ArtFlow: Unbiased Image Style Transfer via Reversible Neural Flows

Jie An[1*] Siyu Huang[2*] Yibing Song[3] Dejing Dou[2] Wei Liu[4] Jiebo Luo[1]
[1]University of Rochester [2]Baidu Research [3]Tencent AI Lab [4]Tencent Data Platform
{jan6,jluo}@cs.rochester.edu {huangsiyu,doudejing}@baidu.com
yibingsong.cv@gmail.com wl2223@columbia.edu

Abstract

Universal style transfer retains styles from reference images in content images. While existing methods have achieved state-of-the-art style transfer performance, they are not aware of the content leak phenomenon that the image content may corrupt after several rounds of stylization process. In this paper, we propose ArtFlow to prevent content leak during universal style transfer. ArtFlow consists of reversible neural flows and an unbiased feature transfer module. It supports both forward and backward inferences and operates in a projection-transfer-reversion scheme. The forward inference projects input images into deep features, while the backward inference remaps deep features back to input images in a lossless and unbiased way. Extensive experiments demonstrate that ArtFlow achieves comparable performance to state-of-the-art style transfer methods while avoiding content leak.

1. Introduction

Neural style transfer aims at transferring the artistic style from a reference image to a content image. Starting from [11, 13], numerous works based on iterative optimization [12, 44, 30, 34] and feed-forward networks [23, 53, 3, 63] improve style transfer from either visual quality or computational efficiency. Despite tremendous efforts, these methods do not generalize well for multiple types of style transfer. Universal style transfer (UST) is proposed to improve this generalization ability. The representative UST methods include AdaIN [20], WCT [32], and Avatar-Net [45]. These methods are continuously extended by [15, 22, 60, 1, 45, 33, 40, 31, 2, 56]. While achieving favorable results as well as generalizations, these methods are limited to disentangling and reconstructing image content during the stylization process. Fig. 1 shows some examples. Existing methods [32, 20, 45] effectively stylize content images in (c). However, image contents are corrupted after several rounds of stylization process where we send the reference image and the output result into these methods. We define this phenomenon as content leak and provide an analysis in the following:

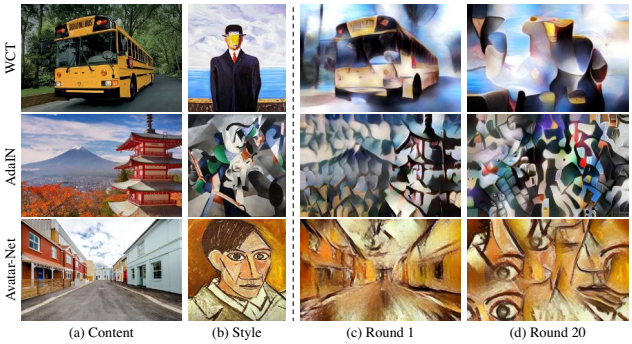
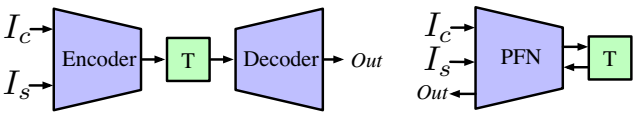


Figure 1. Content leak visualization. Existing style transfer methods are not effective to preserve image content after several rounds of stylization process as shown in (d), although their performance is state-of-the-art in the first round as shown in (c).

Content leak appears due to the design of UST methods that usually consist of three parts: the first part is a fixed encoder for image embedding, the second part is a learnable decoder to remap deep features back to images, and the third part is a style transfer module based on deep features. We observe that the first part is fixed. The appearance of content leak indicates the accumulated image reconstruction errors brought by the decoder, or the biased training process of either the decoder or the style transfer module. Specifically, the content leaks of WCT [32] and its variants [31, 40, 56] is mainly caused by the image reconstruction error of the decoder. The content leak of AdaIN series [20, 22, 60] and Avatar-Net [45] are additionally caused by the biased decoder training and a biased style transfer module, respectively. Sec. 3 shows more analyses.

In this work, we propose an unbiased style transfer framework called ArtFlow to robustify exisiting UST methods upon overcoming content leak. Different from the prevalent encoder-transfer-decoder structure, ArtFlow in-

*J. An and S. Huang contribute equally. This work is done when J. An is an intern in Tencent AI Lab. The code is available at https://github.com/pkuanjie/ArtFlow.



(a) Universal Style Transfer Framework (b) ArtFlow Framework

Figure 2. A comparison between the auto-encoder based style transfer framework and the proposed ArtFlow framework.

troduces both forward and backward inferences to formulate a projection-transfer-reversion pipeline. This pipeline is based on neural flows [5] and only contains a Projection Flow Network (PFN) in conjunction with an unbiased feature transfer module. The neural flow refers to a number of deep generative models [5, 18] which estimate density through a series of reversible transformations. Our PFN follows the neural flow model GLOW [28] which consists of a chain of revertible operators including activation normalization layers, invertible 1×1 convolutions, and affine coupling layers [6]. Fig. 2 shows the structure of ArtFlow. It first projects both the content and style images into latent representations via forward inference. Then, it makes unbiased style transfer upon deep features and reconstructs the stylized images via reversed feature inference.

The proposed PFN avoids the image reconstruction error and image recovery bias which usually appear in the encoder-decoder framework. PFN allows unbiased and lossless feature extraction and image recovery. To this end, PFN facilitates the comparison of style transfer modules in a fair manner. Based on PFN, we perform theoretical and empirical analyses of the inherent biases of style transfer modules adopted by WCT, AdaIN, and Avatar-Net. We show that the transfer modules of AdaIN and WCT are unbiased, while the transfer module of Avatar-Net is biased towards style. Consequently, we adopt the transfer modules of AdaIN and WCT as the transfer modules for ArtFlow to achieve an unbiased style transfer.

