to NaN

For more details: Weights & Biases Report


#access {
display: inline-block;
height: 69px;
float: right;
margin: 11px 28px 0px 0px;
max-width: 800px;
}
#access ul {
font-size: 13px;
list-style: none;
margin: 0 0 0 -0.8125em;
padding-left: 0;
z-index: 99999;
text-align: right;
}
inline-block;
left;

The web app

About the app

Why an application?

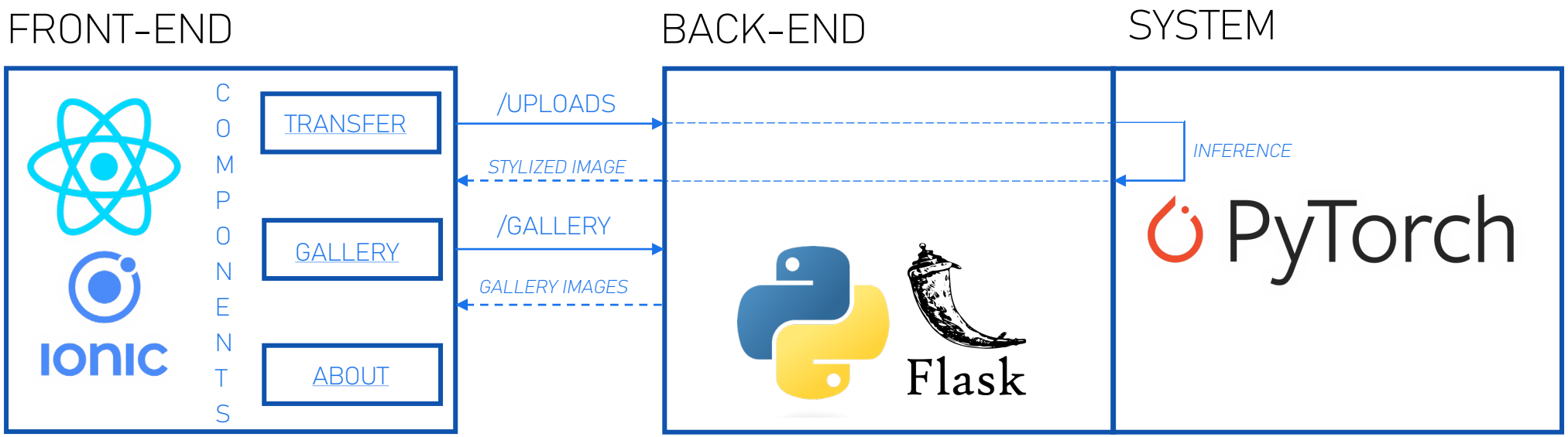
- Easier testing through UIs
- Provide end-users with an easy and customizable tool to edit their images
- Allow end-users to choose and understand differences between models and images

