Collaborative Distillation for Ultra-Resolution Universal Style Transfer

Huan Wang[1,2*], Yijun Li[3], Yuehai Wang[1], Haoji Hu[1†], Ming-Hsuan Yang[4,5]
[1]Zhejiang University [2]Notheastern University [3]Adobe Research [4]UC Merced [5]Google Research
wang.huan@husky.neu.edu yijli@adobe.com {wyuehai,haoji_hu}@zju.edu.cn mhyang@ucmerced.edu

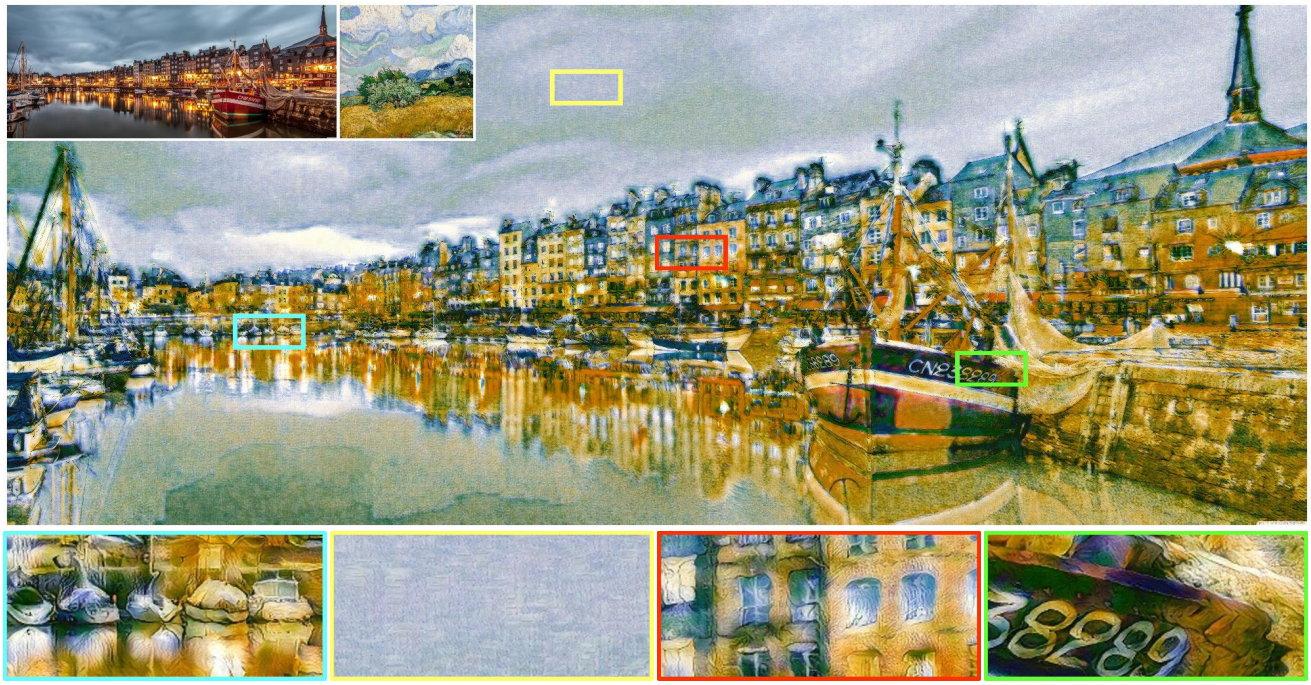


Figure 1: An ultra-resolution stylized sample (10240×4096 pixels), rendered in about 31 seconds on a single Tesla P100 (12GB) GPU. On the upper left are the content and style. Four close-ups (539×248) are shown under the stylized image.

Abstract

Universal style transfer methods typically leverage rich representations from deep Convolutional Neural Network (CNN) models (e.g., VGG-19) pre-trained on large collections of images. Despite the effectiveness, its application is heavily constrained by the large model size to handle ultra-resolution images given limited memory. In this work, we present a new knowledge distillation method (named Collaborative Distillation) for encoder-decoder based neural style transfer to reduce the convolutional filters. The main idea is underpinned by a finding that the encoder-decoder pairs construct an exclusive collaborative relationship, which is regarded as a new kind of knowledge for style transfer models. Moreover, to overcome the feature size mismatch when applying collaborative distillation, a linear embedding loss is introduced to drive the student network to learn a linear embedding of the teacher's features. Extensive experiments show the effectiveness of our method when applied to different universal style transfer approaches (WCT and AdaIN), even if the model size is reduced by 15.5 times. Especially, on WCT with the compressed models, we achieve ultra-resolution (over 40 megapixels) universal style transfer on a 12GB GPU for the first time. Further experiments on optimization-based stylization scheme show the generality of our algorithm on different stylization paradigms. Our code and trained models are available at https://github.com/mingsun-tse/collaborative-distillation.