The contributions of this work are three-fold:

- We reveal the Content Leak issue of the state-of-the-art style transfer algorithms and identify the three main causes of the Content Leak in AdaIN [20], WCT [32], and Avatar-Net [45].
- We propose an unbiased, lossless, and reversible network named PFN based on neural flows, which allows both theoretical and empirical analyses of the inherent biases of the popular style transfer modules.
- Based on PFN in conjunction with an unbiased style transfer module, we propose a novel style transfer framework, *i.e.*, ArtFlow, which achieves comparable style transfer results to state-of-the-art methods while avoiding the Content Leak issue.

2. Related Work

Image Style Transfer. Image style transfer is a long-standing research topic. Before deep neural networks [49, 37, 59, 57] are applied to the style transfer, several algorithms based on stroke rendering [16], image analogy [17, 46, 10, 48, 35, 51, 50], and image filtering [61] are proposed to make artistic style transfer. These methods usually have to trade-off between style transfer quality, generalization, and efficiency. Gatys *et al*. [11, 13] introduce a Gram loss upon deep features to represent image styles, which opens up the neural style transfer era. Inspired by Gatys *et al*., numerous neural style transfer methods have been proposed. We categorize these methods into one style per model [29, 54, 23, 53, 55, 58, 44, 30], multi-style per model [7, 3, 19, 14], and universal style transfer methods [4, 32, 20, 45, 15, 2, 56, 38, 60, 1] with respect to their generalization abilities. In this paper, our ArtFlow belongs to universal style transfer and it consists of reversible neural flows. The forward and backward inferences are utilized for lossless and unbiased image recovery.

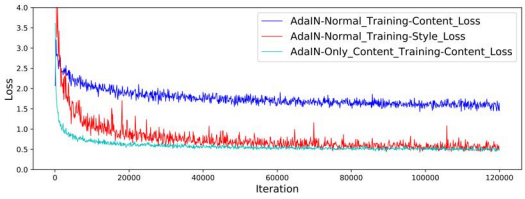


Figure 3. Loss curves of AdaIN [20] training: Using both content and style losses *vs*. only using the content loss.

Neural flows. Neural flows refer to a subclass of deep generative models, which learns the exact likelihood of high dimensional observations (*e.g.*, natural images, texts, and audios) through a chain of reversible transformations. As a pioneering work of neural flows, NICE [5] is proposed to transform low dimensional densities to high dimensional observations with a stack of affine coupling layers. Following NICE, a series of neural flows, including RealNVP [6], GLOW [28], and Flow++ [18], are proposed to improve NICE with more powerful and flexible reversible transformations. The recently proposed neural flows [28, 18, 39] are capable of synthesizing high-resolution natural/face images, realistic speech data [43, 26], and performing make-up transfer [8]. In this work, the proposed ArtFlow consists of a reversible network PFN and an unbiased feature transfer module. The content leak can be addressed via lossless forward and backward inferences and unbiased feature transfer. In comparison, BeautyGlow [8] shares the similar spirits but is not applicable for unbiased style transfer.

3. Pre-analysis

Before introducing the proposed ArtFlow, we first make a pre-analysis to uncover the Content Leak phenomenon of the state-of-the-art style transfer algorithms and analyze the

(a) Input (b) Round 1 (c) Round 20 (d) Round 40

Figure 4. Multiple rounds of image encoding and decoding using the auto-encoder of AdaIN [20].

causes of Content Leak. We make the aforementioned pre-analysis by answering two questions: What Content Leak is and why Content Leak happens.

3.1. What is Content Leak?

For a style transfer algorithm, Content Leak occurs because the stylization results lose some content information. Although the existing state-of-the-art style transfer algorithms, *e.g.*, AdaIN [20], WCT [32], and Avatar-Net [45], can produce good style transfer results, they still suffer from the Content Leak issue. Since it is hard to directly extract the content information from the stylized image and compare it with the input content image, we adopt an alternative way to show empirical evidence of the Content Leak phenomenon. More specifically, we first perform the style transfer with an input content-style pair based on a style transfer algorithm. We then take the stylized image as the new content and repeatedly perform the style transfer process for 20 times. Fig. 1 shows the results of our experiments for AdaIN (row 1), WCT (row 2), and Avatar-Net (row 3). According to Fig. 1, when we perform style transfer for 20 rounds, we can hardly recognize any detail of the content image. Such an empirical evidence indicates that the Content Leak phenomenon occurs in all AdaIN, WCT, and Avatar-Net. In the following, we discuss the causes of the Content Leak, which imply that the Content Leak issue also exists in other state-of-the-art style transfer algorithms.

3.2. Why Does Content Leak Happen?

Taking AdaIN [15], WCT [32], and Avatar-Net [45] as three representatives of style transfer algorithms, we study the causes of the Content Leak phenomenon.

Reconstruction error. A straightforward explanation to Content Leak is that the decoder of existing style transfer algorithms cannot achieve lossless image reconstruction of the input content image. For example, all AdaIN, WCT, and Avatar-Net adopt VGG19 [47] as the encoder and train a structurally symmetrical decoder to invert the features of VGG19 back to the image space. Although an image reconstruction loss [32] or a content loss [20] is used to train the decoder, Li *et al*. [32] acknowledge that the decoder is far from perfect due to the loss of spatial information brought by the pooling operations in the encoder. Consequently, the accumulated image reconstruction error may gradually disturb the content details and lead to the Content Leak.