Client-server Application vs Pure Mobile Application

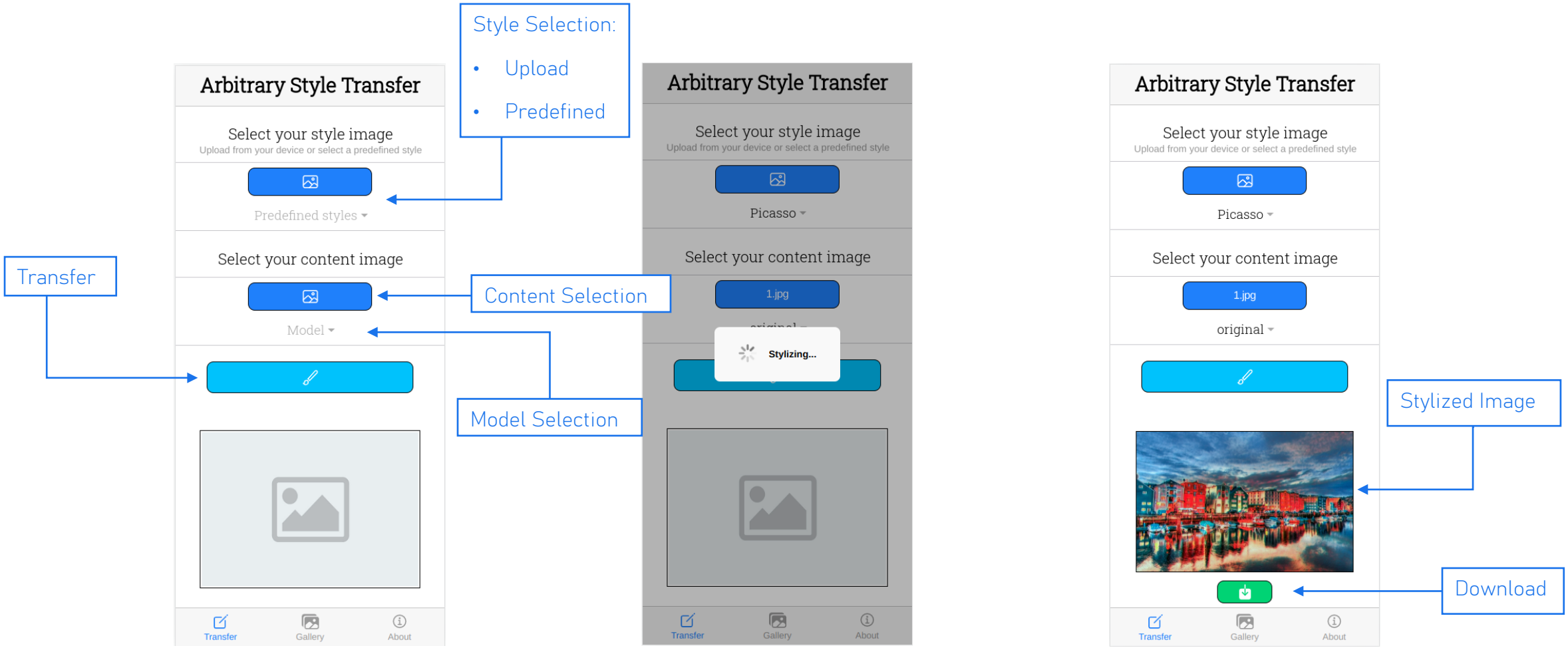
Use-Cases:

- Style Transfer (with relative customization)
- Gallery (with relative information associated to images)
- Source Code access