* This work is mainly done when Huan Wang was with the Department of ISEE, Zhejiang University, China. † Corresponding author.

1. Introduction

Universal neural style transfer (NST) focuses on composing a content image with new styles from any reference image. This often requires a model with considerable capacity to extract effective representations for capturing the statistics of arbitrary styles. Recent universal style transfer methods based on neural networks [13, 4, 24, 39, 40, 37] consistently show that employing the representations extracted by a pre-trained deep neural network like VGG-19 [51] achieves both visually pleasing transferred results and generalization ability on arbitrary style images. However, given limited memory on hardware, the large model size of VGG-19 greatly constrains the input image resolution. Up to now, current universal style transfer approaches only report results around one megapixels (*e.g.*, 1024×1024 pixels) on one single GPU with 12GB memory. Although it is likely to achieve higher resolution style transfer through multiple GPUs, the fundamental problem of the massive model size of VGG-19 remains, hindering NST from practical applications, especially on mobile devices.

Meanwhile, recent years have witnessed rapid development in the area of model compression [17, 21, 36, 19], which aims at reducing the parameters of a large CNN model without considerable performance loss. Despite the progress, most model compression methods only focus on high-level tasks, *e.g.*, classification [18, 58, 56, 48] and detection [63, 19]. Compressing models for low-level vision tasks is still less explored. Knowledge distillation (KD) [2, 1, 21] is a promising model compression method by transferring the knowledge of large networks (called teacher) to small networks (called student), where the knowledge can be softened probability (which can reflect the inherent class similarity structure known as *dark knowledge*) or sample relations (which can reflect the similarity structure among different samples) [44, 46, 60, 47]. This knowledge works as extra information on top of the one-hot labels and hence can boost the student's performance. However, this extra information is mainly *label-dependent*, thus hardly applicable to low-level tasks. What is the dark knowledge in low-level vision tasks (*e.g.*, neural style transfer) remains an open question. Meanwhile, encoder-decoder based models are extensively employed in neural style transfer, where the decoder is typically trained via the knowledge of encoder. Notably, they together construct an *exclusive collaborative relationship* in the stylization process, shown in Fig. 2. Since the decoder D is trained to exclusively work with the encoder E, if another encoder E' can also work with D, it means E' can functionally play the role of E. Based on this idea, we propose a new knowledge to distill the deep models in neural style transfer: the collaborative relationship between the encoder and decoder.

Given a redundant large encoder (*e.g.*, VGG-19), we propose a two-step compression scheme: First, train a collaborator network for the encoder, namely, the decoder in our context; second, replace the large encoder with a small encoder, then train the small encoder with the collaborator *fixed*. Since the small encoder typically has fewer channels, its output feature has a smaller dimension than that of the large encoder. Therefore, the small network cannot directly work with the collaborator. To resolve this, we propose to restrict the student to learn a linear embedding of the teacher's output, so that the teacher's output can be reconstructed through a simple linear combination of the student's output before being fed into the collaborator.

Figure 2: Examples of the exclusive collaboration phenomenon on two different encoder-decoder collaborative relationships: image reconstruction of WCT [39] (Row 1) and style transfer of AdaIN [24] (Row 2). Column 1 is the input, the other four columns show the outputs using different encoder-decoder combinations. If the two encoder-decoder pairs (E1-D1, E2-D2) are trained independently, the encoder can only work with its *matching* decoder.

