Spatial Control in Model-Based Neural Style Transfer

Julian Bültemeier, Christoph-Alexander Holst, Volker Lohweg

inIT – Institute Industrial IT
Technische Hochschule Ostwestfalen-Lippe,
Campusallee 6, D-32657 Lemgo, Germany
E-Mail: {julian.bueltemeier, christoph-alexander.holst,
volker.lohweg}@th-owl.de

1 Introduction

Neural Style Transfer (NST) is an optimisation technique that combines two images – a content image and a style image. The result of the NST is an output image that looks like the content image but is rendered in the style of the style reference image [1]. An example of such a stylisation is shown in Figure 1. The application areas for such a method are manifold. Examples include image and video editing software [2, 3] and virtual reality applications [4].

The style transfer with the original formulation is a gradient-based optimisation procedure [1]. This procedure can take two to three minutes for the stylisation of a single image. For this reason, methods have been developed to speed up this process [5, 6, 7]. The idea of these model-based NST approaches is to train a feed-forward neural network to learn a direct conversion from a content image to an NST result image. After successful training, it is possible to perform the stylisation in a feed-forward pass without optimisation procedures. There are model-based NST methods that are limited to learn a single style [5, 6, 7], and there are Multi Style Transfer (MST) methods that can learn multiple styles [8, 9, 10]. However, although MST techniques can learn multiple styles simultaneously, they do not integrate spatial control or integrate it only after the style has been trained on complete images. The idea

DOI: 10.58895/ksp/1000151141-15 erschienen in:
Proceedings – 32. Workshop Computational Intelligence: Berlin, 1. - 2. Dezember 2022
DOI: 10.58895/ksp/1000151141 | https://www.ksp.kit.edu/site/books/m/10.58895/ksp/1000151141/

Figure 1: Example of a Neural Style Transfer with the content image (left), the style image used "*Wheat Field with Cypresses*" by Vincent van Gogh (centre) and the result of the NST (right).

of spatial control in NST is that images often consist of smaller segmentations that correspond to individual objects or further subdivisions. These regions have their own sub-styles [11]. This means that the style within an image differs in local regions. For this reason, NST procedures exist in which spatial control is integrated [11, 12]. Such a semantic style transfer makes it possible to explicitly assign the sub-style of a region in the style image to a region in the content image [11]. This control helps to improve the overall result of the style transfer. However, semantic style transfer has so far been studied mainly for optimisation techniques on a single image [11, 12].

The main contribution of this paper is therefore to integrate a spatial control into model-based NST. First, it is shown how existing MST procedures need to be adapted to be able to learn the sub-styles of a style image. In the second part, a concept is proposed to integrate the semantic transfer into the training in order to learn local style representations. In the final part, the proposed concept is evaluated using the intaglio style as an example. The intaglio style is created during intaglio printing and is an essential component on banknotes [13]. In a preliminary work in MEIER et al. it was shown [14] that the style in individual regions such as the *eye*, differs strongly from other regions. Therefore Intaglio Style Transfer (IST) has the potential to profit from a semantic style transfer.

2 Preliminaries

Humans have always been attracted and inspired by the art of painting. The imitation of certain styles require special skills of well-trained artists. Stylisation is a complex image processing task. A machine approach to this process

is described by the seminal work by GATYS et al. [1]. They used an encoder V created by the Visual Geometry Group (VGG) at Oxford University [15] to extract content and style representations from images. The result was a significant increase in performance in automatically creating new images with a given style, compared to traditional synthesis in pixel space [16] or in a hand-crafted feature space [17].

There are mainly two loss functions defined in the NST technique – the *content loss function* $\mathrm{L_C}$ and the *style loss function* $\mathrm{L_S}$ [1]. If the synthesised image $\mathbf{I}$ is to have the same content as the content image $\mathbf{I}_\mathrm{C}$, then the difference between deeper levels of an encoder V in the feature representation of those two images must be minimised. For this reason, the content loss is simply the mean squared error (MSE) of the features of the content image and the input image that are passed through the encoder V to the layer l_C [1]:

$$\mathrm{L_C}(\mathbf{I}, \mathbf{I}_\mathrm{C}) = \overline{\sum}(V^{l_C}(\mathbf{I}) - V^{l_C}(\mathbf{I}_\mathrm{C}))^2. \tag{1}$$

The grand sum of the following tensor divided by the number of elements, i. e. the grand mean, is denoted here by the overlined sum symbol without indices [18].

Various implementations do not always work equally well for different data. This is particularly reflected in the style. In NST, recurring patterns in the style image are analysed. This refers to structures and edges as well as colours and shapes [2]. One of the possible approaches is to compute different feature representation correlations of the encoder V [1]. Computation of the style loss $\mathrm{L_S}$ between the synthesised image $\mathbf{I}$ and the style image $\mathbf{I}_\mathrm{S}$ is thus similar to the content loss with the difference that features are not compared directly. Instead the MSE of a correlation measure g is used. The style loss $\mathrm{L}_{\mathrm{S},l_S}$ for a layer l_S thus results in [1]

$$\mathrm{L}_{\mathrm{S},l_S}(\mathbf{I}, \mathbf{I}_\mathrm{S}) = \overline{\sum}(g\Big(V^{l_S}(\mathbf{I})\Big) - g\Big(V^{l_S}(\mathbf{I}_\mathrm{S})\Big))^2, \tag{2}$$

where the total style loss $\mathrm{L_S}$ is the weighted sum of the selected layers $\mathrm{L_S}$ with [1]

$$\mathrm{L_S}(\mathbf{I}, \mathbf{I}_\mathrm{S}) = \sum_{l_S \in L_\mathrm{S}} \lambda_{l_S} \cdot \mathrm{L}_{\mathrm{S},l_S}(\mathbf{I}, \mathbf{I}_\mathrm{S}). \tag{3}$$

Using multiple layers captures style elements of varying detail. Whereby finer details are represented in the shallower layers and coarser details in the deeper layers [1].

