Neural Style Protection: Counteracting Unauthorized Neural Style Transfer

Yaxin Li *
Michigan State University
liyaxin1@msu.edu

Jie Ren*
Michigan State University
renjie3@msu.edu

Han Xu
Michigan State University
xuhan1@msu.edu

Hui Liu
Michigan State University
liuhui7@msu.edu

Abstract

Arbitrary neural style transfer is an advanced AI technique that can effectively synthesize pictures with an artistic style similar to a given source picture. However, if such an AI technique is leveraged by unauthorized individuals, it can significantly infringe upon the copyright of the source picture's owner. In this paper, we study how to protect the artistic style of source images against unauthorized style transfer by adding imperceptible perturbations to the original source pictures. In particular, our goal is to disable the neural style transfer models from producing high-quality pictures with a similar style to the source pictures with slight manipulating the source images. We introduce Neural Style Protection (NSP), which provides protection for source images against various neural style transfer models. Through extensive experiments, we demonstrate the effectiveness and generalizability of the proposed style protection algorithm across numerous style transfer models using varied metrics.

1. Introduction

Recently, neural style transfer techniques [10, 11, 15, 19, 20] have been developed to extract the artistic style from a given ***source image*** and transfer it to generate a picture with the content from a ***content image***. Among these methods, Arbitrary Neural Style Transfer (ANST) [5, 12, 15, 17, 25, 49], one of the most prevalent and flexible techniques, can easily transfer the style from any image to the content image by a single model without the need of fine-tuning the style transfer models. However, if these techniques are leveraged by unauthorized people to synthesize artistic works without permission, it will raise huge concerns in terms of artwork's copyright. For example, ANST can easily imitate any painting of a specific style that would cost an artist several days or even months. These will lead to a severe infringement on the copyright of the original artwork. This scenario raises the urgent need to protect the style of a piece of artwork from being copied by neural style transfer techniques.

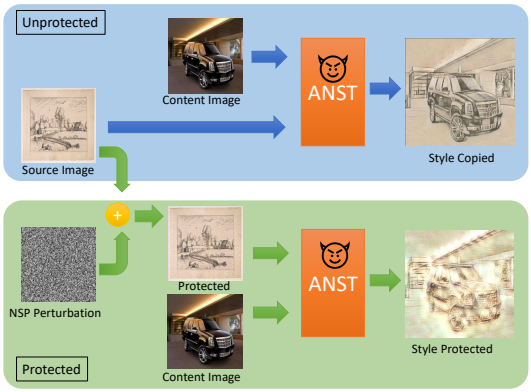

Figure 1. ANST and Neural Style Protection

In this paper, we propose protecting the copyright of the source images against neural style transfer techniques by leveraging the concept of adversarial examples [27, 36, 41, 45]. As shown in Figure 1, we propose to protect the source image style by introducing imperceptible perturbations to the source image, based on which ANST could not generate similar images of high quality. However, there is a major challenge for us to directly adopting adversarial examples and successfully prevent unauthorized neural style transfer. Adversarial examples [2, 13] are usually calculated only based on one specific fixed model. Thus, they lack generalization ability. However, the style of a painting could be copied by various ANST models. The painting is possibly imitated by different ANST models, and the artists cannot control what method others will use to imitate the style in practice. It is possible that the generated perturbation against one ANST model will totally lose its effectiveness if another ANST is adopted for style transfer. Thus, the generaliza-

*Equal contribution.

tion of protection to different ANST models is desired. In other words, the protection should be designed to reduce the effectiveness of various ANST models.

In our work, we propose a novel method, Neural Style Protection (NSP), to demonstrate the potential for successfully safeguarding source images against various ANST models. We mainly focus on the encoder-decoder based ANST, which represents the mainstream category of ANST models [4, 7, 15, 23, 25, 31, 38, 49]. Specifically, we designed two strategies for NSP to enhance the generalization and provide general protection. **First**, instead of directly targeting the final transferred content images on a different style, we alter the intermediate style representation. ANST models extract the style representation from an encoder and transfer it onto content images with a decoder. In general, decoders of ANST are far more diverse than encoders among ANST methods, therefore setting the intermediate style representation as a perturbation objective can mitigate the impact of the diverse decoders on the generalization. **Second**, although the encoders are similar, they have different configurations like fine-tuned parameters [49], methods of feature extraction [25, 31, 38] and so on. To further boost the generalization [8, 26, 39, 44], we use a momentum-based ensemble method to accommodate different encoders adopted by ASNT models. With these two strategies, we can significantly improve the generalization of NSP across ANST models. The main contributions of this paper are as follows:

- We propose a method named NSP to protect the style of images against ANST. This approach addresses the concerns of image generators that their novelty style in the artworks can be readily replicated by neural style transfer techniques.
- We design two strategies: altering intermediate style representation and a momentum-based ensemble method, to enhance the generalization across ANST models and consequently provide a full protection.
- We conduct experiments to demonstrate that our method can protect the style of images from being copied by various ANST models.

2. Related Work

Arbitrary Neural Style Transfer. Neural style transfer [18] aims to model and extract style information from a specific artwork and apply the style to a reconstructed content image. Traditional style transfer methods include optimization based style transfer methods [28–30]and non-parametric neural model approaches [21]. However, these methods require relatively high computational costs due to the complex optimization process compared to later techniques that based on neural networks. Those neural network based technologies range from transferring a specific style type from one model [20, 37] to multiple styles generated from a single model [5, 12, 15, 17, 25, 49]. Arbitrary style transfer [3, 9, 22, 47], which refers to using a single model to generate all given types of style images, is a flexible and efficient approach to achieve the style transfer in different real world scenarios.