Notably, we do not restrict the specific collaboration form in our approach. In this paper, we will show it can be applied to two different state-of-the-art universal style transfer schemes: WCT [39] (where the collaboration is image reconstruction) and AdaIN [24] (where the collaboration is style transfer). The main contributions of this work are:

- We propose a new knowledge distillation method for universal neural style transfer. The exclusive collaborative relationship between the encoder and its decoder is identified as a new kind of knowledge, which can be applied to different collaborative relationships.
- To resolve the feature dimension mismatch problem between the student and teacher networks in our algorithm, we propose to restrict the student to learn linear embedding of the teacher's output, which also acts as a regularizer to fuse more supervision into the middle layers of the student so as to boost its learning.
- Extensive experiments show the merits of our method with different stylization frameworks (WCT [39], AdaIN [24], and Gatys [13]), $15.5\times$ parameter reduction with comparable to even better visual effect. Especially, on WCT, the compressed models enable us to conduct ultra-resolution (40 megapixels) universal style transfer for the first time on a single 12GB GPU.

2. Related Work

Style transfer. Prior to the deep learning era, image style transfer is mainly tackled by non-parametric sampling [9], non-photorealistic rendering [15, 52] or image analogy [20]. However, these methods are designed for some specific styles and rely on low-level statistics. Recently, Gatys *et al*. [13] propose the neural style transfer, which employs deep features from the pre-trained VGG-19 model [51] and achieves the stylization by matching the second-order statistics between the generated image and given style image. Numerous methods have been developed to improve the visual quality [34, 57, 49, 64], speed [35, 54, 29, 11, 37], user controls [41, 59, 14], style diversities [8, 3, 38, 24, 39]. However, one common limitation of all those neural network based approaches is that they cannot handle the ultra-resolution content and style images given limited memory. Some approaches [54, 29, 49] achieve high-resolution stylization results (up to 10 megapixels, *e.g*., 3000×3000 pixels) by learning a small feedforward network for a specific style example or category, but they do not generalize to other unseen styles. In contrast, our goal is to realize the ultra-resolution image style transfer for *universal* styles with *one* model only.

Model compression. Model compression and acceleration have also attracted much attention recently, which aim to obtain a smaller and faster model without a considerable compromise in performance. Existing methods broadly fall into five categories, *i.e*., low-rank decomposition [7, 27, 32, 63], pruning [33, 18, 17, 36, 58, 19, 56, 55], quantization [5, 48, 65, 25], knowledge distillation [2, 1, 21, 61] and compact architecture redesign or search [26, 23, 50, 62, 45, 53, 10]. However, these methods are mainly explored in high-level vision tasks, typically the classification and detection. Few approaches have paid attention to low-level vision tasks such as style transfer, where many methods are also limited by the massive model size of CNNs. Unlike the CNN compression for high-level vision where it only needs to maintain the global semantic information of features to retain accuracies, the extra challenge of model compression for low-level vision may be how to maintain the local structures, *e.g*., local textures and color diversity in style transfer.

In this work, we develop a deeply-supervised knowledge distillation method to learn a much smaller model from pre-trained redundant VGG-19 [51]. The compressed model enjoys more than 15× parameter and computation reduction. More importantly, the decrease of model size enables universal style transfer on ultra-resolution images. To our best knowledge, only one recent work [30] employs GAN [16] to learn unpaired style transfer network on ultra-resolution images. However, they achieve this by working on image *subsamples* and then merging them back to a whole image. In contrast, our method fundamentally reduces the model complexity, which can directly process the *whole* image.

3. Proposed Method

3.1. Collaborative distillation

Style-agnostic stylization methods typically adopt an encoder-decoder scheme to learn deep representations for style rendering and then invert them back into the stylized images. Since the style information is not directly encoded in the model, the encoder part needs to be expressive enough to extract informative representations for universal styles. Existing methods [13, 4, 24, 39] commonly choose VGG-19 [51] as the encoder considering its massive capacity and hierarchical architecture. As for the decoder, depending on different stylization schemes, it can have different collaborative relationships with the encoder. Two state-of-the-art arbitrary style transfer approaches, WCT [39] and AdaIN [24], are discussed here. (i) For WCT, the stylization process is to apply Whitening and Coloring Transform [22] to the content features using the second-order statistics of style feature. Then the transformed content feature is inverted to an image by the decoder. Hence, the decoder training is not directly involved with stylization. The collaborative relationship in WCT is essentially image reconstruction. (ii) For AdaIN, unlike WCT, its decoder training is directly involved in the stylization. Two images (content and style) are fed into the encoder, then in the feature space, the content feature is rendered by the statistics (mean and variance) of the style feature. Finally, the decoder inverts the rendered content feature back to stylized images. The stylized images are supposed to be close to the content (or style) in terms of content (or style) distance. Therefore, the collaborative relationship for AdaIN is style transfer.