GATYS et al. use the GRAM-Matrix as the correlation measure g of the generated feature map. The GRAM-Matrix of a flattened feature map $\mathbf{x}$ of an encoder V is calculated as follows [11]:

$$\text{gram} : \mathbb{R}^{N\times C} \rightarrow \mathbb{R}^{C\times C}, \quad \text{gram}(\mathbf{x}) = \mathbf{x}^T \cdot \mathbf{x}. \tag{4}$$

The images to be compared mostly have a different size. In order for the correlation measure g to be independent of this size, the GRAM-Matrix is divided by the number of elements within the spatial dimensions [1]:

$$g : \mathbb{R}^{H\times W\times C} \rightarrow \mathbb{R}^{C\times C}, \quad g(\mathbf{X}) = \frac{\text{gram}(\text{spatvec}(\mathbf{X}))}{H \cdot W}. \tag{5}$$

The function spatvec$(\cdot)$ is a special case of the vectorisation function. This is used to flatten the feature maps of the encoder V by reducing the spatial dimension of the feature maps to one [19].

Equation (1) for content loss and (3) for style loss are defined in the original formulation of GATYS et al. [1] to automatically identify the content and style of images that need to be merged in a final step for style transfer. The loss function L for the NST consists of a weighted sum of these loss functions and is defined as [1]

$$\text{L}(\mathbf{I}, \mathbf{I}_\text{C}, \mathbf{I}_\text{S}) = \lambda_\text{C} \cdot \text{L}_\text{C}(\mathbf{I}, \mathbf{I}_\text{C}) + \lambda_\text{S} \cdot \text{L}_\text{S}(\mathbf{I}, \mathbf{I}_\text{S}). \tag{6}$$

The weights λ_C and λ_S are hyperparameters and adjusted according to user preferences [1].

The final step in NST is the minimisation of the above mentioned loss function by solving an optimisation algorithm. The synthesis of a new image $\mathbf{I}$ is achieved by iteratively adjusting its pixel values as follows [1]:

$$\mathbf{I} = \arg\min_{\mathbf{I}} \text{L}(\mathbf{I}, \mathbf{I}_\text{C}, \mathbf{I}_\text{S}). \tag{7}$$

In summary, the original formulation of NST consists of a procedure in which an image $\mathbf{I}$ to be synthesised is transformed in an optimisation process. For this purpose, a loss function consisting of two components is determined. The content loss function L_C, which specifies how far the image $\mathbf{I}$ is from the content of the content image $\mathbf{I}_C$ and the style loss function L_S, which specifies how far the image $\mathbf{I}$ is from the style of the style image $\mathbf{I}_S$ [1].

3 Related Work

The seminal work of GATYS et al. [1], presented in the last section, describes the creation of artistic images by separating and recombining image content and style. Since the publication of this first NST approach in 2016 numerous improvements and developments have been published. The following overview is not exhaustive — for a more comprehensive technical overview, please refer to the work of JING et al. [21].

A part of NST work has focused on improving the transfer by various loss functions for style representations. There is no uniform definition for the style of an image. However, two different approaches exist to describe the style in NST [1, 22]. A stochastic approach – such as the GRAM-Matrix concept – assumes that if the global statistics match, the underlying style also matches [23]. A fundamentally different approach is that styles are described by regular or irregular compositions of small patches [23]. Such a structural patch-based approach has been proposed by LI and WAND [22] which also works on the features of an encoder V. Other methods focus on improving the results by including additional losses [24, 25] or a combination of stochastic and structural approaches [14, 24]. That such an image-based method can also be used to transfer the intaglio style has been demonstrated by MEIER et al. [14]. This IST algorithm enables the production of high-quality gravure prints for portrait images within a few hours.

The drawback of all these image optimisation procedures is the slow optimisation process described in (7) in which each individual image is stylised. For a more dynamic NST, these methods have been extended using the same loss function by training a feed forward Convolutional Neural Network (CNN) –

the transformer G – to perform the style transfer. The first to publish such a model-based approach were JOHNSON et al. [5] and ULYANOV et al. [6] whose approaches differ only in the structure of the transformer G. They were able to show that such a network can be trained on one style image and then stylise arbitrary content images in this style without optimisation procedure. However, this also means that a separate network must be trained for each style image. Therefore, other Multi Style Transfer (MST) approaches have investigated the possibility of learning several styles at the same time [8, 9, 10].

In CHEN et al. [10] filter banks consisting of several convolutional layers, each encoding one style, are used for the transformation. ZHANG and DANA [9] on the other hand introduce a *CoMatch* layer that matches feature statistics based on the given styles. In these approaches, a training process is still required to learn weights that are used for transformation. The advantage is that this transformation is adapted to a given style. Nevertheless, there are methods that are based on a more flexible transformation [8, 26, 27]. The basic idea of these approaches is that the statistics of the content features are directly adapted to the style features. DUMOULIN et al. [26] use an adaptive instance normalisation layer to determine the parameters for the transformation. This approach is extended by HUANG and BELONGIE to arbitrary images [27]. An alternative transformation to instance normalisation is a Whitening and Colouring Transformation (WCT) introduced by LI et al. [8], which achieves the transformation by uncorrelating the content features and correlating the features using the style statistics [8]. These approaches have the advantage of being able to perform an universal style transfer with little or no training. The disadvantage is that unlike the previous learnable MST approaches, the transformation cannot be changed by adjusting parameters. This means that if the result is different than expected, a different transformation must be used to change the result.