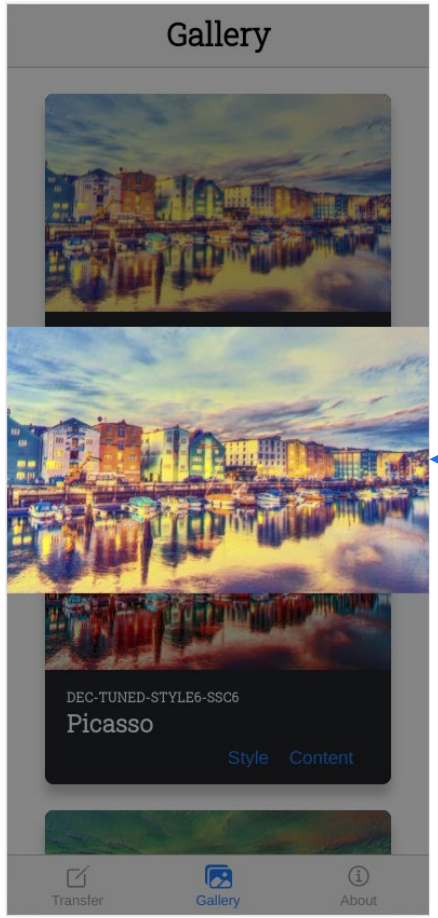
App Architecture

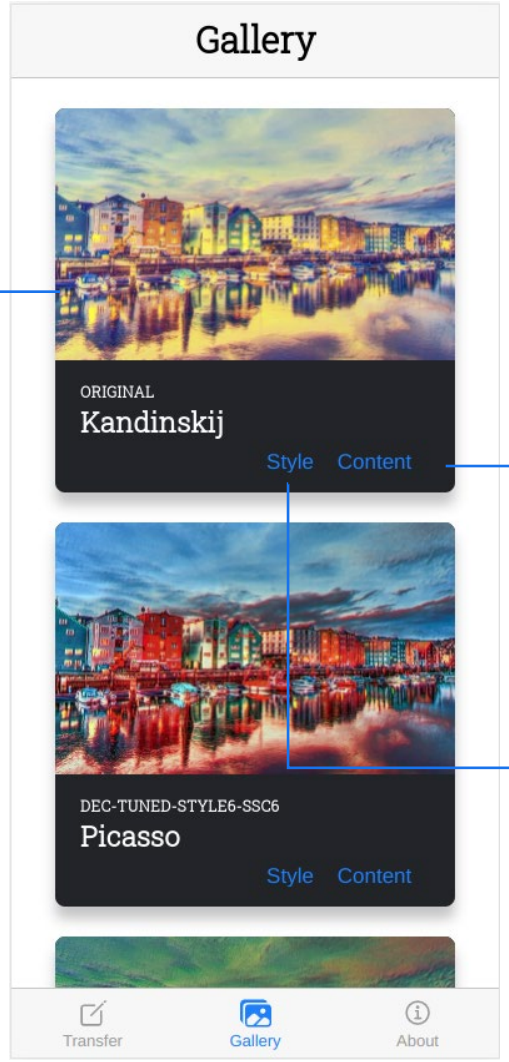


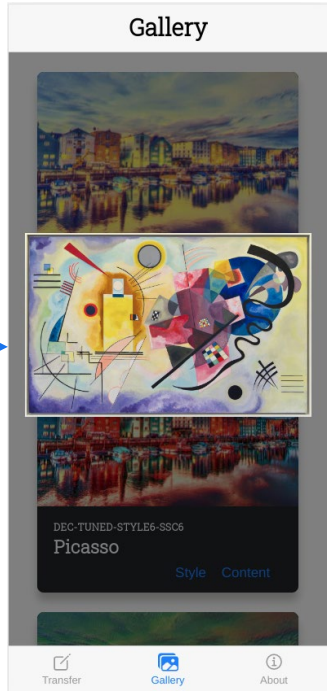
Demo: Style Transfer



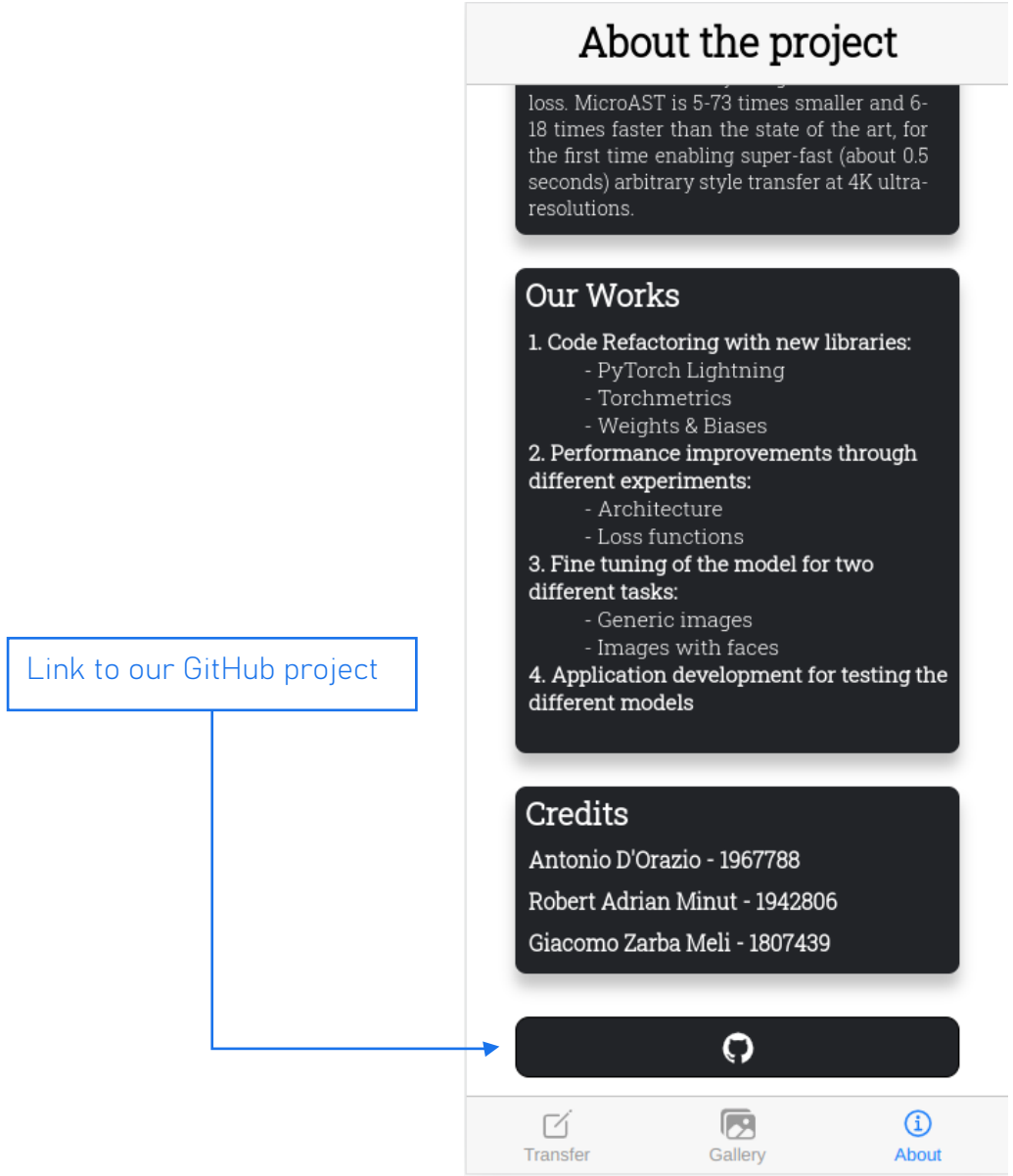
Demo: Gallery