Despite paradigm difference for the above two schemes, they are both encoder-decoder based, and the decoder is trained through the knowledge of the encoder. This means, during the training of the decoder, the knowledge of the encoder is leaked into the decoder. Presumably and confirmed empirically, the decoder D can only work with its *matching* encoder E like a nut with its bolt. For another encoder E', even if it has the same architecture as E, D and E' cannot work together (see Fig. 2). This exclusivity shows the decoder has some inherent information specific to its encoder. If we can find a way to make the network E' compatible with D too, it means E' can functionally replace the original encoder E. If E' is meanwhile much smaller than E, then we achieve the model compression goal. Based on this idea, we propose a new distillation method specific to NST, named *Collaborative Distillation*, consisting of two steps.

For the first step, based on the task at hand, we train a collaborator network for the large encoder. As shown in Fig. 3(a), for WCT [39], the decoder is trained to invert the feature to be as faithful to the input image as possible (*i.e*.,

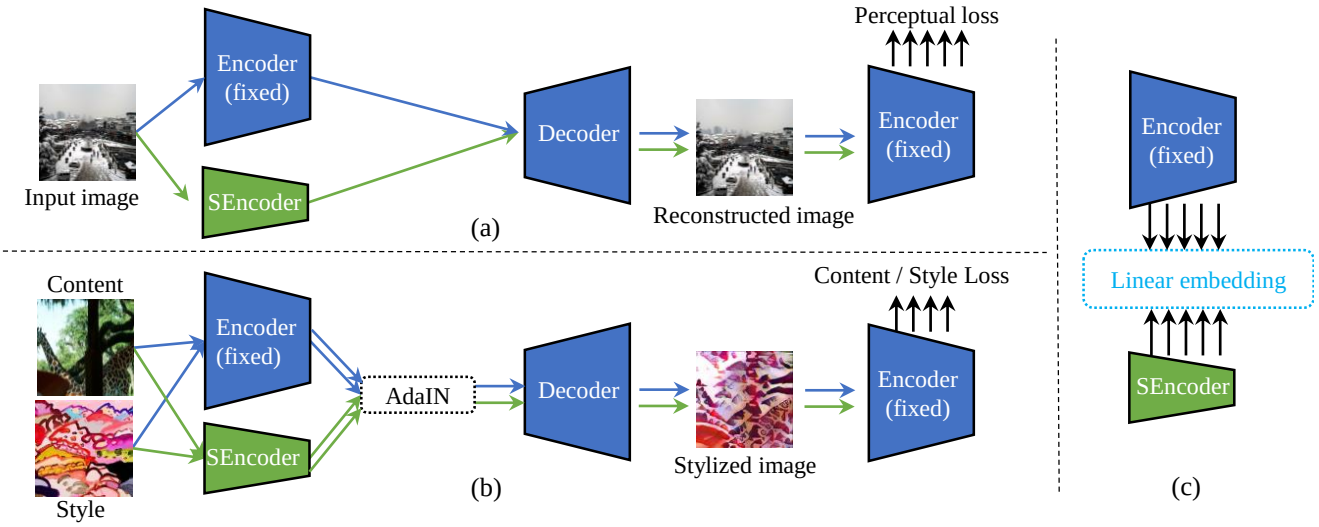


Figure 3: Illustration of the proposed Collaborative Distillation framework (best viewed in color). (a) and (b) depict two kinds of the encoder-decoder collaborative relationship for universal neural style transfer: image reconstruction for WCT [39] and style transfer for AdaIN [24], respectively. Blue arrows show the forward path when training the collaborator network (namely, the decoder). Green arrows show the forward path when the small encoder ("SEncoder") is trained to functionally replace the original encoder ("Encoder"). (c) shows the proposed linear embedding scheme to resolve the feature size mismatch problem and infuse more supervision into the middle layers of the small encoder.