The images used consist of regions that correspond to different foreground objects and backgrounds or other segmentations. Often stylistic artefacts can arise which destroy the image content [11]. For this reason, spatial control is often integrated into NST to learn better semantic style representation. The feasibility of such spatial control for the reduction of stylistic artefacts has been demonstrated by CHAMPANDARD [12]. An approach that GATYS et al. [11]

have also used for a GRAM-Matrix-based optimisation. These works focusing on semantic style transfer have often been developed for the image optimisation process, but can also be applied to model-based NST approaches. HUANG and BELONGIE [27] show localised stylisation control in their network, while LI et al. [8] use the style/content relationship to control their feedforward networks. Other methods, on the other hand, only integrate spatial control after the training [10, 28]. This makes it possible to stylise the individual regions in different styles, but the actual sense of spatial control during training is lost.

Nevertheless, the drawback of the semantic style transfer is that it has been developed mainly for the optimisation techniques on a single image. If MST models are used, then these models integrate the segmentation only after the training. This means that the models are pre-trained without semantic distinction and can subsequently transfer only the globally learned style representations. The application of these global style is not suitable for the IST, as the intaglio style depends strongly on the respective regions. This means that the intaglio style in the eye, for example, is very different from the style in other regions. For this reason, in MEIER et al. the result could be improved by spatial control. However, such a control has not yet been implemented with the existing MST procedures. The next section therefore explains how existing methods can be modified to integrate a semantic style transfer into the MST.

4 Approach

This section is divided into three parts. First, the procedure of semantic style transfer in the optimisation techniques and the basic functioning of an MST transformer are explained. Subsequently, an approach is proposed with which a semantic control is integrated into the model-based NST to restrict the transformation to explicit regions. Finally, the architecture implemented in this paper based on an existing MST approach is presented.

4.1 Explicit Semantic Model-Based Style Transfer

The semantic control in NST approaches is achieved by segmenting out individual regions in the loss calculation. For this purpose, GATYS et al. [11] use spatially guided GRAM-Matrices for each region $r \in R$ in the synthesised image $\mathbf{I}$ and style image $\mathbf{I}_\mathrm{S}$, which are calculated by multiplying the feature maps by a mask $\mathbf{M}_r^{l_S}$. The corresponding GRAM-based regional style loss $\mathrm{L}_{\mathrm{S},l_S}$ can be formally calculated with [11]:

$$\mathrm{L}_{\mathrm{S},l_S}(\mathbf{I},\mathbf{I}_\mathrm{S}) = \overline{\sum}\sum_{r}^{R}(g\left(\mathbf{M}_{C,r}^{l_S} \circ V^{l_S}(\mathbf{I})\right) - g\left(\mathbf{M}_{S,r}^{l_S} \circ V^{l_S}(\mathbf{I}_\mathrm{S})\right))^2. \tag{8}$$

The $\circ$ represents an element-wise multiplication with each feature map. The GRAM-Matrix is calculated as before in (4), but is normalised by the area A_r of the mask $\mathbf{M}_r$ to reduce size differences in the masks. An adjusted formula for the GRAM-Matrix in (5) results in [11]:

$$g : \mathbb{R}^{H\times W\times C} \rightarrow \mathbb{R}^{C\times C}, \quad g(\mathbf{I}) = \frac{\mathrm{gram}(\mathrm{spatvec}(\mathbf{M_r} \circ \mathbf{I}))}{A_r}. \tag{9}$$

Semantic style transfer is thus achieved by masking individual areas. This makes it possible to restrict the transfer of style to user-defined regions in both the content image and the style image. An example of an improvement through the spatial control can be seen in Figure 2. Such a semantic distinction can also be used for the model-based MST approaches to restrict the transformation to masked regions. In order to propose a possible approach for this distinction, the general synthesis of an MST is first defined.

The synthesis with a model-based NST or transformer G architecture is shown in Figure 3. Conceptually, the architecture consists of three components [9]:

1. The *encoder part* E is applied for feature extraction and dimension reduction to perform transformation on features.
2. The *transformation part* T is used to transform the features in a style.
3. The *decoder part* D consists of convolutional layers that enlarge the input and recreates an output image from the transformed features.

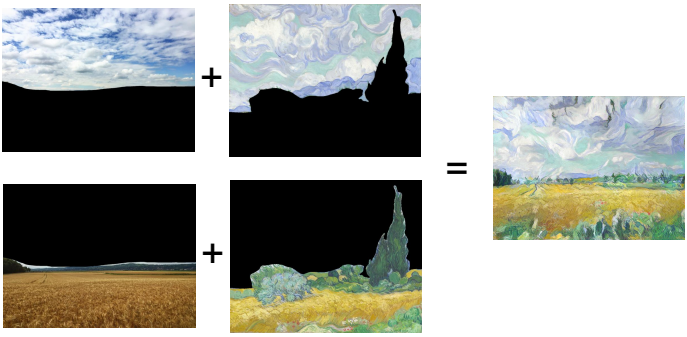


Figure 2: Example of the semantic NST with an explicit transfer for the region *sky* and region *wheatfield*. It can be seen that the result of the semantic style transfer has fewer stylistic artefacts compared to Figure 1, such as the green artefacts in the clouds.

