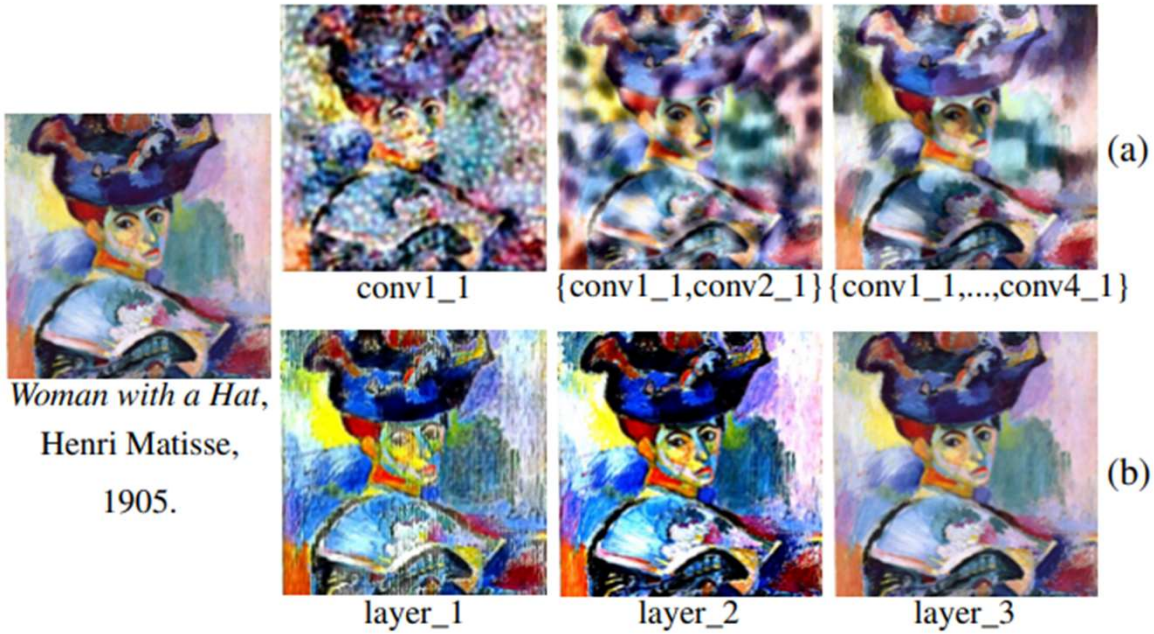
StyTr2:Image Style Transfer with Transformers

(CVPR 2022)

Hakjun Moon
gloriel621@g.skku.edu
TNT Vision
2024/05/07

TRAIN AND TEST

Introduction



(a): CNN Decoder Feature Visualization -> feature and fine detail loss
(b): Transformer Decoder with Feature Visualization