image reconstruction), where both the pixel reconstruction loss and the perceptual loss [29] are employed,

$$\mathcal{L}_r^{(k)} = \|\mathcal{I}_r - \mathcal{I}_o\|_2^2 + \lambda_p \sum_{i=1}^{k} \|\mathcal{F}_r^{(i)} - \mathcal{F}_o^{(i)}\|_2^2, \quad (1)$$

where $k \in \{1, 2, 3, 4, 5\}$ denotes the kth stage of VGG-19; $\mathcal{F}^{(i)}$ denotes the feature maps of the `ReLU_i_1` layer; λ_p is the weight to balance the perceptual loss and pixel reconstruction loss; $\mathcal{I}_o$ and $\mathcal{I}_r$ denote the original image and reconstructed image, respectively. For AdaIN [24], the decoder is involved in style transfer directly. Thus, its decoder loss is made up of both the content loss and the style loss,

$$\mathcal{L}_{\text{st}} = \|\mathcal{F}_{\text{st}}^{(4)} - \mathcal{F}_c^{(4)}\|_2^2 + \lambda_s \sum_{i=1}^{4} \|\mathcal{G}_{\text{st}}^{(i)} - \mathcal{G}_s^{(i)}\|_2^2, \quad (2)$$

where $\mathcal{G}$ is the Gram matrix to describe style [12, 13], λ_s is the weight to balance the style loss and the content loss; subscript "st", "c", and "s" represent the stylized image, content image, and style image, respectively.

After we have the collaborator, the second step of our algorithm is to replace the original encoder E with a small encoder E'. For simplicity, in this work we use E' with the same architecture of E but fewer filters in each layer. We expect that the small encoder E' can functionally equivalent to the original encoder E, as shown in (a) and (b) of Fig. 3. Similar to the first step, the collaborator network loss, denoted as $\mathcal{L}_{\text{collab}}$, can take different forms depending on the specific collaboration tasks. In the context of this work, $\mathcal{L}_{\text{collab}} = \mathcal{L}_r$ for WCT and $\mathcal{L}_{\text{collab}} = \mathcal{L}_{\text{st}}$ for AdaIN.

3.2. Linear embedding

In the proposed collaborative distillation method, the small encoder is connected with the decoder network. In their interface comes a *feature size mismatch* problem. Specifically, if the original encoder outputs a feature of size $C \times H \times W$, and thus the input of the decoder is also supposed to be of size $C \times H \times W$. However, because the small encoder has fewer filters, it will output a feature of size $C' \times H \times W$ ($C' < C$), which cannot be accepted by the decoder. To resolve this, we first look at how the channel numbers play a role in the stylization process. As pioneered by Gatys [12, 13], the style of an image is described by the gram matrix of the deep features extracted from VGG-19,

$$\mathcal{G} = \mathcal{F} \cdot \mathcal{F}^T, \quad (3)$$

where $\mathcal{F}$ is the deep feature of size $C \times HW$ extracted from certain convolutional layer of VGG-19; $\mathcal{G}$ denotes the Gram matrix of size $C \times C$; T stands for matrix transpose. Since we aim to compress these features, *i.e.*, they are regarded as redundant, it can be formulated as that $\mathcal{F}$ is a linear combination of some feature basis vectors in a lower dimension,

$$\mathcal{F} = Q \cdot \mathcal{F}', \quad (4)$$

where Q is a transform matrix of size $C \times C'$, $\mathcal{F}'$ is the feature basis matrix of size $C' \times HW$, which can be viewed as the linear embedding of the original deep feature $\mathcal{F}$. Then it is easy to see that the Gram matrix $\mathcal{G}' = \mathcal{F}' \cdot \mathcal{F}'^T$ for the new feature $\mathcal{F}'$ has the same number of eigenvalues as

the original Gram matrix $\mathcal{G}$. In other words, the style description power is maintained if we adopt $\mathcal{F}'$ in place of the original redundant $\mathcal{F}$. In the context of our method, $\mathcal{F}$ is the output of the original encoder, $\mathcal{F}'$ is the output of the small encoder. The transformation matrix Q is learned through a fully-connected layer *without non-linear activation function* to realize the linearity assumption. Hence, the linear embedding loss can be formulated as

$$\mathcal{L}_{\text{embed}} = ||\mathcal{F} - Q \cdot \mathcal{F}'||_2^2. \tag{5}$$

One more step forward, the proposed solution above is not limited to the final output layer. For the middle layers of the small encoder, it can also find an application. Concretely, we apply the linear embedding to the other four middle layers (`ReLU_k_1`, $k \in \{1, 2, 3, 4\}$) between the original encoder and the small encoder, as shown in Fig. 3(c). We have two motivations for this. First, in the proposed method, when the small encoder is trained with the SGD algorithm, its only gradient source would be the decoder, passed through the fully-connected layer Q. However, Q does not have many parameters, so it will actually form an information bottleneck, slowing down the learning of the student. With these branches plugged into the middle layers of the network, they will infuse more gradients into the student and thus boost its learning, especially for deep networks that are prone to gradient vanishing. Second, in neural style transfer, the style of an image is typically described by the features of *many* middle layers [13, 24, 39]. Therefore, adding more supervision to these layers is necessary to ensure that they do not lose much the style description power for subsequent use in style transfer.