A general synthesis of the transformer G for a content image $\mathbf{I}_\mathrm{C}$ and a style image $\mathbf{I}_\mathrm{S}$ can thus be formulated as:

$$G(\mathbf{I}_\mathrm{C}, \mathbf{I}_\mathrm{S}) = \mathrm{D}(\mathrm{T}(\mathrm{E}(\mathbf{I}_\mathrm{C}), \mathrm{E}(\mathbf{I}_\mathrm{S}))). \tag{10}$$

For a semantic style transfer based on the transformer G in MST, the masks are normally integrated after the training in order to be able to mask the respective areas and to perform the selected transformation T_i in the individual areas. A synthesis with two different style images $\mathbf{I}_{\mathrm{S},1}$ and $\mathbf{I}_{\mathrm{S},2}$ on the basis of a MST transformer G can thus be carried out as:

$$G(\mathbf{I}_\mathrm{C}, \mathbf{I}_{\mathrm{S},1}, \mathbf{I}_{\mathrm{S},2}, \mathbf{M}_C) = \mathrm{D}\left(\sum_{i=1}^{2} \mathrm{T}_i(\mathbf{M}_{i,C} \circ \mathrm{E}(\mathbf{I}_\mathrm{C}), \mathrm{E}(\mathbf{I}_{\mathrm{S,i}}))\right). \tag{11}$$

The transformation T_i has been trained to perform stylisation for the corresponding style images $\mathbf{I}_{\mathrm{S},1}$ and $\mathbf{I}_{\mathrm{S},2}$. This makes it possible to stylise the areas masked with the content masks $\mathbf{M}_C$ in the different styles. However, different sub-styles in a single style image cannot be taken into account in this way.

Therefore, this contribution proposes the implementation of explicit semantic mapping of the transformation. This means that, similar to the spatial control in semantic style transfer, the transformation $\mathrm{T}_{\mathbb{S},r}$ is performed using masks

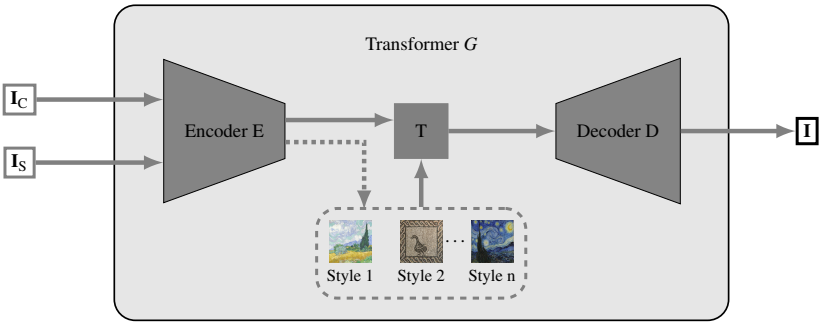


Figure 3: Conceptual structure of an MST transformer G. The structure consists of the encoder part E, transformation part T, and the decoder part D. As an example, three style images are shown for which a stylisation can be carried out with this MST transformer.

for regions $r \in R$ with a certain label. The schematic structure of the explicit semantic transformer is shown in Figure 4. An extended formula for (10) of an explicit semantic transformer $G_{\mathbb{S}}$ is given by

$$G_{\mathbb{S}}(\mathbf{I}_{\mathrm{C}}, \mathbf{I}_{\mathrm{S}}, \mathbf{M}_C, \mathbf{M}_S) = \mathrm{D}\left(\sum_{r}^{R} \mathrm{T}_{\mathbb{S},r}\left(\mathbf{M}^{\mathrm{E}}_{C,r} \circ \mathrm{E}(\mathbf{I}_{\mathrm{C}}), \mathbf{M}^{\mathrm{E}}_{S,r} \circ \mathrm{E}(\mathbf{I}_{\mathrm{S}})\right)\right). \tag{12}$$

Thus, in addition to the content image $\mathbf{I}_{\mathrm{C}}$ and the style image $\mathbf{I}_{\mathrm{S}}$, the semantic transformer $G_{\mathbb{S}}$ also receives the masks $\mathbf{M}$ for all regions $r \in R$ of the content and style images. This masks are used to explicitly distinguish the transformation in the feature space. Or in other words, the transformation $\mathrm{T}_{\mathbb{S},r}$ is performed only for regions with the same label in the content and style image. The fusion of the individual regions can be accomplished by a simple sum while maintaining the same dimension.

This label-based distinction of regions and the application of the transformation only to the masked features of the encoder E results in the explicit semantic style transfer. Furthermore, the transformer will only be able to learn the local style representations of the sub-style.

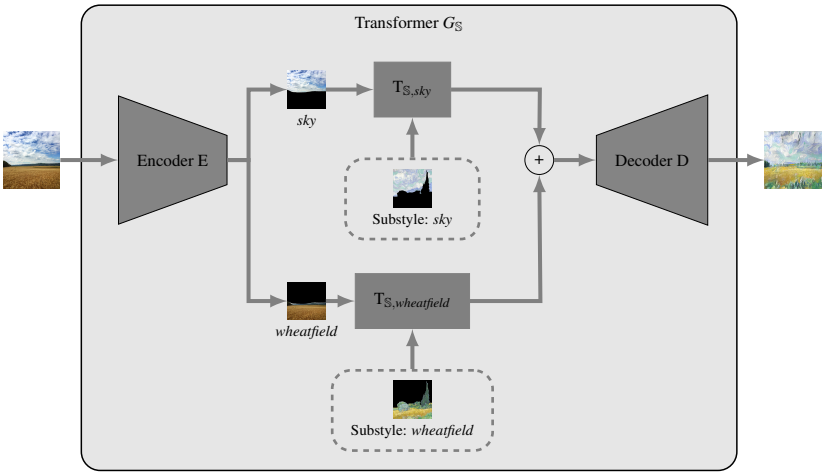


Figure 4: Conceptual structure of a semantic transformer $G_{\mathbb{S}}$. The structure consists of the components encoder E, decoder D and the semantic transformation parts $T_{\mathbb{S},r}$ for the regions $r \in \{sky, wheatfield\}$. The transformation parts are trained to transfer the respective masked sub-styles of the style image $\mathbf{I}_S$ by Vincent van Gogh, "*Wheat Field with Cypresses*".