Many encoder-decoder based ANST models have been proposed, which are the mainstream category. Adaptive instance normalization [15] is one of the first methods to achieve arbitrary style transfer. It extracts the feature of an image by a pre-trained encoder, then simply aligns the mean and variance of the content feature with those of the style feature so that the content feature shares the same distribution with the target style feature. A learnable decoder is trained to reconstruct the feature to style transfer image. Later on, feature statistic adaptation becomes a unified model to handle arbitrary style transfer tasks. One line of work focuses on improving the local transformation performance thus achieves better balance of style transformation and less content distortion [25, 31, 34]. A representative solution is adopted the widely used attention mechanism due to its ability to model the correspondence among local features of the input content image and target style image. Style-Attentional Network (SANet) [31] is a learnable soft-attention-based network to model the semantic correlations between the content features and the style features, and match the style feature to the content with closest semantic meaning. AdaAttN uses both shallow layer feature and deep layer feature to get attention score and calculate element-wise adjusted mean and variance map based on the attention score compared to originally content feature simply conduct channel-wise shift and re-scale to align the statistics of style feature [25]. CAST fine-tuned the encoder to generate better style encoding [49]

Adversarial Examples. Adversarial Examples are proposed to attack the deployed machine learning models in the test phase [2, 13, 16, 27, 43]. It can be divided into white-box attacks, which have full access to the target model, and block-box attacks, which have no access to the target model [46]. In while-box attacks, PGD attack proposed by Madry et al. [27] has been widely spread for its effectiveness and invisibility. It optimizes the perturbations on the data to reduce the attack loss as follows,

$$\min_{\boldsymbol{\delta}} \mathcal{L}_{attack}(f(\boldsymbol{x}+\boldsymbol{\delta}), y) \text{ s.t. } \|\boldsymbol{\delta}\|_\infty \leq \epsilon, \tag{1}$$

where $\boldsymbol{\delta}$ is the changed perturbation added on the image data $\boldsymbol{x}$, ϵ constraints the change of the data to be invisible and (x, y) is the input data and its label. PGD achieves the minimization by updating the perturbation in an iterative way,

$$\boldsymbol{\delta}_{t+1} = \text{clip}_{(-\epsilon,\epsilon)} \left\{ \boldsymbol{\delta}_t - \alpha \cdot \text{sign}\left(\nabla_{\boldsymbol{\delta}_t} \mathcal{L}_{attack}(f(\boldsymbol{x}+\boldsymbol{\delta}_t), y)\right) \right\}, \tag{2}$$

where $\text{sign}(\cdot)$ keeps the sign of each value of the gradient. The methods based on PGD attack have inspired other

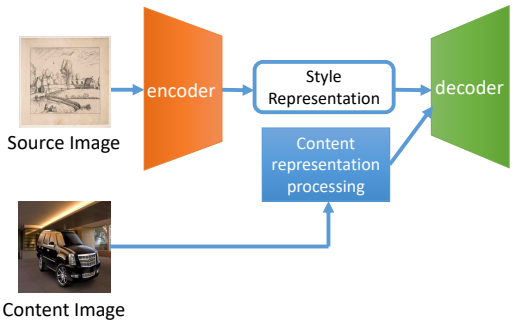

Figure 2. A General Framework of ASNT

applications like Unlearnable Examples [14], poisoning attack [32] and so on. The development of adversarial attack has also aroused the research in adversarial defenses which promoted the safety and robustness of DNNs [33, 42]. In this work, we also use PGD as our optimization method to perturb our image data for protection against ANST.

3. Method

In this section, we introduce the details of the proposed NSP. We first show the process of ANST and define the protection problem in Section 3.1. Then we discuss how to enhance the generalization across models by altering the intermediate style representation in Section 3.2 and how to further assemble different encoders across different ANST models in Section 3.3.

3.1. Problem Statement

In this subsection, we first provide the general framework of the whole process of the widely used encoder-decoder based ANST, which has both good performance and fast inference, and then discuss the challenges we face on the protection against the ANST models.

ANST. The game of ANST and its counteracting are conducted between two roles, the artwork owner and the style attacker. The artwork owner has to release its own pieces of artwork, but does not want others to imitate the novel style by ANST models. In contrast, the style attacker is assumed to imitate and transfer the style onto a content image by ANST models as shown in Figure 2, which is, however, unauthorized by the artwork owner. We formulate this process as

$$\boldsymbol{c_s} = g\left(\Phi\left(f, \boldsymbol{s}\right), \boldsymbol{c}\right), \tag{3}$$

where $\boldsymbol{s}$ is the source image, $\boldsymbol{c}$ is the content image, f is the encoder, Φ is the function to extract style representation from f, g is the decoder which transfers the style representation to the content image, and $\boldsymbol{c_s}$ is the transferred image by ANST with the style of $\boldsymbol{s}$ and the content of $\boldsymbol{c}$. Most of the ANST models follow this encoder-decoder architecture [4, 7, 15, 23, 25, 31, 38, 49]. The encoder f and Φ extract the style representation, while the decoder transfers this style representation onto the content images. In this work, for convenience, we consider all the other components in the ANST, except the encoder, as a part of the decoder. In this work, we focus on this architecture since it covers a variety of ANST models.