Biased decoder training. The above-mentioned reconstruction error can only partially explain the Content Leak phenomenon. In addition, biased decoder training is another cause. We take the training scheme of AdaIN as an example to explain how its loss function settings lead to Content Leak. AdaIN trains the decoder with a weighted combination of a content loss L_c and a style loss L_s, where

$$L_c = \|F(G(t)) - t\|_2, \tag{1}$$

$$L_s = \sum_{i=1\ldots L} \|\mu(\phi_i(G(t))) - \mu(\phi_i(s))\|_2 + \sum_{i=1\ldots L} \|\sigma(\phi_i(G(t))) - \sigma(\phi_i(s))\|_2. \tag{2}$$

Here t denotes the output of the adaptive instance normalization, F and G represent the encoder and the decoder, respectively, ϕ_i denotes a layer in VGG19 used to compute the style loss, and μ, σ represent the mean and standard deviation of feature maps, respectively. Due to L_s, the decoder is trained to trade off between L_c and L_s, rather than trying to reconstruct images perfectly. Fig. 3 shows the training loss curves of AdaIN with and without L_s. When we train the decoder of AdaIN with only L_c, the converged value of L_c (cyan curve) is significantly smaller than training with the weighted combination of L_c and L_s (blue curve). Consequently, the auto-encoder of AdaIN is biased towards rendering more artistic effects, which causes Content Leak. Fig. 4 shows the image reconstruction results by propagating through the auto-encoder of AdaIN for 50 rounds. We take the output of the auto-encoder in the previous round as the input of the next round and perform image reconstruction repeatedly. With the increase of the inference rounds, weird artistic patterns gradually appear in the produced results, which indicates that the auto-encoder of AdaIN may memorize image styles in training and bias towards the training styles in inference.

Biased style transfer module. Biased style transfer module is another cause of the Content Leak. We take the Style Decorator in Avatar-Net as an example. For the normalized content feature f_c and style feature f_s, the key mechanism of the Style Decorator is motivated by the deep image analogy [35], which is composed of two steps. In the first step, the algorithm finds a corresponding patch in f_s for every patch in f_c according to the content similarity between two patches. In the next step, f_{cs} is formed by replacing patches in f_c with the corresponding patches in f_s. Since such a patch replacement is irreversible, f_c cannot be recovered from f_{cs}, which makes f_{cs} be biased towards style and consequently causes the Content Leak phenomenon.

We summarize and illustrate three main causes of Content Leak in Fig. 5. While the reconstruction error may disturb the content information in the output image, the biased

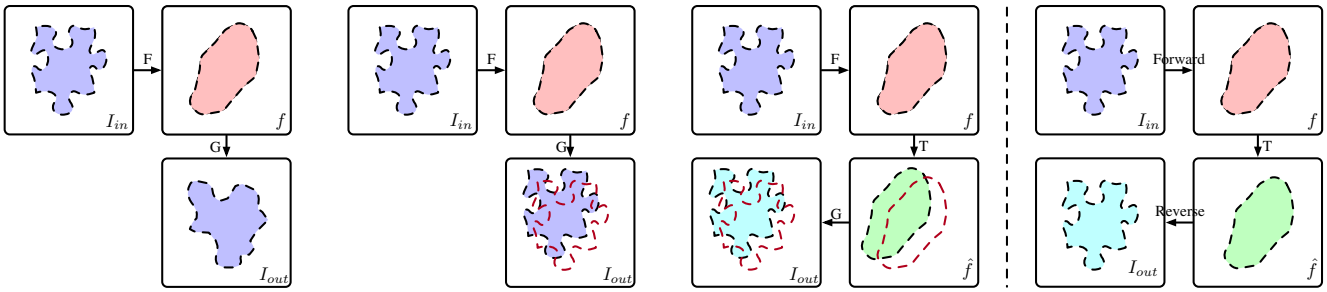


Figure 5. Causes of the Content Leak phenomenon. (a) Reconstruction error, *i.e.*, the content of output image is disturbed. (b) Biased image recovery, *i.e.*, the output image shifts to a biased style via the decoder. (c) Biased style transfer module, *i.e.*, the stylized feature shifts to a biased style via feature stylization. The red dash lines in (b) and (c) denote unbiased positions of the manifolds. (d) The proposed ArtFlow scheme, where both the network and the transfer module are not biased, while the backbone network does not introduce any reconstruction error. Notations – F and G: encoder and decoder used by existing style transfer algorithms, respectively. T: style transfer module, *e.g.*, AdaIN or WCT. I_{in} and I_{out}: input and output images. f and $\hat{f}$: vanilla and stylized deep features of I_{in}.

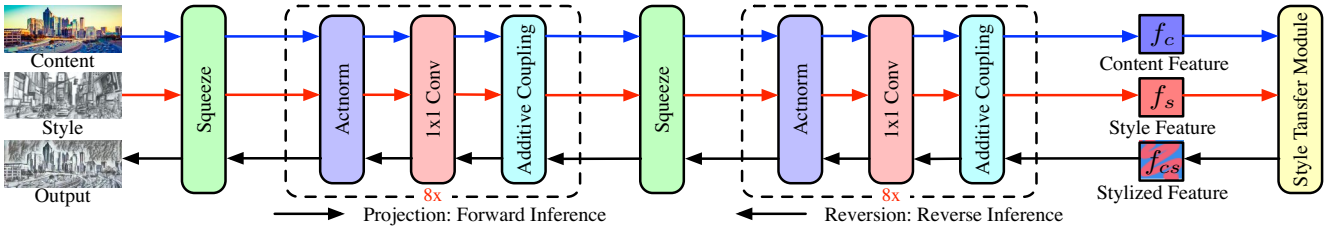


Figure 6. The framework of the proposed ArtFlow. For style transfer, Artflow works in a *projection-transfer-reversion* scheme. *Projection*: Extracting deep features of content and style images via the forward inference. *Transfer*: Transferring the content and style features to the stylized feature via the style transfer module. *Reversion*: Transforming the stylized feature to the stylized image via the reverse inference.

image recovery and the biased transfer module may lead to a style shift in the output image.

4. Method

4.1. Overview of the ArtFlow Framework

In this work, we present a novel unbiased style transfer framework named ArtFlow to address the Content Leak issue of the state-of-the-art style transfer approaches. Different from the *encoder-transfer-decoder* scheme commonly used in existing neural style transfer algorithms, ArtFlow performs image style transfer through a *projection-transfer-reversion* scheme. As shown in Fig. 6, ArtFlow relies on a reversible neural flow model, named Projection Flow Network (PFN). In the *projection* step, the content images and style images are fed into PFN for lossless deep feature extraction via the forward propagation of PFN. In the *transfer* step, the content and style features are transferred to the stylized feature with an unbiased style transfer module. In the *reversion* step, the stylized feature is reconstructed to a stylized image via the reverse propagation of PFN. Since the information flow in PFN and the unbiased style transfer module are both lossless and unbiased, ArtFlow achieves unbiased image style transfer to avoid the Content Leak.