4.2 Transformer Architecture

The approach presented in the last section can be implemented on the basis of existing methods by adding the semantic transformation part. In this paper, the generative multi-style network (MSG-Net) architecture of ZHANG and DANA [9] is applied. The reason for this selection is that it can be shown that the semantic approach can be integrated into existing approaches. On the other hand, this approach can be learned. Therefore, the transformation does not have to be selected or adapted for each sub-style as in universal style transfer [8, 27].

ZHANG and DANA use a residual block architecture for the MST that can learn the transformation for multiple styles simultaneously [9]. Each residual block contains a branch that leads to a series of convolutional blocks whose outputs are added to the input $\mathbf{X}$ of the block. For a reduction (downsampling) and increase (upsampling) of the input dimension, additional convolutional layers

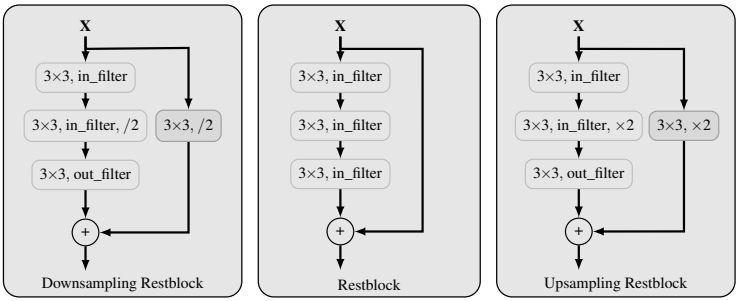


Figure 5: Schematic representation of the residual blocks used in this work for the transformer architecture. The downsampling residual block reduces the input dimension by half, the residual block maintains the input dimension, and the upsampling residual block increases the input dimension by twice. (Based on Figure 5 in [9]).

are used in the direct branch in MSG-Net. A general representation of the residual blocks used in this work is shown in Figure 5.

This architecture allows the layers to learn modifications of the identity mapping rather than the entire transformation, which has been shown to be beneficial for deep neural networks [29]. Each residual block consists of several convolutional blocks. These convolution blocks consist of the following three components [29]:

1. an instance normalisation layer [25],
2. a Rectified Linear Unit (ReLU) Activation Function [9], and
3. a convolutional layer.

For the intaglio style, an adjustment must also be made. The reason for this is that this style is binary, i.e. black or white. Therefore, for simplicity, the intaglio style is treated as greyscale-image in this work. For this purpose, the transformer is adapted accordingly by setting the input and output dimension of the transformer to one, in contrast to the RGB images with three dimensions.

4.2.1 Encoder and Decoder

The encoder E and the decoder D are symmetrically constructed. The encoder consists of an input convolutional layer with a kernel size of 7×7 and 32 filters, followed by two downsampling residual blocks. The decoder consists of two upsampling residual blocks and one output convolutional block with a kernel size of 7×7 [9].

4.2.2 Transformation

The transformation part T is the core element of MSG-Net. The transformation is achieved with the help of a *CoMatch* layer and several residual blocks [9]. In this layer, the GRAM-Matrix of the style is computed at runtime. A weight matrix ω is used to adjust the content features based on the style. The transformation of such a *CoMatch* layer can be defined as [9]:

$$\mathrm{T}(\mathbf{I}_\mathrm{C}, \mathbf{I}_\mathrm{S}) = \mathrm{spatvec}^{-1}\Big(\mathrm{spatvec}(\mathrm{E}(\mathbf{I}_\mathrm{C}))^T \omega g(\mathrm{E}(\mathbf{I}_\mathrm{S}))\Big)^T.$$

For the semantic transformer $G_\mathbb{S}$, a semantic transformation $\mathrm{T}_{\mathbb{S},r}$ is integrated for each region $r \in R$. For this purpose, the features from the encoder E are masked from both the style image $\mathbf{I}_\mathrm{S}$ and the content image $\mathbf{I}_\mathrm{C}$ by the respective masks $\mathbf{M}_{C,r}$ and $\mathbf{M}_{S,r}$. This results in the transformation $\mathrm{T}_{\mathbb{S},r}$ learning and applying only local style representations of the sub-style in region r. A semantic transformation with an *CoMatch* layer can thus be defined as:

$$\mathrm{T}_{\mathbb{S},r}(\mathbf{I}_\mathrm{C}, \mathbf{I}_\mathrm{S}) = \mathrm{spatvec}^{-1}\Big(\mathrm{spatvec}(\mathbf{M}_{C,r} \circ \mathrm{E}(\mathbf{I}_\mathrm{C}))^T \omega g(\mathrm{E}(\mathbf{M}_{S,r} \circ \mathbf{I}_\mathrm{S}))\Big)^T.$$

Each transformation $\mathrm{T}_{\mathbb{S},r}$ consists of a *CoMatch* layer followed by three residual blocks with 128 channel. Furthermore, the *CoMatch* layer is adapted to the use of masks by normalising the GRAM-Matrix by the area A_r of the mask, as shown in (9). This serves to reduce intensity differences due to different mask sizes and thus intensity artefacts [11].