TRAIN AND TEST

Two differences from NLP Transformer

1. Sequence token of image is related to Contents Information.

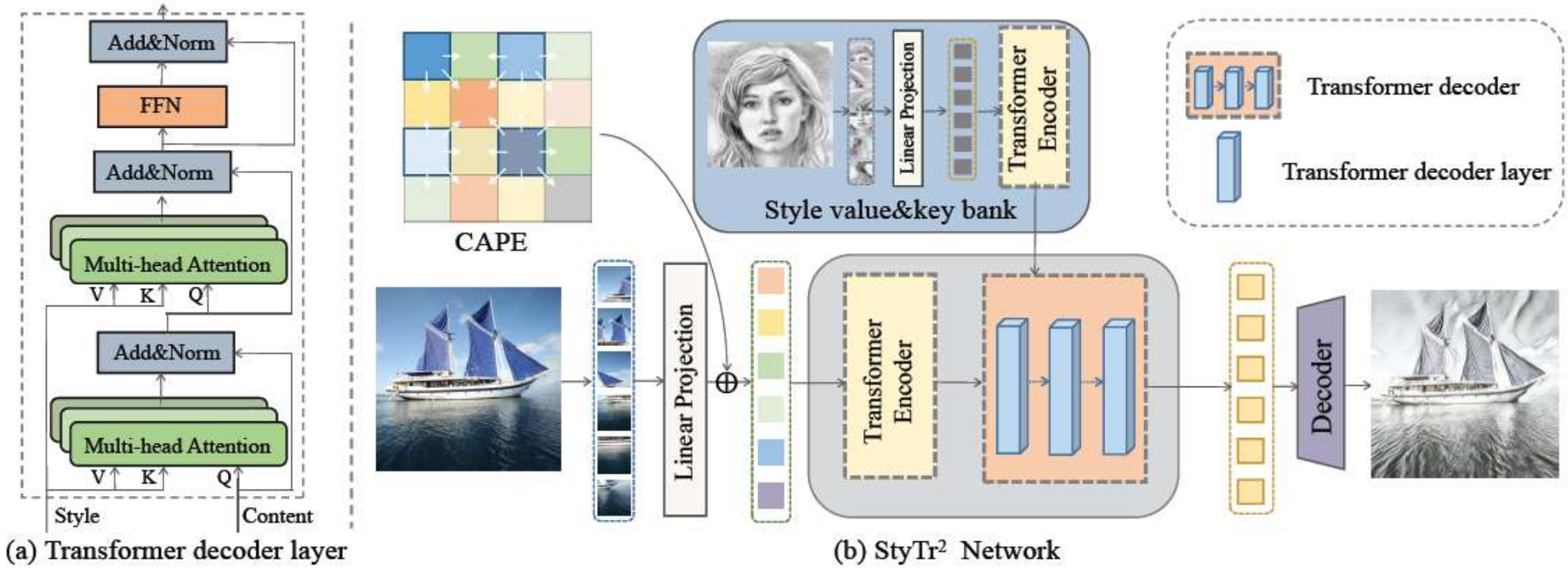
2. Uses CAPE, which is robust against semantic features and dynamic resolution of the image.

Main Contributions

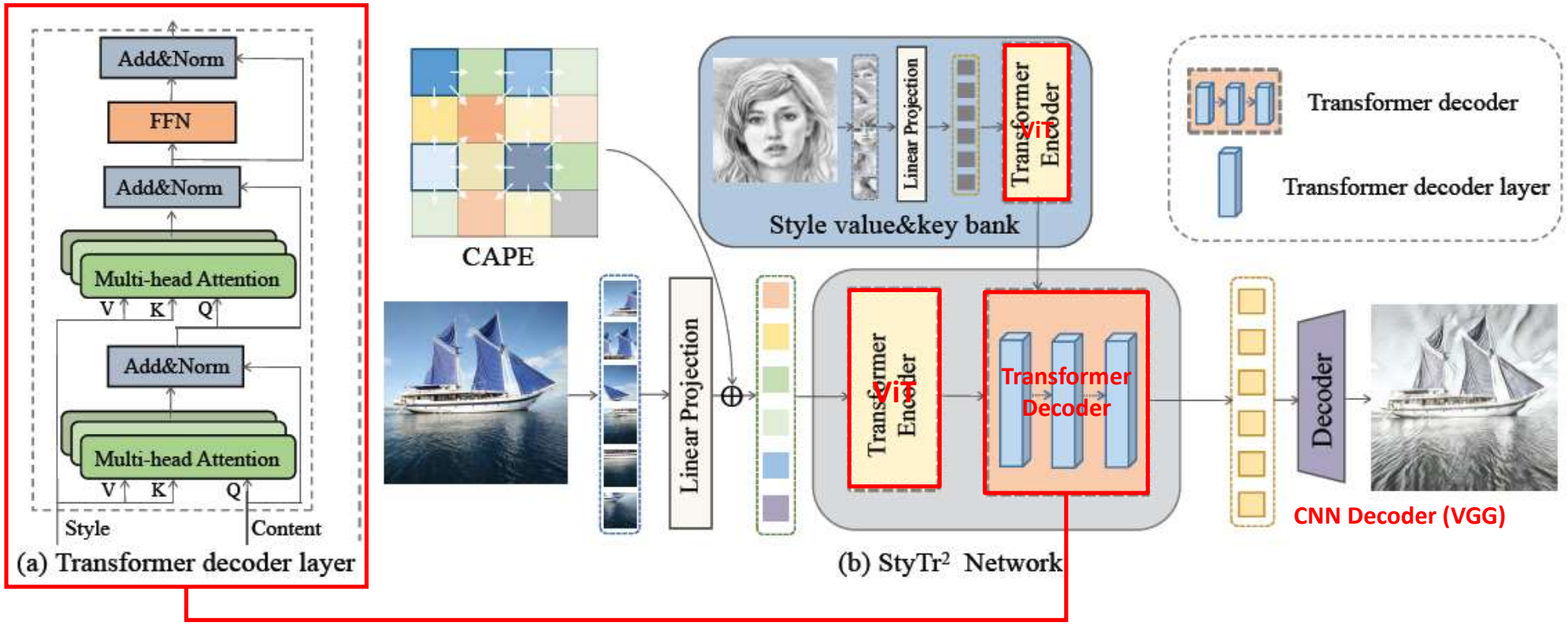
1. A transformer-based style transfer framework called StyTr2
2. A content-aware positional encoding scheme (CAPE)
3. Good experiment results