In the following, we first discuss the details of PFN in Section 4.2. Then, we discuss the choice of the unbiased style transfer module by performing both theoretical and quantitative analyses of the inherent biases of existing transfer modules in Section 4.3.

4.2. Projection Flow Network

Projection Flow Network (PFN) serves as both the deep feature extractor and image synthesizer of our ArtFlow framework. In this work, we construct PFN by following the effective Glow model [28]. As shown in Fig. 6, PFN consists of a chain of three learnable reversible transformations, *i.e.*, additive coupling, invertible 1×1 convolution, and Actnorm. All the components of PFN are reversible, making PFN fully reversible that the information is lossless during the forward and reverse propagation. In the following, we describe the three reversible transformations.

Additive coupling. Dinh *et al.* [5, 6] proposed an expressive reversible transformation named affine coupling layer. In this work, we adopt a special case of affine coupling, *i.e.*, additive coupling, for PFN. The forward computation of additive coupling is

$$\begin{aligned} x_a, x_b &= \text{split}(x) \\ y_b &= \text{NN}(x_a) + x_b \\ y &= \text{concat}(x_a, y_b). \end{aligned}$$

The $\mathrm{split}()$ function splits a tensor into two halves along the channel dimension. $\mathrm{NN}()$ is (any) neural network where the input and the output have the same shape. The $\mathrm{concat}()$ function concatenates two tensors along the channel dimension. The reverse computation of additive coupling can be easily derived.

We observe that a flow model with additive coupling layers is sufficient to handle the style transfer task in experiments. Moreover, the additive coupling is more efficient and stable than the affine coupling in model training. Therefore, we employ additive coupling instead of affine coupling as the expressive transformation layer in PFN.

Invertible 1×1 convolution. Since the additive coupling layer only processes a half of the feature maps, it is necessary to permute the channel dimensions of feature maps, so that each dimension can affect all the other dimensions [5, 6]. We follow Glow [28] to use a learnable invertible 1×1 convolution layer for flexible channel permutation, as

$$y_{i,j} = W x_{i,j}. \tag{3}$$

W is the weight matrix of shape $c\times c$, where c is the channel dimension of tensor x and y. Its reverse function is $x_{i,j} = W^{-1} y_{i,j}$.

Actnorm. We follow Glow [28] to use the activation normalization layer (Actnorm) as an alternative to batch normalization [21]. Actnorm performs per-channel affine transformation on tensor x, as

$$y_{i,j} = w \odot x_{i,j} + b, \tag{4}$$

where i, j denote a spatial position on the tensor. w and b are the scale and bias parameters of affine transformation, and they are learnable in model training. The reverse funciton is $x_{i,j} = (y_{i,j} - b)/w$.

In addition to the three reversible transformations, the $\mathrm{squeeze}$ operation is inserted into certain parts of PFN to reduce the spatial size of 2D feature maps. The $\mathrm{squeeze}$ operation splits the features into smaller patches along the spatial dimension and then concatenates the patches along the channel dimension.

4.3. Unbiased Content-Style Separation

Which style transfer module should ArtFlow use to achieve the unbiased style transfer? To answer this question, we first make a theoretical analysis of the biases of two popular style transfer modules, *i.e.*, the adaptive instance normalization in AdaIN, and the whitening and coloring transforms in WCT.

The mechanism of the universal style transfer methods can be regarded as a natural evolution of the bilinear model proposed by Tenenbaum and Freeman in [52], which separates an image into a content factor C and a style factor S and then makes style transfer by replacing the style factor S in the content image with that in the target image. Similarly, the universal style transfer methods assume that the content information and the style information in the deep feature space are disentangled explicitly [20, 32, 31, 40, 2, 1, 56, 22, 60] or implicitly [4, 45]. For example, AdaIN [20] separates deep features into normalized feature maps and mean/std vectors, which can be regarded as the content factor C and style factor S, respectively.

Following the theoretical framework of the Bilinear Model [52], we can define the unbiased style transfer as:

Definition 1 *Suppose we have a bilinear style transfer module $f_{cs} = C(f_c)S(f_s)$, where C, S denote the content factor and the style factor in the bilinear model, respectively. f_{cs} is an unbiased style transfer module if $C(f_{cs}) = C(f_c)$ and $S(f_{cs}) = S(f_s)$.*

Based on Def. 1, we have the following two theorems.

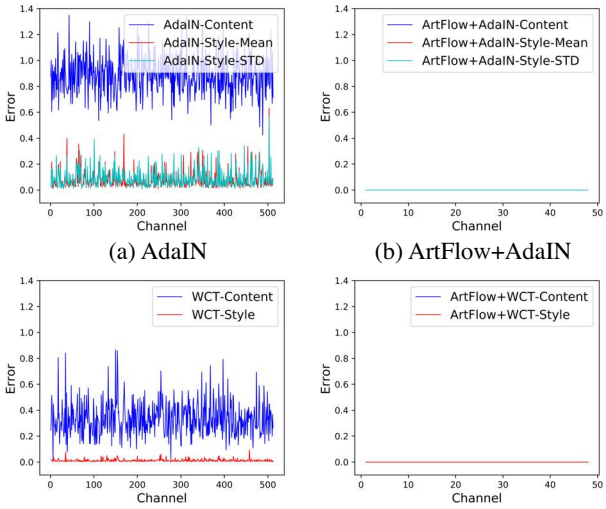
Theorem 1 *The adaptive instance normalization in AdaIN is an unbiased style transfer module.*

Theorem 2 *The whitening and coloring transform in WCT is an unbiased style transfer module.*

The proofs for Theorems 1 and 2 can be found in the supplementary material. The Style Decorator in Avatar-Net [45] does not fit the bilinear model, while the empirical analysis in Sec. 3.2 shows that *Style Decorator is a biased style transfer module*.