5 Evaluation

This section is divided into two subsections. At first, the dataset and the methods utilised throughout this section are described[1]. Secondly, the results of the trained semantic transformer are shown. Since there is no objective measure of comparison for the evaluation, this part is only of qualitative nature.

5.1 Dataset

The dataset *CelebAMask-HQ* [30] contains $30,000$ portrait images from the larger portrait dataset *CelebA* [31]. For each of these images, there are up to 19 of labelled masks for all facial components and accessories, such as *hair*, *eyes*, *earring* and *cloth*, which are additionally split into left and right side of the face [30].

For semantic transfer with a transformer $G_{\mathbb{S}}$, such a division into 19 masks is not necessary. Furthermore, the style will not differ significantly when the left or right component is considered [32]. For this reason, the number of masks is reduced for this work. In Table 1 this classification is shown with the corresponding segmentation masks from the *CelebAMask-HQ* dataset. This twelve mutually exclusive binary masks are used for the training, whereby individual regions, such as *'headwear'*, *'accessories'* or *'glasses'* are optional. A script for the conversion of the masks is available in the implementation.

5.2 Methodology

The implementation of the described transformer architecture has been implemented within the *PyTorch* framework [33], using the NST library *pystiche* [34]. This library allows to implement approaches for NST without much prior knowledge about NST and Machine Learning (ML).

[1] The source code to reproduce the results is published under `https://github.com/jbueltemeier/masterthesis_bueltemeier`

Table 1: Listing of the reduced number of segmentations and the corresponding labels from the *CelebAMask-HQ* dataset.

Training segmentation label	*CelebAMask-HQ* segmentation label
'background'	'background'[1]
'skin'	'skin', 'neck'
'nose'	'nose'
'glasses'	'eye_g'
'eye'	'l_eye', 'r_eye'
'brows'	'l_brow', 'r_brow'
'ears'	'l_ear', 'r_ear'
'lips'	'mouth', 'u_lip', 'l_lip'
'hair'	'hair'
'headwear'	'hat'
'accessoire'	'ear_r', 'neck_l'
'body'	'cloth'

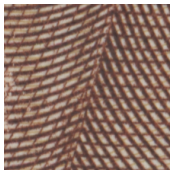
[1] Corresponds to the image pixel that is not covered by the segmentation masks in *CelebAMask-HQ*.

An implementation of the MSG-Net model training is implemented once with the proposed semantic differentiation in the transformer and once without. These two procedures can be described as follows:

1. $\mathrm{E} \rightarrow \mathrm{T} \rightarrow \mathrm{D}$ transformer G and
2. $\mathrm{E} \rightarrow \mathrm{T}_{\mathbb{S}} \rightarrow \mathrm{D}$ semantic transformer $G_{\mathbb{S}}$.

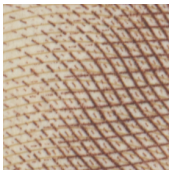
The first procedure without semantic control only involves one transformation for the style image used. The second method performs the transformation of the transformer using the approach proposed in (12), whereby the transformation is performed for all possible areas $r \in R$.

The intaglio motifs required for the style images have been cut out from high-resolution scans of banknotes and converted into greyscale. The scans were provided by *Koeing & Bauer Banknote Solutions*. The style images used in this work are shown in Figure 9 in the appendix. Each image is resized within the *pystiche* library with a bilinear interpolation to a size of 512 on the short side.



Substyle *body*

Substyle *hair*



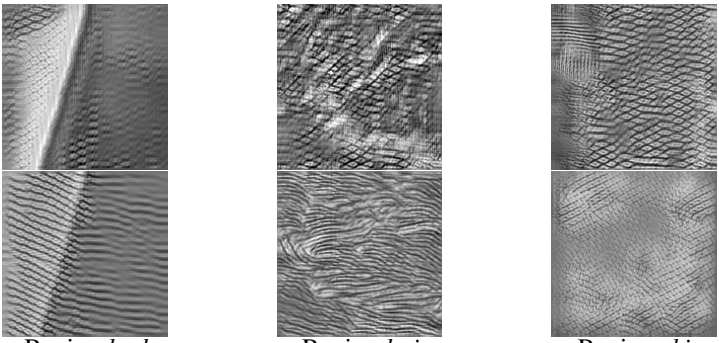
Substyle *skin*

Figure 6: Different sections of the *UAH020 S1997* Banknote are shown (Courtesy of *Koeing & Bauer Banknote Solutions*). There are clear differences in the shapes used between the individual images, each representing its own sub-style.

The transformers are trained with a batch size of 1 for 30000 iterations and the optimisation is performed by the Adam algorithm [35] with a learning rate of 10^{-4} [33]. For the calculation of the content loss and the style loss, the 19-layer VGG network [15] is adapted for the feature extraction of the encoder E. The weights and the necessary preprocessing from the *Caffe* framework[36] are used for this encoder. The style loss of the transformer G is calculated with (2) and the transformer $G_{\mathbb{S}}$ with (8). The feature extraction is performed in shallower layers of the encoder V than in the original papers because the intaglio structures are very fine. The layer $l_C = \texttt{relu\_2\_2}$ has been used for content loss and the layers $l_S \in L_S = \{\texttt{relu\_1\_1}, \texttt{relu\_2\_1}, \texttt{relu\_3\_1}\}$ for the style loss. The weights of the individual layers in the style loss are calculated by $\lambda_{l_S} = 1/n_{l_S}^2$, where n_{l_S} denotes the number of channels on the layer l_S [11]. The hyperparameters for the weighting of the loss functions from (6) have been chosen empirically. They correspond to $\lambda_C = 1$ for the content loss and $\lambda_S = 10^1$ for the style loss of the transformer G and the transformer $G_{\mathbb{S}}$.