TRAIN AND TEST

Overall Architecture



Overall Architecture



Style Transfer Transformer Encoder

1. the embedding of an input content sequence

$$Z_c = \{\mathcal{E}_{c1} + \mathcal{PCA}1, \mathcal{E}_{c2} + \mathcal{PCA}2, ..., \mathcal{E}_{cL} + \mathcal{PCA}L\}$$

$$Q = Z_c W_q, \quad K = Z_c W_k, \quad V = Z_c W_v, \tag{4}$$

where $W_q, W_k, W_v \in \mathbb{R}^{C \times d_{head}}$. The multi-head attention is then calculated by

$$\mathcal{F}_{\text{MSA}}(Q, K, V) = \text{Concat}(\text{Attention}_1(Q, K, V), \ldots, \text{Attention}_N(Q, K, V))W_o, \tag{5}$$

(MSA = Multi Head Self Attention)

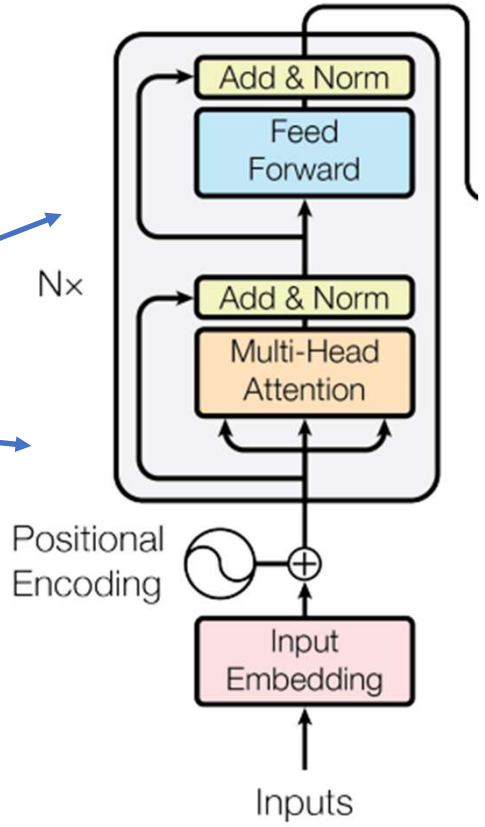
2. the embedding of an input style sequence

$$Z_s = \{\mathcal{E}_{s1}, \mathcal{E}_{s2}, ..., \mathcal{E}_{sL}\}$$

Similarly, the embedding of an input style sequence $Z_s = \{\mathcal{E}_{s1}, \mathcal{E}_{s2}, ..., \mathcal{E}_{sL}\}$ is encoded into a sequence Y_s following the same calculation process, except that positional encoding is not considered because we do not need to maintain structures of the input style in the final output.

[Add and Norm + Feed Forward]

$$Y'_c = \mathcal{F}_{\text{MSA}}(Q, K, V) + Q,$$
$$Y_c = \mathcal{F}_{\text{FFN}}(Y'_c) + Y'_c,$$



TRAIN AND TEST

Style Transfer Transformer (Decoder)