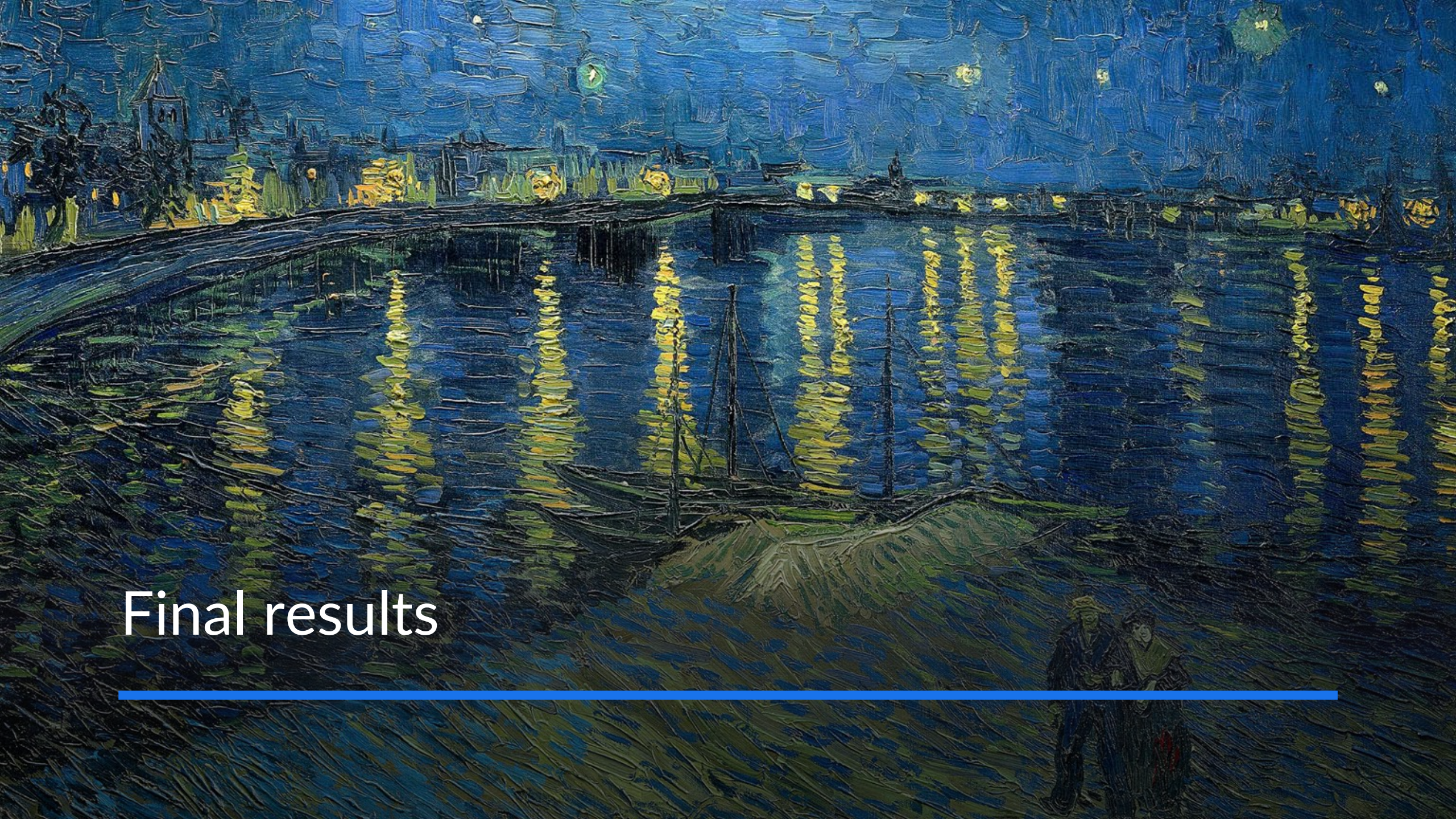




Demo: About

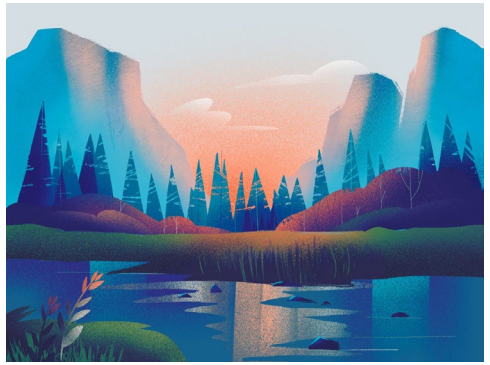


Link to our GitHub project

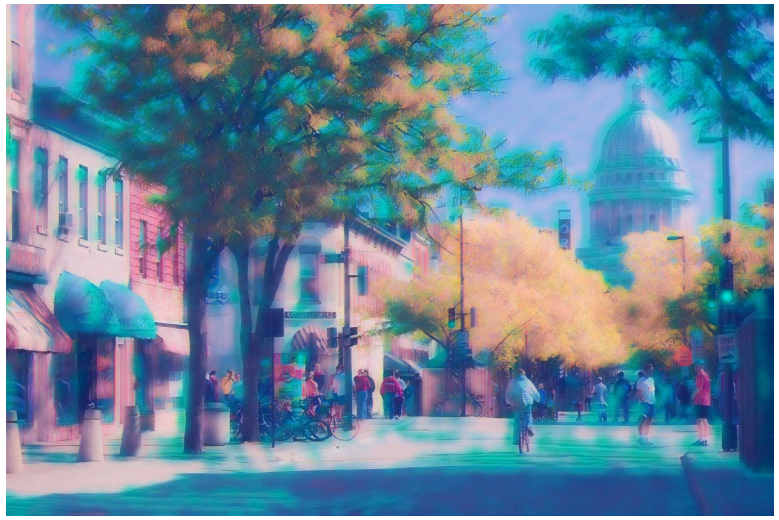


Final results

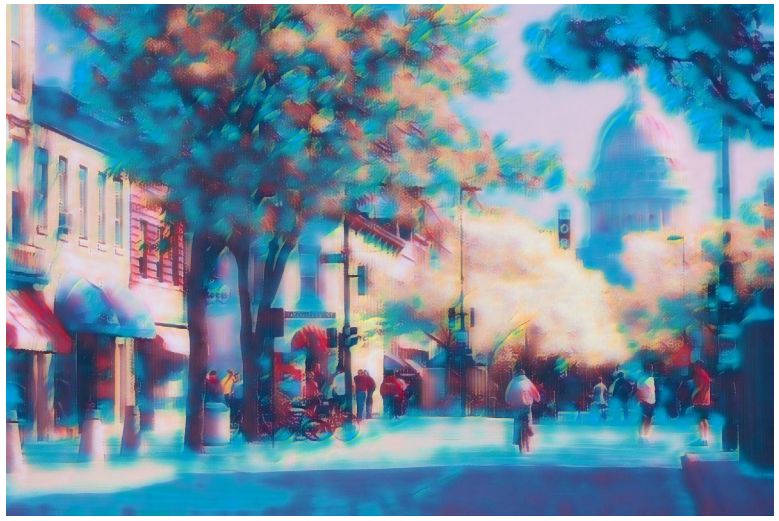
Generic



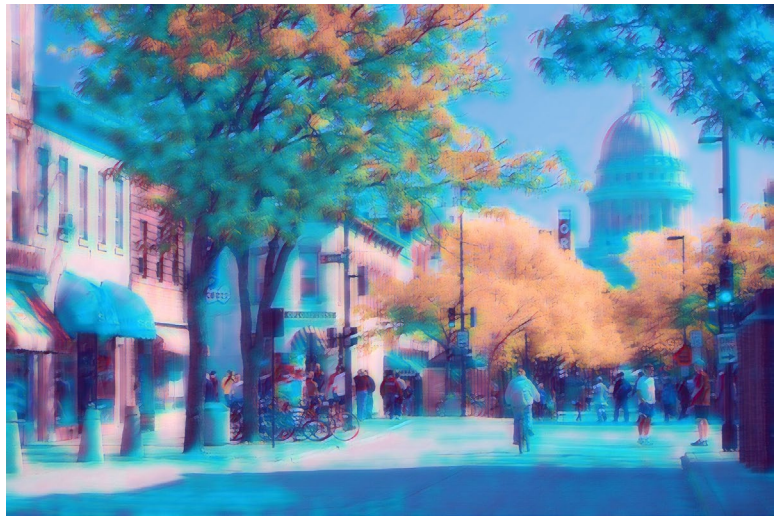