5.3 Intaglio Style Transfer

The attention to detail and the different ways in which the engraver can engrave the print design distinguish the intaglio style [37]. Typical features of engraving are lines and shading by cross-hatching, where overlapping lines create darker areas [37]. These possibilities lead to differences in style. This is illustrated in Figure 6.



Region *body* Region *hair* Region *skin*

Figure 7: Style transfer of a transformer G (top) and transformer $G_{\mathbb{S}}$ (bottom) for the regions *body*, *hair* and *skin* of a portrait image.

The fact that a transformer G is not capable of learning such a distinction is particularly evident in a detailed view. For this purpose, stylised image sections for the regions *hair*, *skin* and *body* can be seen in the Figure 7. It can be seen that the results of the transformer G do not differ in all three areas. In contrast, the results of the semantic transformer $G_{\mathbb{S}}$ show that these transformers are able to learn and transfer a semantic distinction of the sub-styles. The difference is particularly noticeable in the *hair* sub-style with the wavy lines compared to the diamond pattern in the other two detail images.

These differences in the sub-styles in the regions can also be found in the stylisation of complete portraits. Figure 8 shows an example of this. To emphasise the intaglio structures in the portrait, the background in the result images is displayed uniformly in white. Overall, it shows that both the transformer G and the semantic transformer $G_{\mathbb{S}}$ can be used to create high-quality intaglio images. However, the transfer of the different sub-styles in the semantic transformer $G_{\mathbb{S}}$ lead to an improvement of the results in terms of the intaglio style.

Nevertheless, a disadvantage of the semantic transformer $G_{\mathbb{S}}$ is that transition areas between the regions arise. This effect can be seen especially at the hairline in the result image of the semantic transformer $G_{\mathbb{S}}$ in Figure 8. This makes this result appear qualitatively inferior compared to the transformer G. To reduce this effect, it makes sense to create a binary edge between the

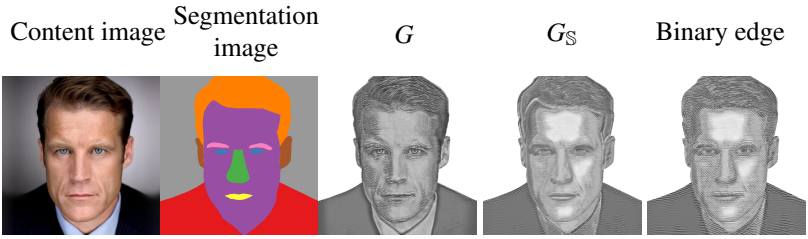


Figure 8: Result of a stylisation of a complete content image (left) with the segmentation used for this, which marks the different regions in colour. From left to right, the result of a transformer G, a semantic transformer $G_{\mathbb{S}}$, and the result with a binary edge between the regions is shown.

individual regions. The reason for this is that with very different sub-styles, no transitional area is expected anyway. This binary edge can be achieved by the semantic transformer $G_{\mathbb{S}}$ stylising the respective regions separately one after the other and merging them in pixel space. The result of such a stylisation is shown in Figure 8 on the right. The result shows that the transition areas can be reduced and the overall result is better.

6 Conclusion and Outlook

This paper has proposed an approach for a semantic treatment of sub-styles in a model-based NST. It is shown that the semantic treatment of a style image leads to an improvement of the result. This is because with a semantic transformer $G_{\mathbb{S}}$, in contrast to a transformer G without semantic transfer, it is possible to transfer the different sub-styles to certain regions in the content image. The semantic transformer is thus able to produce high quality intaglio images.

However, there are still questions to be answered in future work:

1. An assumption for simplification is that the intaglio image is greyscale. This means that there are transitions between black and white. This is not the case in practice. For this reason, a binary post-processing must take place in order not to complicate the machine readability of the Intaglio style.

2. It has been shown that with the intaglio style, edge artefacts occur between different regions. Therefore, merging the regions at pixel space has been investigated to improve the results. These regions should be further investigated. In this context, it is also important to check for which regions masking is necessary at all. However, there is still a lack of a suitable comparison measure.

3. In addition, a single style image has initially been taught for each region. With the implemented approach, it is also possible to learn several sub-styles for a region at the same time. Learning several sub-styles can be used to improve the overall result. For example, it is helpful to use a style image with long hair for long hair and one with short hair for short hair.

Acknowledgment

This contribution is part of the project *Fused Security Features*, which is funded by the *Ministry for Culture and Science of North Rhine-Westphalia* (MKW NRW) under the Grant ID 005-1703-0013. The authors thank the *Koeing & Bauer Banknote Solutions* for supplying high-resolution scans of various banknotes that were utilised as style images for the IST.

7 Template Images

References

[1] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[2] Andrew Glassner. Deep Learning: A Visual Approach. *No Starch Press*, 2021.