To this end, the total loss to train the small encoder in our proposed algorithm can be summarized as

$$\mathcal{L}_{\text{total}} = \beta \sum_{i=1}^{k} \mathcal{L}_{\text{embed}} + \mathcal{L}_{\text{collab}}, \tag{6}$$

where β is the weight factor to balance the two loss terms.

4. Experimental Results

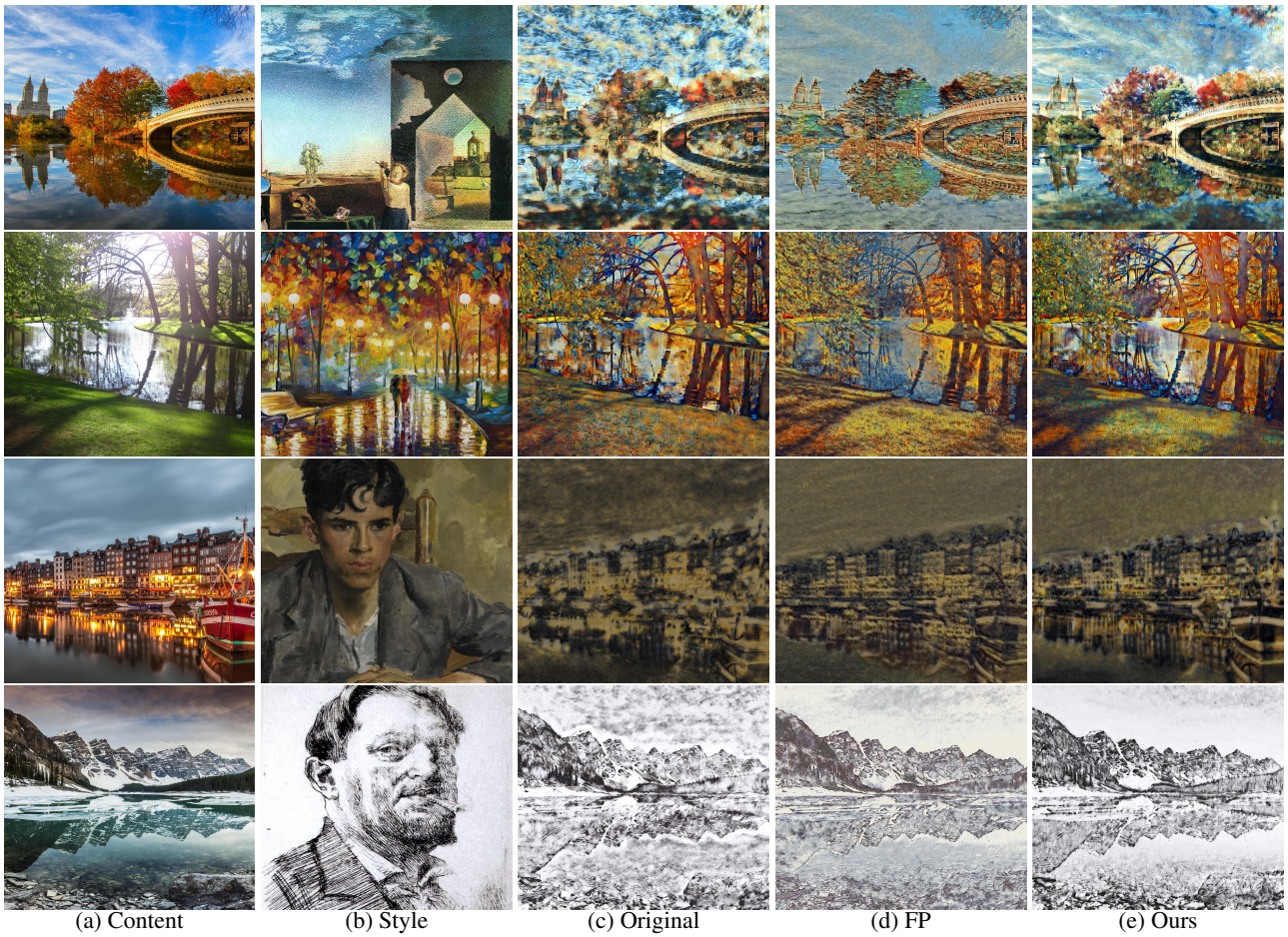
In this section, we will first demonstrate the effectiveness of our compressed VGG-19 compared with the original VGG-19 within the universal style transfer frameworks WCT [39]. Then we show the proposed collaborative distillation is not limited to one kind of collaborative relationship on AdaIN [24]. Finally, to show the generality of the proposed approach, we also evaluate on the optimization-based style transfer using Gatys [13], where the collaborative relationship is the same as WCT, *i.e.*, image reconstruction. We first conduct comparisons in the maximal image resolution which the original VGG-19 can possibly handle (3000×3000), then show some stylized samples in larger resolutions (*i.e.*, ultra-resolutions) with the small models. All experiments are conducted on one Tesla P100 12GB GPU, namely, given the same limited memory.