In addition to the theoretical analyses, we also quantitatively verify the unbiased property of the transfer modules in AdaIN and WCT. Quantitatively studying the property of popular style transfer modules is an unsolved question because the auto-encoder used by existing universal style transfer methods has significant image reconstruction errors and may be biased towards styles as discussed in Sec. 3.2. Consequently, the produced style transfer results using auto-encoders cannot precisely reflect the effects of the style transfer modules upon deep features. The proposed PFN addresses this issue. Specifically, if we take the forward inference and the reverse inference of the proposed PFN as the encoder and decoder, respectively, we can obtain a lossless and unbiased "auto-encoder" for style transfer, which can avoid the influence of the image reconstruction error and the biased image recovery brought by the decoder.

By using the proposed PFN as the lossless feature projector/inverter, we make a quantitative analysis about the content and style reconstruction errors of the transfer modules in AdaIN and WCT. Fig. 7 demonstrates two findings: 1) Considering (a) *vs*. (b) and (c) *vs*. (d), the proposed PFN can indeed make lossless and unbiased content and style reconstruction while the auto-encoder based on VGG19 cannot.



(a) AdaIN (b) ArtFlow+AdaIN

(c) WCT (d) ArtFlow+WCT

Figure 7. Content error between $F(G(f_{cs}))$ and f_c and the style error between $F(G(f_{cs}))$ and f_s. ArtFlow with the transfer modules of AdaIN/WCT achieves lossless content/style reconstruction.

2) (b) and (d) quantitatively verify that the transfer module of AdaIN and WCT are unbiased.

Based on theoretical and quantitative analyses to transfer modules in AdaIN and WCT, we let the adaptive instance normalization and the whitening and coloring transforms be two options for ArtFlow to achieve unbiased style transfer.

5. Experiments

To demonstrate that ArtFlow can achieve unbiased style transfer, we conduct extensive experiments. We make a comparison between the proposed ArtFlow and state-of-the-art style transfer algorithms in terms of stylization effect, computing time, content leak, and content factor visualization. Moreover, we present a new interesting application named reverse style transfer, which can only be performed by ArtFlow. More qualitative results, portrait style transfer images, and user study results are available in supplementary materials.

5.1. Experimental Settings

Dataset. Following the existing style transfer methods [20, 32], we use the MS-COCO dataset [36] as the content images and the WikiArt dataset [41] as the style images. In training, we resize the input images to 512×512 and randomly crop each image to 256×256.

Network architecture. As shown in Fig. 6, the proposed PFN consists of two flow blocks, where each block contains eight neural flows. Each flow is a stack of an Actnorm layer, an invertible 1×1 convolution, and an additive coupling layer. More studies on the number of blocks and flows can be found in the supplementary material.

Training. We implement our ArtFlow on the PyTorch framework [42]. We train ArtFlow for 60,000 iterations using an Adam optimizer [27] with a batch size of 2, an initial learning rate of 1e-4, and a learning rate decay of 5e-5. The training of ArtFlow takes about 12 hours on a single RTX 2080Ti GPU. We adopt the content loss L_c in Eq. 1 and style loss L_s in Eq. 2 as the training objective of ArtFlow. The loss weights of L_c and L_s are set to 0.1 and 1, respectively.

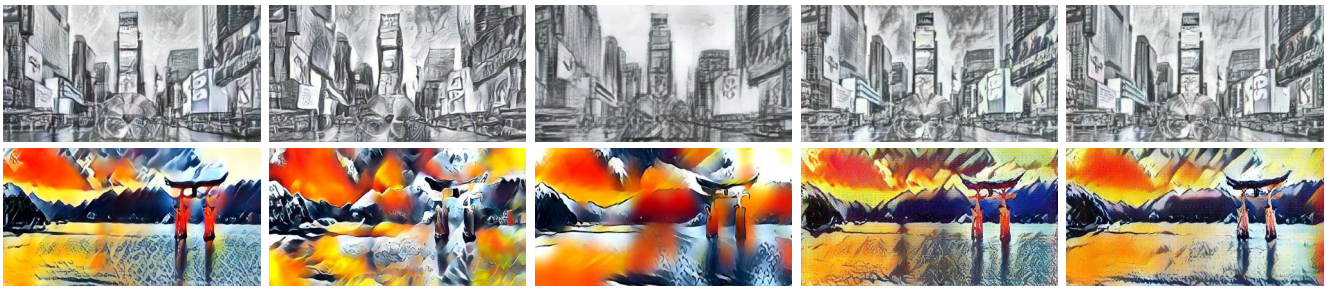
5.2. Style Transfer Results

To demonstrate the style transfer ability of the proposed ArtFlow, we compare the style transfer results of ArtFlow in conjunction with the transfer module of AdaIN/WCT with the state-of-the-art universal style transfer algorithms, *i.e.*, StyleSwap [4], AdaIN [20], WCT [32], LinearWCT [31], OptimalWCT [40], and Avatar-Net [45].

Visual comparison. Fig. 8 shows the style transfer results by all the compared algorithms. While all the compared methods can produce good style transfer results, different methods create distinct artistic effects, *e.g.*, WCT and OptimalWCT can create more colorful artistic effects, LinearWCT, AdaIN, ArtFlow can preserve more content details, and Avatar-Net can render more fine textures. The proposed ArtFlow in conjunction with AdaIN/WCT can produce visually comparable style transfer results to the state-of-the-art style transfer algorithms. It is worth noting that the style transfer results by ArtFlow preserve more details of the content image (please zoom in to compare the details of the billboards in the top row results), which confirms that ArtFlow corrects the biased style transfer issue of the compared methods and avoids the Content Leak. Moreover, we also perform portrait style transfer with the proposed ArtFlow. To train the portrait style transfer model, we use FFHQ [25] as the content and Metfaces [24] as the style. As Fig. 9 shows, ArtFlow can also achieve good artistic style transfer results on portrait images.