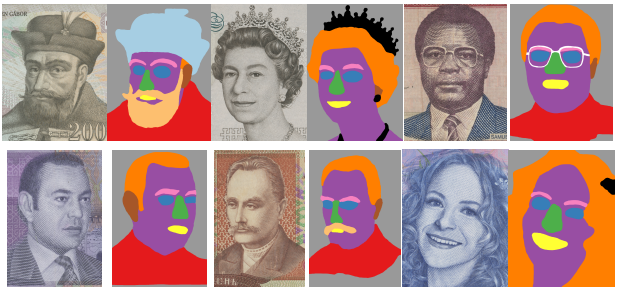


Figure 9: Overview over the Intaglio portraits and corresponding segmentation, which are used as style images within the IST. From left to right and top to bottom, the portraits were extracted from the following banknotes: *HUF2000 S2007*, *GBP005 S2002*, *LRD50 S2008*, *MAD050 S2002*, *UAH020 S1997* and *Specimen02*.

[3] Turn Photos From Simple To Sublime With The Style Transfer Tool. *Picsart*, 2022. Online. Retrieved: 15/08/2022. `https://picsart.com/style-transfer`.

[4] Piotr Bojanowski, David Lopez-Paz, Hervé Jegou, and Antoine Bordes. Using AI for new visual storytelling techniques in VR. *Meta*, 2017. Online. Retrieved: 15/08/2022. `https://engineering.fb.com/2017/07/26/virtual-reality/using-ai-for-new-visual-storytelling-techniques-in-vr/`.

[5] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In: *European Conference on Computer Vision (ECCV)*, 2016.

[6] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor S. Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. *Computing Research Repository (CoRR)*, 2016.

[7] Artsiom Sanakoyeu, Dmytro Kotovenko, Sabine Lang, and Björn Ommer. A style-aware content loss for real-time hd style transfer. In: *European Conference on Computer Vision (ECCV)*, 2018.

[8] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal Style Transfer via Feature Transforms. *Neural Information Processing Systems (NIPS)*, 2017.

[9] Hang Zhang and Kristin Dana. Multi-style Generative Network for Real-time Transfer. *ArXiv*: `abs/1703.06953`, 2018.

[10] Dongdong Chen, Lu Yuan, Jing Liao, Nenghai Yu, and Gang Hua. StyleBank: An Explicit Representation for Neural Image Style Transfer. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[11] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. Controlling perceptual factors in neural style transfer. *Computing Research Repository (CoRR)*, 2017.

[12] Alex J. Champandard. Semantic Style Transfer and Turning Two-Bit Doodles into Fine Artworks. *ArXiv*: `abs/1603.01768`, 2016.

[13] Rudolph L. van Renesse. *Optical Document Security*. Artech House, 3. edition, 2005.

[14] Philip Meier, Julian Bültemeier, Volker Lohweg, Helene Dörksen, and Johannes Schaede. Intaglio Style Transfer – Partially Automating the Intaglio Image Creation Process. *Conference on Optical Document Security (ODS)*, 2020.

[15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *Computing Research Repository (CoRR)*, 2014.

[16] Alexei A. Efros and Thomas K. Leung,. Texture synthesis by non-parametric sampling. In: *IEEE International Conference on Computer Vision (ICCV)*, 1999.

[17] Javier Portilla and Eero P. Simoncelli. A Parametric Texture Model Based on Joint Statistics of Complex Wavelet Coefficients. *International Journal of Computer Vision (IJCV)*, 2000.

[18] Brian S. Everitt and Anders Skrondal. *The Cambridge dictionary of statistics*, 2010.

[19] Leslie Hogben. Handbook of Linear Algebra *Chapman & Hall / CRC*, 2007.

[20] Ethem Alpaydin. Introduction to Machine Learning. 4th Edition, *MIT Press*, 2020.

[21] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. Neural style transfer: A review. *IEEE Transactions on Visualization and Computer Graphics*, 2019.

[22] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In: *European Conference on Computer Vision (ECCV)*, 2016.

[23] Dongxiao Zhou. *Texture analysis and synthesis using a generic Markov-Gibbs image model*. PhD thesis, University of Auckland, 2006.

[24] Eric Risser, Pierre Wilmot, and Connelly Barnes. Stable and controllable neural texture synthesis and style transfer using histogram losses. *Computing Research Repository (CoRR)*, 2017.

[25] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Instance Normalization: The Missing Ingredient for Fast Stylization. *ArXiv*: `abs/1607.08022`, 2016.

[26] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A Learned Representation For Artistic Style. *ArXiv*: `abs/1610.07629`, 2016.

[27] Xun Huang and Serge J. Belongie. Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization. *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[28] Zixuan Huang, Jinghuai Zhang, and Jing Liao. Style Mixer: Semantic-aware Multi-Style Transfer Network. *Computer Graphics Forum*, 2019.

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[30] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[31] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep Learning Face Attributes in the Wild. In: *International Conference on Computer Vision (ICCV)*, 2015.

[32] Ahmed A. S. Seleim, Mohamed A. Elgharib, and Linda Doyle. Painting style transfer for head portraits using convolutional neural networks. *ACM Transactions on Graphics (TOG)*, 2016.

[33] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In: *Proceedings of the NIPS Autodiff Workshop*, 2017.

[34] Philip Meier and Volker Lohweg. pystiche: A Framework for Neural Style Transfer. *Journal of Open Source Software (JOSS)* `10.21105/joss.02761`, 2020.

[35] Diederik P. Kingma and Jimmy Lei Ba. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations (ICLR)*, 2015.

[36] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *Computing Research Repository (CoRR)*, 2014.

[37] Adrianus C. J. Stijnman. A history of engraving and etching techniques: developments of manual intaglio printmaking processes, 1400-2000 *Archetype Publications*, London 2012. Online `http://hdl.handle.net/11245/1.378167`.