Protection against ANST and the challenge. In order to protect the style from ANST models, before releasing the images to the public, the artwork owner slightly changes the images in an imperceptible way to get $\boldsymbol{s} + \boldsymbol{\delta}$ ($\|\boldsymbol{\delta}\|_\infty \leq \epsilon$) for releasing. It aims to decrease the performance but does not make a difference to the images in human eyes. In other words, the goal is to make $\boldsymbol{c_s}$ have a different style from $\boldsymbol{s}$ by adding $\boldsymbol{\delta}$ on $\boldsymbol{s}$. Obviously, the artwork owner cannot change the data after releasing it and has no control which ANST model the style attacker will use. Thus, this leads to the challenge of generalization onto unknown ANST models as we mentioned in Section 1. If we only design the protection based on one ANST model, the images are still exposed under the risk of being imitated by other methods. Therefore, Neural Style Protection (NSP) is proposed to enhance the ability in generalization, which is necessary to provide comprehensive protection. In the following subsections, we show that NSP solves the problem by removing the most diverse decoders and assembling the inconsistent encoders via two strategies.

3.2. Altering the Intermediate Style Representation

To enhance generalization, we try to find a way to prevent NSP from being specific to a single ANST model and its architecture. In this subsection, we propose to alter the intermediate style representation to avoid overfitting on one specific decoder within an ANST model. As introduced in Section 3.1, most of the ANST models follow the encoder-decoder architecture. For different ANST models, The encoders are often similar and many even share the same network and parameters, like the pre-trained VGG-19 [35]. In contrast, the decoders receive different input style representations, have different auxiliary networks and are usually extensively trained for transferring the style in different manners, which are more diverse than the encoders. Thus, different from changing the final transferred image $\boldsymbol{c_s}$ (which involves the whole ANST process including both the encoder and decoder in an end-to-end manner), altering the intermediate style representation $\Phi\left(f, \boldsymbol{s}\right)$ can remove the interaction with the decoder part, and thus prevent overfitting on one specific decoder. In this subsection, we introduce two strategies of altering the style representation to avoid the decoder part for two cases according to the source style and source content usage accordingly, and then in Section 3.3 we show how to further improve the protection by assembling different methods of extracting the style representations.

For the first case where most of the ANST models whose style representations are extracted independently on the content images like AdaIN [15] and CAST [49], NSP only involves the part of style representation extraction. In other words, instead of decreasing the performance of the whole process $g\left(\Phi\left(f, \boldsymbol{s}\right), \boldsymbol{c}\right)$, NSP alters the style representation $\Phi\left(f, \boldsymbol{s}\right)$ to be as different from the protected style as possible. We denote the distance between style representations as $\mathcal{D}$, and define the protection objective to be

$$\arg\max_{\boldsymbol{\delta}} \mathcal{D}(\Phi\left(f, \boldsymbol{s}+\boldsymbol{\delta}\right), \Phi\left(f, \boldsymbol{s}\right)), \text{ s.t. } \|\boldsymbol{\delta}\|_\infty \leq \epsilon. \quad (4)$$

We use Mean Square Error (MSE) as $\mathcal{D}$ in this work. By maximizing MSE between the style representation of the clean source image and the source image protected by $\boldsymbol{\delta}$, NSP can make the extracted style representation by ANST to be very different from the protected images and thus provide protection for the style of source image. Although in different ANST models, Φ and f have different outputs, MSE is easy to adopt for different models. For example, the statistics of feature maps, i.e. the mean and variance of some layers of f, is widely used as the style representation by many ANST models like AdaIN. Replacing $\mathcal{D}$ in Eq. 4 with the statistics, the objective for AdaIN is

$$\begin{aligned} \arg\max_{\boldsymbol{\delta}} & \sum_{i=1}^{m} \|\mu(f_i(\boldsymbol{s})) - \mu(f_i(\boldsymbol{s}+\boldsymbol{\delta}))\|_2 \\ & + \sum_{i=1}^{m} \|\sigma(f_i(\boldsymbol{s})) - \sigma(f_i(\boldsymbol{s}+\boldsymbol{\delta}))\|_2, \\ & \text{s.t. } \|\boldsymbol{\delta}\|_\infty \leq \epsilon, \end{aligned} \quad (5)$$

where f_i is one specified layer of encoder f, m is the number of the layers used in style representation, $\mu(\cdot)$ is the mean function, and $\sigma(\cdot)$ is the variance function.

For the second case, not all the ANST models extract the style representation independently on the content images. A small group of ANST models including AdaAttN [25] extract the style representation based on the pixel-level information of content images, which means Φ takes both the source image $\boldsymbol{s}$ and the content image $\boldsymbol{c}$ as input. We denote the Φ in this case as Φ'. For these models, NSP randomly chooses n content images and calculates the average loss of the distance between style representations to enhance the generalization as follows,

$$\arg\max_{\boldsymbol{\delta}} \sum_{i=1}^{n} \mathcal{D}(\Phi'\left(f, \boldsymbol{s}+\boldsymbol{\delta}, \boldsymbol{c}_i\right), \Phi'\left(f, \boldsymbol{s}, \boldsymbol{c}_i\right)), \quad (6)$$

$$\text{s.t. } \|\boldsymbol{\delta}\|_\infty \leq \epsilon. \quad (7)$$

Altering the style representation does not involve the decoder part, which reduces the potential overfitting to the decoder. Meanwhile, randomly selected content images can prevent δ from overfitting on a specific content image.