Quantitative comparison. In addition to the visual comparison, we also make a quantitative comparison. Inspired by [62], we adopt the Structural Similarity Index (SSIM) and the content loss between the original contents and stylized images as the metric to measure the performance of the content information preservation in style transfer. To measure the ability to create artistic effects of a style transfer algorithm, inspired by [32], we use the mean square error of Gram matrices between the style and the style-transferred images. As Tab. 1 shows, ArtFlow achieves the highest SSIM score, which indicates that the proposed methods have a stronger ability to preserve more content information. While StyleSwap achieves the best content loss and a good SSIM score, its style transfer results do not look as good as the results produced by ArtFlow. Regarding the Gram loss, since ArtFlow mainly addresses the

(a) Content (b) Style (c) StyleSwap (d) AdaIN (e) WCT



(f) LinearWCT (g) OptimalWCT (h) Avatar-Net (i) ArtFlow+AdaIN (j) ArtFlow+WCT

Figure 8. Style transfer results of the state-of-the-art universal style transfer algorithms.

Table 1. Quantitative evaluation results of universal stylization methods. The computing time is evaluated on 512×512 images.

Method	StyleSwap	AdaIN	WCT	LinearWCT	OptimalWCT	Avatar-Net	Self-Contained	ArtFlow+AdaIN	ArtFlow+WCT
SSIM ↑	0.44	0.29	0.27	0.35	0.21	0.31	0.23	0.45	**0.47**
Content Loss ↓	**2.22**	3.10	3.35	2.57	4.33	3.35	3.00	2.58	2.80
Gram Loss ↓	0.00482	0.00127	0.00074	0.00093	**0.00035**	0.00099	0.00473	0.00098	0.00078
Time (seconds) ↓	0.272	**0.064**	0.997	0.092	1.808	9.129	0.156	0.408	0.428

(a) Content (b) Style (c) ArtFlow

Figure 9. Portrait style transfer results using the proposed ArtFlow.

Content Leak issue and corrects the biases towards style images, it is reasonable that ArtFlow does not achieve the lowest Gram loss. It is worth noting that ArtFlow in conjunction with AdaIN achieves a lower Gram loss than vanilla AdaIN while ArtFlow in conjunction with WCT has a similar Gram loss compared with WCT itself, indicating that ArtFlow can solve the Content Leak issue without hurting the stylization ability of AdaIN/WCT. In the third row of Tab. 1, we also show the computing time for all the compared methods. ArtFlow+AdaIN is slower than vanilla AdaIN since PFN does not adopt any pooling operations. Therefore, it requires more computations in the higher layers than AdaIN. Comparing with WCT, since ArtFlow does not need the multi-level stylization strategy used by WCT, ArtFlow+WCT is faster than vanilla WCT.

5.3. Content Leak

As discussed in Sec. 3, if the Content Leak happens, the content information would gradually disintegrate when we perform style transfer repeatedly. To demonstrate that the proposed ArtFlow can avoid the Content Leak phenomenon, we use the above way to visualize and compare the Content Leak phenomenon in AdaIN, WCT, Avatar-Net, and their counterparts in conjunction with the proposed ArtFlow. We also show the result by [9] because it also addresses the content leak issue. As Fig. 10 shows, the Content Leak appears in vanilla AdaIN, WCT, Avatar-Net, and Self-Content [9] when we perform the style transfer for 20 rounds. In contrast, when we replace the VGG19 based auto-encoder with the proposed PFN in AdaIN/WCT, the Content Leak disappears completely, which indicates that ArtFlow in conjunction with AdaIN/WCT can effectively solve the Content Leak problem and therefore achieve unbiased style transfer. Regarding Avatar-Net, as discussed in Sec. 3.2, since the Style Decorator in Avatar-Net is inherently biased towards style, ArtFlow combining the Style

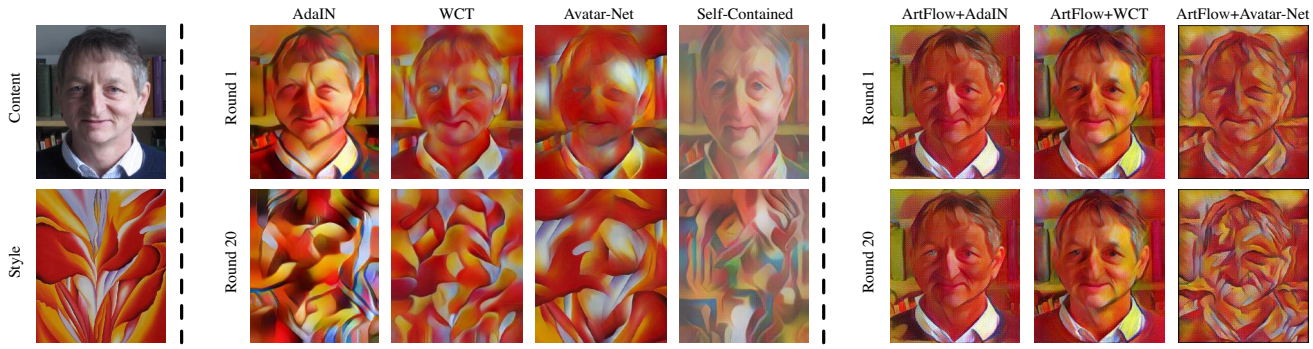
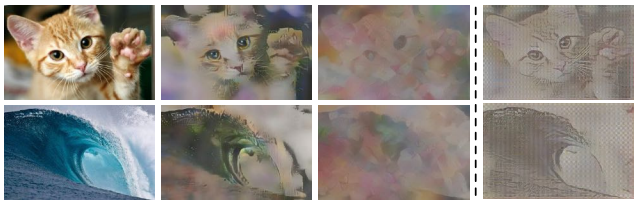


Figure 10. A comparison of the Content Leak phenomenon. We show the style transfer results of the first round and the 20-th round.

Table 2. User study results of universal stylization methods.