Original



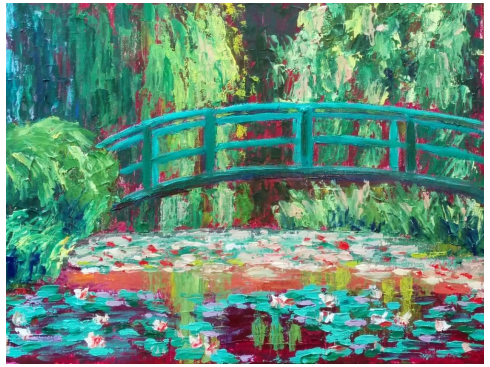
CoCo/WikiArt + Decoder Tuned



CoCo/WikiArt + Decoder Tuned + TV



Generic



Original

CoCo/WikiArt + Decoder Tuned

CoCo/WikiArt + Decoder Tuned + TV

- *total variation regularization
- using MSE unless specified

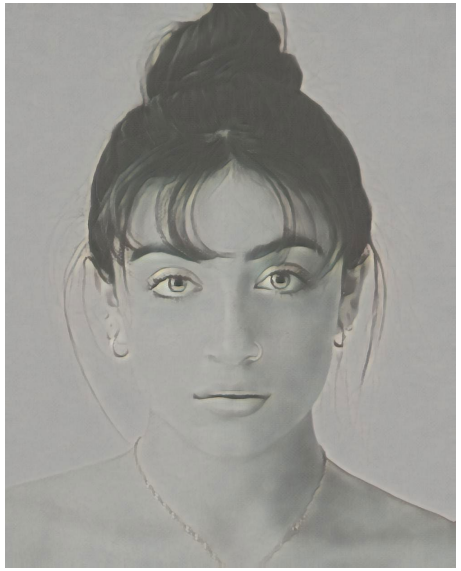
Faces

Original

FFHQ/LAION*

CoCo/WikiArt* +
Finetune FFHQ/LAION*

CoCo/WikiArt



CoCo/WikiArt*

CoCo/WikiArt* +
Finetune WikiArt Portraits*

- *total variation regularization
- using MSE unless specified

Faces

Original

CoCo/WikiArt* + Finetune FFHQ/WikiArt Portraits*, L1, Style=6

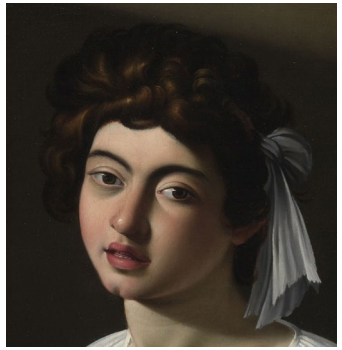
CoCo/WikiArt* + Finetune FFHQ/LAION*, L1, Style=9, Content=0.75