Based on these two measures of alterning the intermediate style representation, NSP can avoid changing the final $\boldsymbol{c_s}$ that is directly correlatesd with the decoder part. In such way, our method can mitigate the overfitting on the diverse decoders and advance the generalization across different models. To further reduce the influence of the differences among encoders which are less diverse than decoders, we introduce an ensemble method in the next subsection.

3.3. Model Ensemble with Momentum-iterative

As mentioned above, although we mitigate the influence of the diverse decoders, ANST's encoders still generate slightly different features. It means that it is still possible for us to improve the generalization via considering the difference of encoders. Thus, to further boost the generalization, we propose to assemble style representations from different encoders in NSP.

We choose three ANST methods which use different representative encoders to increase NSP's ability of generalization in different models. These models are AdaIN that uses original pre-trained VGG-19 as encoder, CAST that uses fine-tuned VGG-19 as encoder and AdaAttN that involves content images when extracting the style representation. We assemble these models by summing up the gradient of PGD as follows,

$$\begin{aligned} \arg\max_{\boldsymbol{\delta}} & \sum_{i=1}^{M} \lambda_i \mathcal{D}\left(\Phi_i\left(f_i, \boldsymbol{s}+\boldsymbol{\delta}\right), \Phi_i\left(f_i, \boldsymbol{s}\right)\right), \\ & \text{s.t. } \|\boldsymbol{\delta}\|_\infty \leq \epsilon, \end{aligned} \quad (8)$$

where f_i represents the encoders from the ANST models to assemble, Φ_i is the corresponding function to extract the style representation and λ_i is the weight to balance between different models. For convenient, we simplify the extraction function of AdaAttN $\Phi_i'\left(f_i, \boldsymbol{s}+\boldsymbol{\delta}, \boldsymbol{c}_j\right)$ into $\Phi_i\left(f_i, \boldsymbol{s}+\boldsymbol{\delta}\right)$. To solve this objective, we use PGD as follows,

$$\boldsymbol{\delta}_{t+1} = \text{clip}_{(-\epsilon,\epsilon)}\left\{\boldsymbol{\delta}_t + \alpha \cdot \text{sign}\left(\nabla_{\boldsymbol{\delta}_t} \mathcal{L}_{\text{ensemble}}\right)\right\}, \quad (9)$$

where

$$\begin{aligned} \nabla_{\boldsymbol{\delta}_t} \mathcal{L}_{\text{ensemble}} &= \nabla_{\boldsymbol{\delta}_t} \sum_{i=1}^{M} \lambda_i \mathcal{D}\left(\Phi_i\left(f_i, \boldsymbol{s}+\boldsymbol{\delta}_t\right), \Phi_i\left(f_i, \boldsymbol{s}\right)\right) \\ &= \sum_{i=1}^{M} \lambda_i \nabla_{\boldsymbol{\delta}_t} \mathcal{D}\left(\Phi_i\left(f_i, \boldsymbol{s}+\boldsymbol{\delta}_t\right), \Phi_i\left(f_i, \boldsymbol{s}\right)\right). \end{aligned} \quad (10)$$

By directly using Eq. 9, we will empirically find that because of the sign$(\cdot)$ operation, in each step, the sign of every element is dominant by only one ANST model and the influence of other ANST models will be overlooked. During the PGD updating process, the sign of different steps are unstable due

to the change of the gradient from the dominant model in each step. Thus, we adopt the momentum-based PGD in [8] as:

$$\begin{aligned} \boldsymbol{\delta}_{t+1} &= \text{clip}_{(-\epsilon,\epsilon)}\left\{\boldsymbol{\delta}_t + \alpha \cdot \text{sign}\left(\boldsymbol{g}_{t+1}\right)\right\}, \\ \boldsymbol{g}_{t+1} &= \eta * \boldsymbol{g}_t + \nabla_{\boldsymbol{\delta}_t} \mathcal{L}_{\text{ensemble}}. \end{aligned} \tag{11}$$

We remove the softmax function in [8] and directly sum up the gradients from different models following Eq. 10. The accumulated gradient $\boldsymbol{g}_t$ can keep the influence of all the assembled ANST models and make the sign of PGD updating more stable. With the momentum-based ensemble, NSP can have better generalization on different ANST models.

4. Experiment

In this subsection, we conduct experiments to illustrate the effectiveness of protection. We describe the experiment setting in Section 4.1 and results in Section 4.2 and 4.3.

4.1. Experimental Settings

Dataset. We randomly select a subset of places365 dataset [50] to be the content images. Style images are collected from WikiArt [6]. In total, we create 10,000 style transfer images based on 1000 different content images and 300 style images. To fairly evaluate our proposed algorithm, we generate transferred images based on clean source style images and corresponding protected style images along with the same content. The comparison of these two transferred images demonstrates the effectiveness of protection.