Method	StyleSwap	AdaIN	WCT	LinearWCT	OptimalWCT	Avatar-Net	ArtFlow
Votes ↑	7	19	64	257	60	78	**314**



(a) Input (b) AdaIN (c) WCT (d) ArtFlow+AdaIN

Figure 11. Visualization of content features of AdaIN, WCT, and the proposed ArtFlow.

Decorator as the transfer module cannot achieve unbiased style transfer. However, by replacing the auto-encoder with PFN, the Content Leak phenomenon is still significantly alleviated by Avatar-Net.

5.4. Content-Style Separation

As discussed in Sec. 4.3, AdaIN and WCT can be regarded as an evolution of the Bilinear Model in [52]. Taking the view of the bilinear model, the mechanism of AdaIN and WCT can be regarded as: 1) disentangling the content and style factors in the deep feature space, and 2) replacing the style factor of the content image with the style factor of the style image. Since such a disentangled representation of the content and style exists in the feature space, we can visualize the pure content by inverting the content factor back to an image. Fig. 11 shows the inverted content factor in AdaIN, WCT, and ArtFlow in conjunction with AdaIN. Compared with the inverted content factor of AdaIN and WCT, the results by ArtFlow contain significantly less style effects (*e.g.*, colors) along with sharper image structures. Fig. 11 shows that ArtFlow can achieve unbiased content-style separation while AdaIN and WCT cannot.

6. User Study

To quantitatively demonstrate that the proposed ArtFlow has the comparable style transfer performance with the state-of-the-art algorithms, we perform a user study. Our user study is based on the validation dataset that consists of 43 content images and 27 style images. We obtain the style transfer results of StyleSwap, AdaIN, WCT, LinearWCT, OptimalWCT, Avatar-Net, and the proposed ArtFlow on every content-style pair, respectively. We finally obtain 1161 style transfer results for each method. In user study, we list all style transfer results of a content-style pair and let the user to choose ONE most preferable style transfer result. We eventually collect 799 effective votes. Tab. 2 shows the style transfer results. The proposed ArtFlow obtains more votes compared with other style transfer methods, which demonstrates that our method has comparable style transfer performance with the state-of-the-art methods.

7. Conclusion

In this paper, we reveal a common issue in the state-of-the-art style transfer algorithms, *i.e.*, the Content Leak phenomenon. Upon analyzing the main causes of the Content Leak, we present a new style transfer framework named ArtFlow. Unlike the existing style transfer algorithms, which adopt the VGG19 based auto-encoder to extract deep features, ArtFlow introduces a reversible neural flow-based network named PFN, thus enabling both the forward and reverse inferences to project images into the feature space and invert features back to the image space, respectively. ArtFlow in conjunction with an unbiased style transfer module, *e.g.*, either AdaIN or WCT, achieves comparable style transfer results while avoiding the Content Leak phenomenon. Furthermore, because PFN can achieve lossless and unbiased image projection and reversion, the proposed ArtFlow can facilitate a better content-style separation and thus enable the reversion of the style transfer in a lossless manner.

8. Acknowledgement

This work is supported in part by NSF awards IIS-1704337, IIS-1813709, and our corporate sponsors.

References

[1] Jie An, Tao Li, Haozhi Huang, Li Shen, Xuan Wang, Yongyi Tang, Jinwen Ma, Wei Liu, and Jiebo Luo. Real-time universal style transfer on high-resolution images via zero-channel pruning. *arXiv preprint arXiv:2006.09029*, 2020.

[2] Jie An, Haoyi Xiong, Jun Huan, and Jiebo Luo. Ultrafast photorealistic style transfer via neural architecture search. In *AAAI*, 2020.

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

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

[5] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[6] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. In *ICLR*, 2017.

[7] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In *ICLR*, 2017.

[8] Chen et al. Beautyglow: On-demand makeup transfer framework with reversible generative network. In *CVPR*, 2019.

[9] Chen et al. Self-contained stylization via steganography for reverse and serial style transfer. In *WACV*, 2020.

[10] Oriel Frigo, Neus Sabater, Julie Delon, and Pierre Hellier. Split and match: example-based adaptive patch sampling for unsupervised style transfer. In *CVPR*, 2016.

[11] Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. In *NeurIPS*, 2015.

[12] Leon A Gatys, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. Preserving color in neural artistic style transfer. *arXiv preprint arXiv:1606.05897*, 2016.

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

[14] Xinyu Gong, Haozhi Huang, Lin Ma, Fumin Shen, Wei Liu, and Tong Zhang. Neural stereoscopic image style transfer. In *ECCV*, 2018.

[15] Shuyang Gu, Congliang Chen, Jing Liao, and Lu Yuan. Arbitrary style transfer with deep feature reshuffle. In *CVPR*, 2018.

[16] Aaron Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. In *SIGGRAPH*, 1998.

[17] Aaron Hertzmann, Charles E Jacobs, Nuria Oliver, Brian Curless, and David H Salesin. Image analogies. In *SIGGRAPH*, 2001.

[18] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *ICML*, 2019.

[19] Haozhi Huang, Hao Wang, Wenhan Luo, Lin Ma, Wenhao Jiang, Xiaolong Zhu, Zhifeng Li, and Wei Liu. Real-time neural style transfer for videos. In *CVPR*, 2017.

[20] Xun Huang and Serge J Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017.

[21] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015.

[22] Yongcheng Jing, Xiao Liu, Yukang Ding, Xinchao Wang, Errui Ding, Mingli Song, and Shilei Wen. Dynamic instance normalization for arbitrary style transfer. In *AAAI*, 2020.

[23] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016.

[24] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *arXiv preprint arXiv:2006.06676*, 2020.

[25] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019.

[26] Jaehyeon Kim, Sungwon Kim, Jungil Kong, and Sungroh Yoon. Glow-tts: A generative flow for text-to-speech via monotonic alignment search. *arXiv preprint arXiv:2005.11129*, 2020.

[27] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[28] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *NeurIPS*, 2018.

[29] Chuan Li and Michael Wand. Combining markov random fields and convolutional neural networks for image synthesis. In *CVPR*, 2016.