content sequence = Query

style sequence = Key, Value

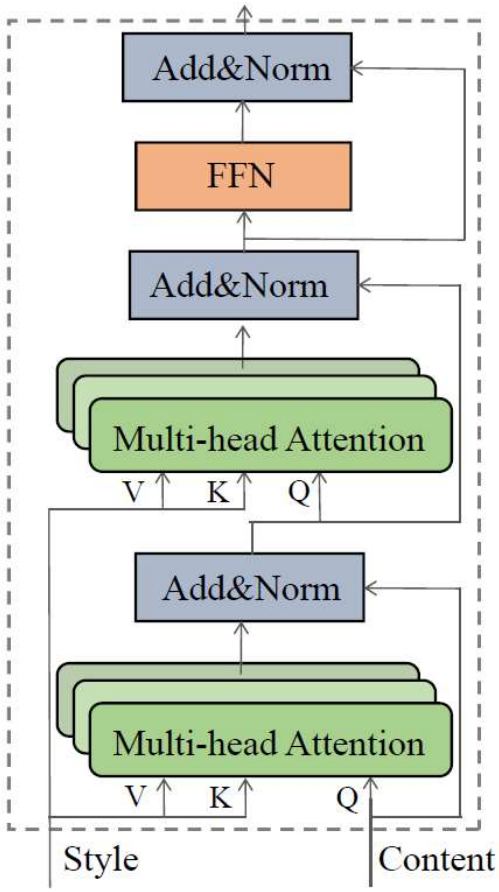
transformer decoder includes the encoded content sequence, i.e., $\hat{Y}_c = \{Y_{c1} + \mathcal{P}_{\mathcal{CA}1}, Y_{c2} + \mathcal{P}_{\mathcal{CA}2}, ..., Y_{cL} + \mathcal{P}_{\mathcal{CA}l}\}$, and the style sequence $Y_s = \{Y_{s1}, Y_{s2}, ..., Y_{sL}\}$. We use the content sequence to generate the query Q, and use the style sequence to generate the key K and the value V:

$$Q = \hat{Y}_c W_q, \quad K = Y_s W_k, \quad V = Y_s W_v. \tag{7}$$

Then, the output sequence X of the transformer decoder can be calculated by

$$\begin{aligned} X'' &= \mathcal{F}_{\text{MSA}}(Q, K, V) + Q, \\ X' &= \mathcal{F}_{\text{MSA}}(X'' + \mathcal{P}_{\mathcal{CA}}, K, V) + X'', \\ X &= \mathcal{F}_{\text{FFN}}(X') + X'. \end{aligned} \tag{8}$$

Layer normalization (LN) is also applied at the end of each block [22].



TRAIN AND TEST

CAPE: Content-Aware Positional Encoding

[Traditional Positional Encoding using Transformer]

1. Attention Score for ith, jth patch

$$A_{i,j} = ((\varepsilon_i + \mathcal{P}_i)W_q)^T((\varepsilon_j + \mathcal{P}_j)W_k) = W_q^T\varepsilon_i^T\varepsilon_j W_k + W_q^T\varepsilon_i^T\mathcal{P}_j W_k + W_q^T\mathcal{P}_i^T\varepsilon_j W_k + W_q^T\mathcal{P}_i^T\mathcal{P}_j W_k, \quad (1)$$

2. The relationship between pixel (x_i, y_i) and (x_j, y_j)

$$\mathcal{P}(x_i, y_i)^T\mathcal{P}(x_j, y_j) = \sum_{k=0}^{\frac{d}{4}-1}[cos(w_k(x_j - x_i) + cos(w_k(y_j - y_i))], \quad (2)$$