ASNT models used by style attacker. Since NSP only assembles three ANST models, we also test the performance on unknown models to evaluate the protection of unknown models. We use AdaIN [15], AdaAttN [25] and CAST [49] as the known models for generating NSP perturbations and test the protection performance, and use SANet [31] and ArtFlow [1] as unknown models for only testing the protection. All the models are trained with MS-COCO [24] as the content images, except CAST which is trained with Place365 [50] as content images. AdaIN [15] uses the first few layers of a VGG-19 pretrained on ImageNet as the encoder. The feature outputs right after the ReLU4_1 layer go through adaptive instance normalization and then feed into the decoder. The decoder architecture almost mirrors the encoder, with all pooling layers replaced by the nearest up-sampling. AdaAttN [7] uses the same pretrained VGG-19 encoder. It integrates the outputs of different layers of the encoder after being normalized by an AdaAttN model. Each layer's output is concatenated with the former layers before going through the AdaAttN module to further utilize the features of shallow layers. CAST [49] adopts the same architecture skeleton with AdaIN. The pretrained VGG-19 is further fine-tuned via contrastive learning. For the unseen model we consider, SANet is also an element-wise attention-based style transfer model. It adopts the last two layers as the style representation. ArtFlow is built with a projection flow network instead of the encoder-decoder pipeline. It encodes the style and content images through the forward flow and transforms the stylized feature to the stylized image via reverse inference through the same network module with the forward pass.

Implementation. We use 50-step PGD for optimization on the NSP perturbations. We set the perturbation budget of the perturbed style images to 8/255 for l_∞ norm. In each update step, the step size is set to 0.8/255. For the models that require multiple contents, we set n = 5. For each optimization process, we select 5 content images different from the 1000 content images that will be used in the evaluation.

Baseline methods. We use two kinds of baseline methods. The first one is to perturb the clean style images with random noise, including Gaussian noise and Uniform noise. For Gaussian noise, we set the mean to 0 and variance to 1, and to fit into the l-norm bound, we clip the noise by [-8/255, 8/255]. We sample uniform noise of the input sample shape and each element is uniformly sampled from [-8/255, 8/255]. The second baseline is the end-to-end adversarial attack, which tries to increase the l_2 distance of the final image as we compared in Section 3.2.

Evaluation Metrics. We quantify the average perceptual similarity between protected transferred images and corresponding unprotected transferred images of the 10000 image pairs with two metrics, LPIPS [48] and SSIM [40]. Specifically, For LPIPS, we use VGG as the feature extractor.

Table 1. Protection on known models in the ensemble

	Protection method	AdaIN	CAST	AdaAttN	AVG	WORST
LPIPS	Uniform	0.064	0.011	0.036	0.037	0.011
	Gaussian	0.096	0.286	0.106	0.163	0.096
	End (AdaIN)	0.301	0.130	0.142	0.191	0.130
	End (CAST)	0.068	**0.722**↑	0.079	0.290	0.079
	End (AdaAttN)	0.146	0.150	**0.455**↑	0.250	0.146
	NSP (ours)	**0.360**↑	0.308	0.327	**0.332**↑	**0.308**↑
SSIM	Uniform	0.901	0.972	0.917	0.930	0.973
	Gaussian	0.867	0.658	0.806	0.777	0.867
	End (AdaIN)	0.512	0.816	0.745	0.691	0.816
	End (CAST)	0.898	**0.149**↓	0.845	0.631	0.845
	End (AdaAttN)	0.788	0.796	**0.387**↓	0.657	0.796
	NSP (ours)	**0.504**↓	0.639	0.561	**0.568**↓	**0.639**↓

Table 2. Generalization on unknown models. (AVG and WORST results contain the results of known models.)

	Protection method	SANet	ArtFlow	AVG	WORST
LPIPS	End (AdaIN)	0.224	0.093	0.178	0.093
	End (CAST)	0.106	0.103	0.216	0.103
	End (AdaAttN)	0.253	0.065	0.214	0.065
	NSP (ours)	**0.307**↑	**0.135**↑	**0.282**↑	**0.135**↑
SSIM	End (AdaIN)	0.439	0.835	0.669	0.867
	End (CAST)	0.696	0.790	0.676	0.898
	End (AdaAttN)	0.412	0.862	0.649	0.862
	NSP (ours)	**0.349**↓	**0.778**↓	**0.566**↓	**0.778**↓

Figure 3. Examples of Style Protection against AdaIN, CAST, and AdaAttN by NSP and baseline methods. We observe that NSP can change the transferred style by all the ANST models apparently, while end-to-end baseline methods have only influence on one ANST model, while random noise baseline has almost no protection. More examples can be found in the supplementary materials.

4.2. Protection against ANST Models

In this subsection, we demonstrate that our method can not only protect the styles against known ANST models that are used in the ensemble but also protect against unknown ANST models to some extent. In Table 1 and Table 2, we report both the protection on known and unknown models, respectively. We first use ANST models to transfer the style from the unprotected source images onto content images and use the same model to transfer the style from the protected source images onto the same content images. Then LPIPS and SSIM are calculated to compare the difference between the style of transferred images from unprotected source images and protected images.