[30] Shaohua Li, Xinxing Xu, Liqiang Nie, and Tat-Seng Chua. Laplacian-steered neural style transfer. In *ACM MM*, 2017.

[31] Xueting Li, Sifei Liu, Jan Kautz, and Ming-Hsuan Yang. Learning linear transformations for fast arbitrary style transfer. In *CVPR*, 2019.

[32] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. In *NeurIPS*, 2017.

[33] Yijun Li, Ming-Yu Liu, Xueting Li, Ming-Hsuan Yang, and Jan Kautz. A closed-form solution to photorealistic image stylization. In *ECCV*, 2018.

[34] Yanghao Li, Naiyan Wang, Jiaying Liu, and Xiaodi Hou. Demystifying neural style transfer. *arXiv preprint arXiv:1701.01036*, 2017.

[35] Jing Liao, Yuan Yao, Lu Yuan, Gang Hua, and Sing Bing Kang. Visual attribute transfer through deep image analogy. In *SIGGRAPH*, 2017.

[36] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755, 2014.

[37] Hongyu Liu, Bin Jiang, Yibing Song, Wei Huang, and Chao Yang. Rethinking image inpainting via a mutual encoder-decoder with feature equalizations. In *ECCV*, 2020.

[38] Xiao-Chang Liu, Xuan-Yi Li, Ming-Ming Cheng, and Peter Hall. Geometric style transfer. *arXiv preprint arXiv:2007.05471*, 2020.

[39] Xuezhe Ma, Xiang Kong, Shanghang Zhang, and Eduard Hovy. Macow: Masked convolutional generative flow. In *NeurIPS*, 2019.

[40] Lu Ming, Zhao Hao, Yao Anbang, Chen Yurong, Xu Feng, and Zhang Li. A closed-form solution to universal style transfer. In *ICCV*, 2019.

[41] K Nichol. Painter by numbers, wikiart. https://www.kaggle.com/c/painter-by-numbers, 2016.

[42] 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 *NIPS Autodiff Workshop*, 2017.

[43] Ryan Prenger, Rafael Valle, and Bryan Catanzaro. Waveglow: A flow-based generative network for speech synthesis. In *ICASSP*, pages 3617–3621, 2019.

[44] Eric Risser, Pierre Wilmot, and Connelly Barnes. Stable and controllable neural texture synthesis and style transfer using histogram losses. *arXiv preprint arXiv:1701.08893*, 2017.

[45] Lu Sheng, Ziyi Lin, Jing Shao, and Xiaogang Wang. Avatar-net: multi-scale zero-shot style transfer by feature decoration. In *CVPR*, 2018.

[46] YiChang Shih, Sylvain Paris, Connelly Barnes, William T Freeman, and Frédo Durand. Style transfer for headshot portraits. *ACM Transactions on Graphics*, 33(4):148, 2014.

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

[48] Yibing Song, Linchao Bao, Shengfeng He, Qingxiong Yang, and Ming-Hsuan Yang. Stylizing face images via multiple exemplars. *CVIU*, 2017.

[49] Yibing Song, Chao Ma, Lijun Gong, Jiawei Zhang, Rynson WH Lau, and Ming-Hsuan Yang. Crest: Convolutional residual learning for visual tracking. In *ICCV*, 2017.

[50] Yibing Song, Jiawei Zhang, Lijun Gong, Shengfeng He, Linchao Bao, Jinshan Pan, Qingxiong Yang, and Ming-Hsuan Yang. Joint face hallucination and deblurring via structure generation and detail enhancement. *IJCV*, 2019.

[51] Yibing Song, Jiawei Zhang, Shengfeng He, Linchao Bao, and Qingxiong Yang. Learning to hallucinate face images via component generation and enhancement. In *IJCAI*, 2017.

[52] Joshua B Tenenbaum and William T Freeman. Separating style and content with bilinear models. *Neural Computation*, 12(6):1247–1283, 2000.

[53] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor S Lempitsky. Texture networks: feed-forward synthesis of textures and stylized images. In *ICML*, 2016.

[54] D Ulyanov, A Vedaldi, and VS Lempitsky. Instance normalization: the missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

[55] Dmitry Ulyanov, Andrea Vedaldi, and Victor S Lempitsky. Improved texture networks: maximizing quality and diversity in feed-forward stylization and texture synthesis. In *CVPR*, 2017.

[56] Huan Wang, Yijun Li, Yuehai Wang, Haoji Hu, and Ming-Hsuan Yang. Collaborative distillation for ultra-resolution universal style transfer. In *CVPR*, 2020.

[57] Ning Wang, Wengang Zhou, Yibing Song, Chao Ma, Wei Liu, and Houqiang Li. Unsupervised deep representation learning for real-time tracking. *IJCV*, 2021.

[58] Xin Wang, Geoffrey Oxholm, Da Zhang, and Yuan-Fang Wang. Multimodal transfer: a hierarchical deep convolutional neural network for fast artistic style transfer. In *CVPR*, 2017.

[59] Yinglong Wang, Yibing Song, Chao Ma, and Bing Zeng. Rethinking image deraining via rain streaks and vapors. In *ECCV*, 2020.

[60] Zhizhong Wang, Lei Zhao, Haibo Chen, Lihong Qiu, Qihang Mo, Sihuan Lin, Wei Xing, and Dongming Lu. Diversified arbitrary style transfer via deep feature perturbation. In *CVPR*, 2020.

[61] Holger Winnemöller, Sven C. Olsen, and Bruce Gooch. Real-time video abstraction. *ACM Transactions on Graphics*, 25(3):1221–1226, 2006.

[62] Jaejun Yoo, Youngjung Uh, Sanghyuk Chun, Byeongkyu Kang, and Jung-Woo Ha. Photorealistic style transfer via wavelet transforms. In *ICCV*, 2019.

[63] Hang Zhang and Kristin Dana. Multi-style generative network for real-time transfer. *arXiv preprint arXiv:1703.06953*, 2017.