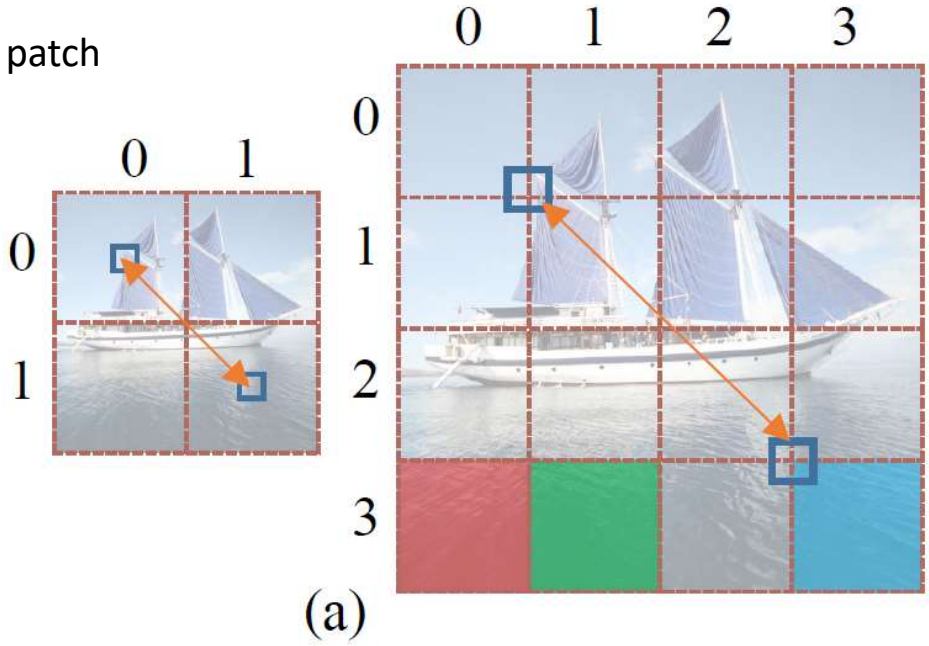
TRAIN AND TEST

CAPE: Content-Aware Positional Encoding

[Problems of Traditional Method]

1. Distance of red and cyan patch > Distance of red and green patch (in terms of embedding vectors)

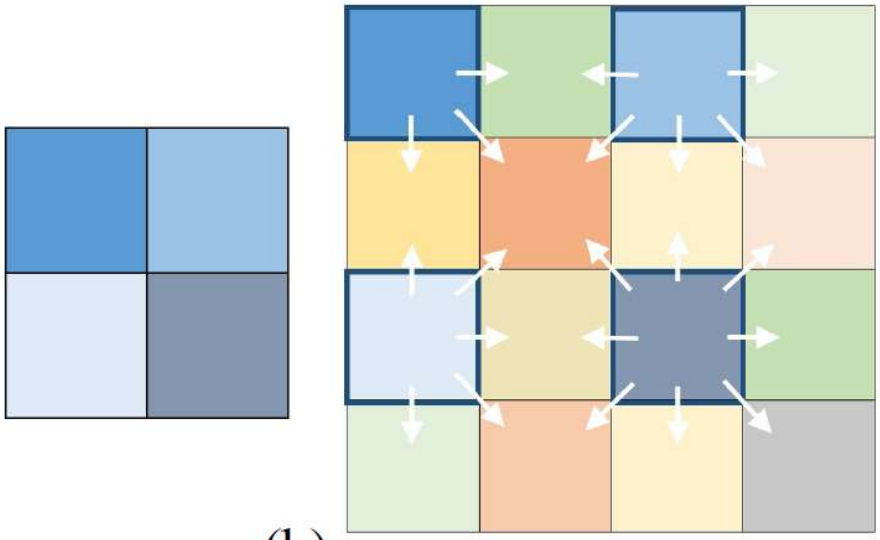
2. The positional Encoding using sin and cos does not fit well If the size of the input image changes.



TRAIN AND TEST

CAPE: Content-Aware Positional Encoding

For an image $I \in \mathbb{R}^{H \times W \times 3}$, we rescale the fixed $n \times n$ positional encoding to $\frac{H}{m} \times \frac{W}{m}$, as shown in Figure 3(b). In this way, various image scales will not influence the spatial relation between two patches.



(b)

TRAIN AND TEST

CAPE: Content-Aware Positional Encoding

[$n \times n$ CAPE for RGB image]

The CAPE of patch (x, y),
namely , $\mathcal{P}_{\mathcal{CA}}(x, y)$, is formulated as

$$\mathcal{P}_{\mathcal{L}} = \mathcal{F}_{\text{pos}}(\text{AvgPool}_{n \times n}(\mathcal{E})),$$
$$\mathcal{P}_{\mathcal{CA}}(x, y) = \sum_{k=0}^{s} \sum_{l=0}^{s} (a_{kl} \mathcal{P}_{\mathcal{L}}(x_k, y_l)), \quad (3)$$

where $\text{AvgPool}_{n \times n}$ is the average pooling function, $\mathcal{F}_{\text{pos}}$ is 1×1 convolution operation used as a learnable positional encoding function, $\mathcal{P}_{\mathcal{L}}$ is learnable PE following the sequence $\mathcal{E}$, n is set to 18 in our experiments, a_{kl} is the interpolation weight, and s is the number of neighboring patches. Lastly, we add $\mathcal{P}_{\mathcal{CA}i}$ to $\mathcal{E}_i$ as the final feature embedding of the i-th patch at a pixel location (x, y).