Protection on known models. As shown in Table 1, NSP provides remarkably better protection than random noise and better generalization than end-to-end baselines. We denote the end-to-end method as "End (model name)" in the table. In detail, **1)** after protection by Gaussian noise and uniform noise, LPIPS is nearly 0, which means the f difference of visual appearance between transferred images from unprotected and protected source images is quite small. The worst case of uniform noise is 0.011, which can be interpreted as random noise providing almost no protection. The only exception is that CAST is sensitive to Gaussian noise. In contrast, our method can increase LPIPS to 0.332 on average, which makes the protected style clearly different. Similarly, the SSIM of random noise is close to 1, which means the similarity between transferred images is high and the unprotected images do not change the transferred style. **2)** Although the end-to-end method can protect against the model that is used to generate the adversarial perturbation, it is hard to generalize onto other methods. Instead, our method can perform much better when generalizing on all the models. For example, end (CAST) can protect CAST well, but it provides almost no protection on AdaIN, which gets 0.068 in LPIPS and 0.898 in SSIM. Thus, protection by AVG and WORST of end-to-end is also reduced because of the bad generalization performance, while our proposed NSP can have a much better LPIPS which is around 0.1 higher than others, and SSIM which is around 0.1 better than others, which indicates the best protection against all models. This suggests that our NSP can provide better protection on ANST models that are based on the pre-trained VGG19 encoder (because this is used by two of the three methods in the ensemble model). This can be an advantage since most ANST models use VGG19 as the encoder for extracting the style representation. Figure 3 presents visualizations of the style transferred images and protected transferred images of baseline algorithms and our NSP. Starting from row 2, each column represents a style-transferred image generated by a model. As we can see, the last row shows the NSP protected style transfer images, which are clearly different from the images from the second row. A local detail illustration can be found in Figure 4

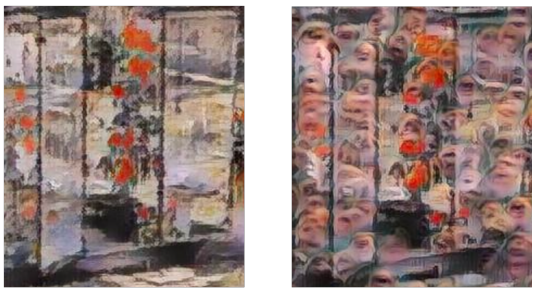

Figure 4. Example Comparison. Local details of clean transferred images(Figure 3 Row 2, Left 1) and protected transferred images(Row 7, Left 1) generated by AdaAttN.

Protection on unknown models. Table 2 shows the generalization ability on unknown ANST models. Compared with end-to-end baselines, our performance is much better on both LPIPS and SSIM, no matter in single models, AVG and WORST scores. The AVG LPIPS is around 0.1 higher than baseline methods and AVG SSIM is around 0.1 lower than baseline methods. This implies that our model can also generalize well on unknown models. Comparing the protection against SANet and ArtFlow, NSP can prevent SANet better, because SANet uses the pre-trained VGG19 as the extraction encoder, while ArtFlow uses Projection Flow Network to replace the VGG19 network.

4.3. Ablation Studies

In this subsection, we discuss how the two proposed strategies in NSP influence the protection effect.

1) Altering style representations. As discussed in Section 3.2, altering the style representations can increase the generalization ability across different ANST models because this can remove the interaction with the diverse decoder and thus prevent overfitting on decoders. We compare altering the style representation (without ensemble) and end-to-end adversarial perturbations in Table 3. The results show that altering intermediate style representation demonstrates greater transferability than end-to-end perturbation. Even though the models are not assembled, altering intermediate style representation also has better WORST performance in all cases and better AVG for some cases among all the ANST models. For instance, LPIPS of s.r. (AdaAttN) with end-to-end adversarial perturbation increases the LPIPS from 0.146 to 0.195 for AdaIN and 0.150 to 0.179 for CAST. Similar increases can be observed in the altering style representation protection on the other two models.

2) Model Ensemble based on Momentum-iterative. In this subsection, we demonstrate that the ensemble can reduce the influence of the gap between different encoders and increase the ability of the generalization, while the momentum-iterative method can reduce the unstable of gradient direc-

Table 3. End-to-end vs. altering style representations

Metric	Model		AdaIN	CAST	AdaAttN	AVG	WORST
LPIPS	AdaIN	End	0.301	0.130	0.142	0.191	0.130
		s.r.	0.443↑	0.182↑	0.248↑	0.291↑	0.182↑
	CAST	End	0.068	0.722	0.079	0.290	0.079
		s.r.	0.105↑	0.543↑	0.098↑	0.249↓	0.098↑
	AdaAttN	End	0.146	0.150	0.455	0.250	0.150
		s.r.	0.195↑	0.179↑	0.304↓	0.226↓	0.179↑
SSIM	AdaIN	End	0.512	0.816	0.745	0.691	0.816
		s.r.	0.403↓	0.759↓	0.657↓	0.606↓	0.759↓
	CAST	End	0.898	0.149	0.845	0.631	0.845
		s.r.	0.857↓	0.354↑	0.827↓	0.679↑	0.827↓
	AdaAttN	End	0.788	0.796	0.387	0.657	0.796
		s.r.	0.735↓	0.763↓	0.589↑	0.696↑	0.763↓