FFHQ/WikiArt Portraits*

CoCo/WikiArt Portraits*

- *total variation regularization
- using MSE unless specified

Faces



Original

FFHQ/LAION*

CoCo/WikiArt* +
Finetune FFHQ/LAION*

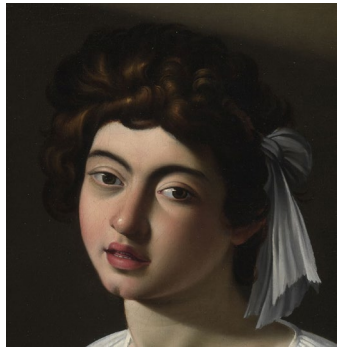
CoCo/WikiArt

CoCo/WikiArt*

CoCo/WikiArt* +
Finetune WikiArt Portraits*

- *total variation regularization
- using MSE unless specified

Faces



Original

CoCo/WikiArt* + Finetune FFHQ/WikiArt Portraits*, L1, Style=6

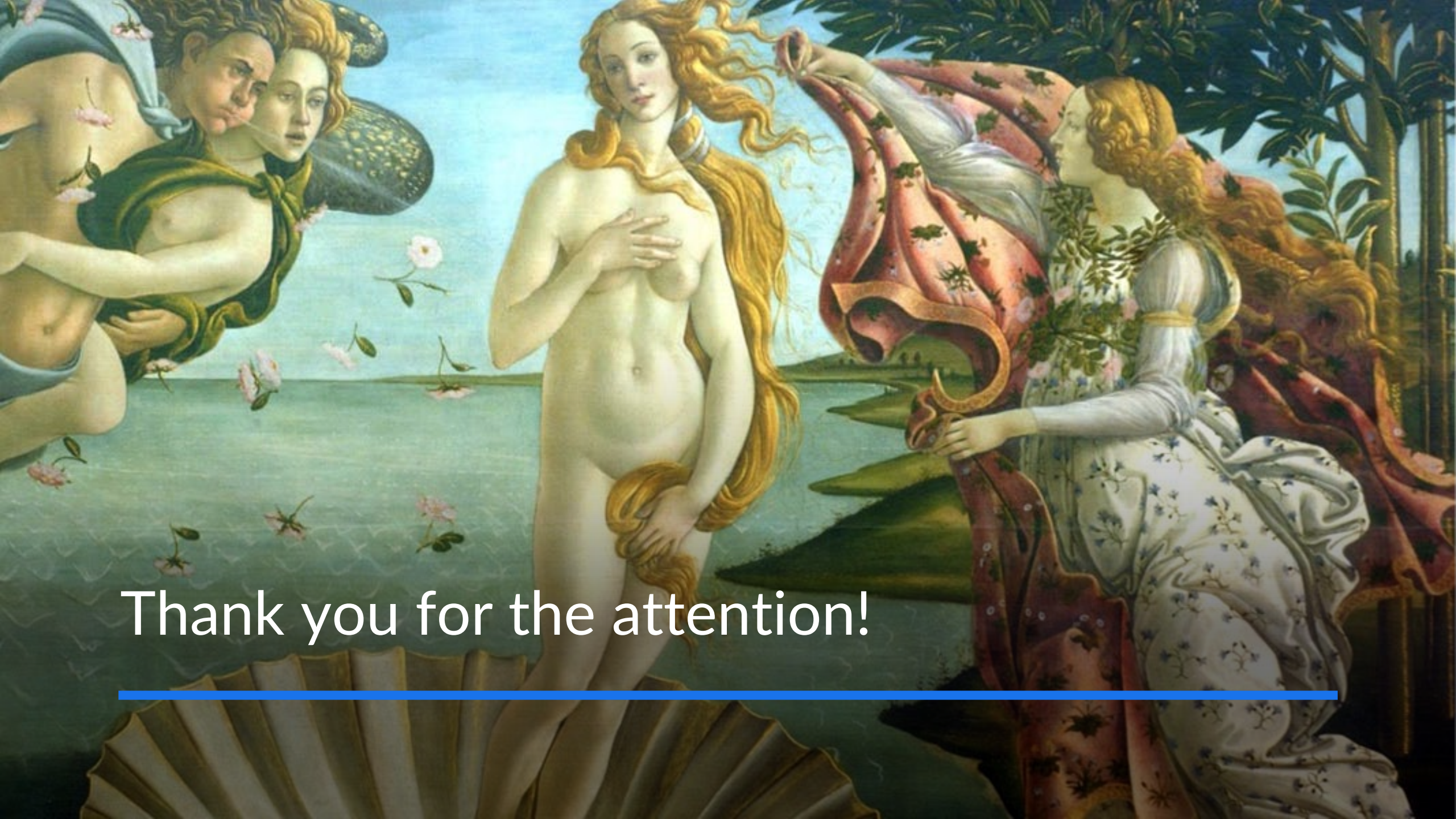
CoCo/WikiArt* + Finetune FFHQ/LAION*, L1, Style=9, Content=0.75

FFHQ/WikiArt Portraits*

CoCo/WikiArt Portraits*

References

- MicroAST: Towards Super-Fast Ultra-Resolution Arbitrary Style Transfer
- How to get beautiful results with neural style transfer
- Winning at loss functions: 2 important loss functions in computer vision
- Experiments on different loss configurations for style transfer
- Deep image quality assessment
- The Unreasonable Effectiveness of Deep Features as a Perceptual Metric (arxiv.org)



Thank you for the attention!