P_L = Positional encoding for ε

TRAIN AND TEST

Loss Functions

Style: I_s

Input: I_c

Output: I_o

The generated results should maintain the original content structures and the reference style patterns. Therefore, we construct two different perceptual loss terms to measure the content difference between the output image I_o and the input content image I_c, as well as the style difference between I_o and the input style reference I_s.

where $\phi_i(\cdot)$ denotes features extracted from the i-th layer in a pretrained VGG19 and N_l is the number of layers.

The content perceptual loss $\mathcal{L}_c$ is defined as

$$\mathcal{L}_c = \frac{1}{N_l}\sum_{i=0}^{N_l}\|\phi_i(I_o) - \phi_i(I_c)\|_2, \tag{9}$$

The style perceptual loss $\mathcal{L}_s$ is defined as

$$\begin{aligned}\mathcal{L}_s = &\frac{1}{N_l}\sum_{i=0}^{N_l}\|\mu(\phi_i(I_o)) - \mu(\phi_i(I_s))\|_2 \\ &+ \|\sigma(\phi_i(I_o)) - \sigma(\phi_i(I_s))\|_2,\end{aligned} \tag{10}$$

where $\mu(\cdot)$ and $\sigma(\cdot)$ denote the mean and variance of extracted features, respectively.

TRAIN AND TEST

Loss Functions

We also adopt identity loss [15] to learn richer and more accurate content and style representations. Specifically, we take two of the same content (style) images into StyTr^2, and the generated output $I_{cc}(I_{ss})$ should be identical to the input $I_c(I_s)$. Therefore, we compute two identity loss terms to measure the differences between $I_c(I_s)$ and $I_{cc}(I_{ss})$:

$$\mathcal{L}_{id1} = \|I_{cc} - I_c\|_2 + \|I_{ss} - I_s\|_2,$$

$$\mathcal{L}_{id2} = \frac{1}{N_l}\sum_{i=0}^{N_l}\|\phi_i(I_{cc}) - \phi_i(I_c)\|_2 + \|\phi_i(I_{ss}) - \phi_i(I_s)\|_2.$$

TRAIN AND TEST

Loss Functions

$$\mathcal{L} = \lambda_c \mathcal{L}_c + \lambda_s \mathcal{L}_s + \lambda_{id1} \mathcal{L}_{id1} + \lambda_{id2} \mathcal{L}_{id2}. \qquad (12)$$

TRAIN AND TEST

Experiment Results

TRAIN AND TEST

Experiment Results

TRAIN AND TEST

Experiment Results

TRAIN AND TEST

Experiment Results

TRAIN AND TEST

Experiment Results

	Ours	StyleFormer	IEST	AdaAttN	ArtFlow	MCC	MAST	AAMS	SANet	Avatar	AdaIN
$\mathcal{L}_c \downarrow$	**1.91**	2.86	<u>1.97</u>	2.29	2.13	2.38	2.46	2.44	2.44	2.84	2.34
$\mathcal{L}_s \downarrow$	<u>1.47</u>	2.91	3.47	2.45	3.08	1.56	1.55	3.18	**1.18**	2.86	1.91

TABLE II

QUANTITATIVE COMPARISONS. WE COMPUTE THE AVERAGE CONTENT AND STYLE LOSS VALUES OF RESULTS BY DIFFERENT METHODS TO MEASURE HOW WELL THE INPUT CONTENT AND STYLE ARE PRESERVED. THE BEST RESULTS ARE IN **BOLD** WHILE THE SECOND-BEST RESULTS ARE MARKED WITH AN <u>UNDERLINE</u>.

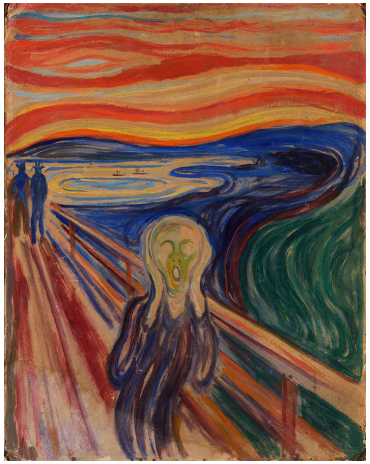
L_c : concept perceptual loss
L_s : style perceptual loss

TRAIN AND TEST

Inference Example

Input/Output

Style



TRAIN AND TEST

TRAIN AND TEST