Table 4. Single ANST model vs. ensemble

Metric	Defense	AdaIN	CAST	AdaAttN	AVG	WORST
LPIPS	s.r. (AdaIN)	**0.443**↑	0.182	0.248	0.291	0.182
	s.r. (CAST)	0.105	**0.543**↑	0.098	0.249	0.098
	s.r. (AdaAttN)	0.195	0.179	0.304	0.226	0.179
	Ensemble	0.341	0.322	0.319	0.327	**0.319**↑
	NSP (ours)	0.360	0.309	**0.327**↑	**0.332**↑	0.309
SSIM	s.r. (AdaIN)	**0.403**↓	0.759	0.657	0.606	0.759
	s.r. (CAST)	0.857	**0.354**↓	0.827	0.679	0.827
	s.r. (AdaAttN)	0.735	0.763	0.589	0.696	0.763
	Ensemble	0.528	0.626	0.576	0.577	**0.626**↓
	NSP (ours)	0.504	0.639	**0.561**↓	**0.568**↓	0.639

tions and further improve the sweet-point in the trade-off between all the models. In Table 4, we compare the protection performance of single models without assembling, ensemble without momentum and momentum-iterative version, NSP. As we can see, a benign ensemble without momentum can increase the generalization significantly, especially in AVG score. NSP with momentum can further improve the AVG by 0.005 in LPIPS and 0.009 in SSIM. To understand the improvement made by momentum, we count the number of the elements which have a changed sign compared with last step in the protection perturbation for each updating step of the PGD process, as shown in Figure 5. This value reflects the instability of the updating process. We can see that in each updating step, our NSP can reduce the change rate by at least 3%. Helped from this, all the models in the ensemble can have an effect in each updating step, instead of only the dominant model, which further increases the generalization performance.

4.4. Robustness

Robustness against noise and image processing is important since the protected images are under risk of distortion, compression, and deliberate preprocessing during storage and distribution. Thus, we tested the protection performance of our method under different image corruptions including Gaussian Noise (GN), Gaussian Filter (GF), JPEG Compression, Crop, and Rotation. As shown in Table 5, although most of the image corruption can slightly reduce the protection, the performance under the corruptions is still better than the baseline methods in Table 1 in AVG and WORST

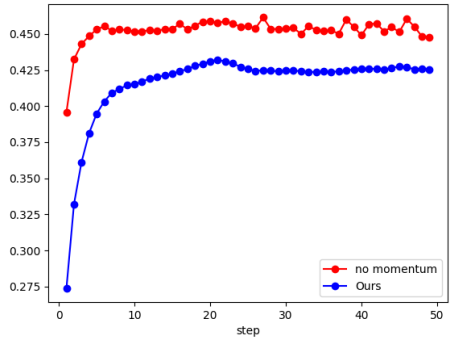

Figure 5. Sign change rate in the optimization process of NSP vs. ensemble without momentum.

Table 5. Style Image Preprocessing as Defenses

Metric	Defense	AdaIN	CAST	AdaAttN	AVG	WORST
LPIPS	Original	0.360	0.309	0.327	0.332	0.309
	GN	0.339	0.349	0.297	0.328	0.297
	GF(3×3)	0.353	0.247	0.271	0.290	0.247
	GF(5×5)	0.355	0.245	0.274	0.291	0.245
	JPEG	0.353	0.237	0.269	0.286	0.237
	Crop	0.378	0.268	0.363	0.337	0.268
	Rotation	0.354	0.243	0.270	0.289	0.243
SSIM	Original	0.504	0.639	0.561	0.568	0.639
	GN	0.549	0.594	0.591	0.578	0.594
	GF(3×3)	0.517	0.690	0.602	0.603	0.690
	GF(5×5)	0.518	0.693	0.603	0.605	0.693
	JPEG	0.517	0.703	0.602	0.607	0.703
	Crop	0.580	0.641	0.616	0.612	0.641
	Rotation	0.518	0.694	0.604	0.605	0.694

score. The Gaussian noise has even improved the LPIPS for CAST and also the average AVG LPIPPS. This result is consistent with the results in Table 1 that Gaussian noise also can prevent the style transferring in some extent. For other image processing, NSP still achieves a protection effect only with a drop on LPIPS around 0.02.

5. Conclusions

In this paper, we propose NSP to protect the style of images from being imitated by Arbitrary Style Neural Transfer, which alleviates the concerns of artists. To address the challenge that artworks might be imitated by unknown ANST, NSP uses style representation as the objective to prevent adversarial perturbations from overfitting the diverse decoder part in ANST. Additionally, a momentum-based model ensemble is employed to further align differences in encoders of different ANST models. These two strategies enhance ANST's ability to generalize across various ASNT models. Our experiments have shown that NSP can provide effective protection for both seen and unseen ANST models. In future work, we will focus on enhancing protection performance and extending the protection to other Neural Style Transfer methods.

References

[1] Jie An, Siyu Huang, Yibing Song, Dejing Dou, Wei Liu, and Jiebo Luo. Artflow: Unbiased image style transfer via reversible neural flows. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 862–871, 2021. 5

[2] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. Ieee, 2017. 1, 2

[3] Dongdong Chen, Lu Yuan, Jing Liao, Nenghai Yu, and Gang Hua. Stylebank: An explicit representation for neural image style transfer, 2017. 2

[4] Haibo Chen, Zhizhong Wang, Huiming Zhang, Zhiwen Zuo, Ailin Li, Wei Xing, Dongming Lu, et al. Artistic style transfer with internal-external learning and contrastive learning. *Advances in Neural Information Processing Systems*, 34:26561–26573, 2021. 2, 3

[5] Tian Qi Chen and Mark Schmidt. Fast patch-based style transfer of arbitrary style. *arXiv preprint arXiv:1612.04337*, 2016. 1, 2

[6] Michael Danielczuk, Matthew Matl, Saurabh Gupta, Andrew Li, Andrew Lee, Jeffrey Mahler, and Ken Goldberg. Segmenting unknown 3d objects from real depth images using mask r-cnn trained on synthetic data. In *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2019. 5

[7] Yingying Deng, Fan Tang, Weiming Dong, Haibin Huang, Chongyang Ma, and Changsheng Xu. Arbitrary video style transfer via multi-channel correlation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1210–1217, 2021. 2, 3, 5

[8] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018. 2, 5

[9] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style, 2017. 2

[10] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015. 1

[11] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016. 1

[12] Golnaz Ghiasi, Honglak Lee, Manjunath Kudlur, Vincent Dumoulin, and Jonathon Shlens. Exploring the structure of a real-time, arbitrary neural artistic stylization network. *arXiv preprint arXiv:1705.06830*, 2017. 1, 2

[13] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 1, 2

[14] Hanxun Huang, Xingjun Ma, Sarah Monazam Erfani, James Bailey, and Yisen Wang. Unlearnable examples: Making personal data unexploitable. *arXiv preprint arXiv:2101.04898*, 2021. 3

[15] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. (arXiv:1703.06868), Jul 2017. 1, 2, 3, 4, 5

[16] Linxi Jiang, Xingjun Ma, Shaoxiang Chen, James Bailey, and Yu-Gang Jiang. Black-box adversarial attacks on video recognition models. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 864–872, 2019. 2

[17] Yongcheng Jing, Xiao Liu, Yukang Ding, Xinchao Wang, Errui Ding, Mingli Song, and Shilei Wen. Dynamic instance normalization for arbitrary style transfer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4369–4376, 2020. 1, 2

[18] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. Neural style transfer: A review. Oct 2018. arXiv:1705.04058 [cs, eess, stat]. 2

[19] 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*, 26(11):3365–3385, 2019. 1

[20] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 694–711. Springer, 2016. 1, 2

[21] Chuan Li and Michael Wand. Combining markov random fields and convolutional neural networks for image synthesis, 2016. 2

[22] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Diversified texture synthesis with feed-forward networks, 2017. 2

[23] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. *Advances in neural information processing systems*, 30, 2017. 2, 3

[24] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Doll'a r, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 5

[25] Songhua Liu, Tianwei Lin, Dongliang He, Fu Li, Meiling Wang, Xin Li, Zhengxing Sun, Qian Li, and Errui Ding. Adaattn: Revisit attention mechanism in arbitrary neural style transfer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6649–6658, 2021. 1, 2, 3, 4, 5

[26] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks, 2017. 2

[27] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 1, 2

[28] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them, 2014. 2

[29] Aravindh Mahendran and Andrea Vedaldi. Visualizing deep convolutional neural networks using natural pre-images. *International Journal of Computer Vision*, 120(3):233–255, may 2016. 2

[30] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. Inceptionism: Going deeper into neural networks, 2015. 2

[31] Dae Young Park and Kwang Hee Lee. Arbitrary style transfer with style-attentional networks, 2019. 2, 3, 5

[32] Ali Shafahi, W Ronny Huang, Mahyar Najibi, Octavian Suciu, Christoph Studer, Tudor Dumitras, and Tom Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. *Advances in neural information processing systems*, 31, 2018. 3

[33] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32, 2019. 3

[34] Lu Sheng, Ziyi Lin, Jing Shao, and Xiaogang Wang. Avatar-net: Multi-scale zero-shot style transfer by feature decoration, 2018. 2

[35] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3

[36] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 1

[37] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images, 2016. 2

[38] Pei Wang, Yijun Li, and Nuno Vasconcelos. Rethinking and improving the robustness of image style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 124–133, 2021. 2, 3

[39] Xiaosen Wang and Kun He. Enhancing the transferability of adversarial attacks through variance tuning. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 1924–1933, Nashville, TN, USA, Jun 2021. IEEE. 2

[40] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 5

[41] Cheng-Hsin Weng, Yan-Ting Lee, and Shan-Hung Brandon Wu. On the trade-off between adversarial and backdoor robustness. *Advances in Neural Information Processing Systems*, 33:11973–11983, 2020. 1

[42] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020. 3

[43] Dongxian Wu, Yisen Wang, Shu-Tao Xia, James Bailey, and Xingjun Ma. Skip connections matter: On the transferability of adversarial examples generated with resnets. *arXiv preprint arXiv:2002.05990*, 2020. 2

[44] Yifeng Xiong, Jiadong Lin, Min Zhang, John E. Hopcroft, and Kun He. Stochastic variance reduced ensemble adversarial attack for boosting the adversarial transferability. (arXiv:2111.10752), Apr 2022. arXiv:2111.10752 [cs]. 2

[45] Han Xu, Yao Ma, Haochen Liu, Debayan Deb, Hui Liu, Jiliang Tang, and Anil K. Jain. Adversarial attacks and defenses in images, graphs and text: A review, 2019. 1

[46] Han Xu, Yao Ma, Hao-Chen Liu, Debayan Deb, Hui Liu, Ji-Liang Tang, and Anil K Jain. Adversarial attacks and defenses in images, graphs and text: A review. *International Journal of Automation and Computing*, 17:151–178, 2020. 2

[47] Hang Zhang and Kristin Dana. Multi-style generative network for real-time transfer, 2017. 2

[48] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric, 2018. 5

[49] Yuxin Zhang, Fan Tang, Weiming Dong, Haibin Huang, Chongyang Ma, Tong-Yee Lee, and Changsheng Xu. Domain enhanced arbitrary image style transfer via contrastive learning. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–8, 2022. 1, 2, 3, 4, 5

[50] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 5