Evaluated compression method. Since there are few model compression methods specifically designed for low-level image synthesis tasks, we compare our method with a typical compression algorithm in the high-level image classification task, *i.e.*, filter pruning (FP) [36]. Specifically, we first apply FP to VGG-19 in classification to obtain a compressed model with the same architecture as ours. It is then fine-tuned on ImageNet [6] to regain performance. Finally, its decoder is obtained by optimizing the loss in (1).

4.1. Style transfer on WCT

Since we need the original decoder as collaborator , we first train a decoder with the mirrored architecture of VGG-19 on the MS-COCO dataset [42] for image reconstruction. During the training, the encoder is *fixed*, with $\lambda_p = 1$. We randomly crop 256×256 patches from 300×300 images as input. Adam [31] is used as optimization solver with fixed learning rate 10^{-4}, batch size 16. In WCT [39], a cascaded coarse-to-fine stylization procedure is employed for best results, so the decoders of the 5-stage VGG-19 (up to `ReLU_k_1`, $k \in \{1, 2, 3, 4, 5\}$) are all trained. Then an encoder-decoder network is constructed with loss (6), in which β is set to 10. The compressed encoder can be randomly initialized, but we empirically find using the largest filters (based on L_1-norms) from the original VGG-19 as initialization will help the compressed model converge faster, so we use this initialization scheme in all our experiments. After 20-epoch training, we obtain the compressed encoders. Their mirrored decoders are trained by the same rule as the first step using the loss (1).

Fig. 4 shows the comparison of the stylized results. Generally, our model achieves comparable or even more visually pleasing stylized images with much fewer parameters. Stylized images produced by original VGG-19 model and our compressed model are fairly better-looking (more colorful and sharper) than those by the FP-slimmed model. Compared with the original VGG-19 model, our compressed model tends to produce results with *fewer* messy textures, while the original model often highlights too many textures in a stylized image. For example, sky and water usually look smooth in an image, but actually, there are no absolutely smooth parts in natural images, so there are still nuances in the sky and water area. In Fig. 5, the original VGG-19 model tends to highlight these nuances to so obvious an extent that the whole image looks messy, while our model only emphasizes the most semantically salient parts. This phenomenon can be explained since a model with fewer parameters has limited capacity, which is less prone to overfitting. It is natural that an over-parameterized model will spend its extra capacity fitting the noises in data, *e.g.*, the nuanced textures of the sky and water in this con-

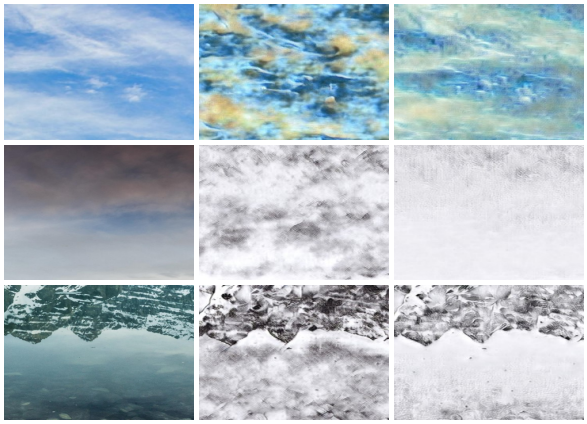


(a) Content (b) Style (c) Original (d) FP (e) Ours

Figure 4: Comparison of stylized images in 3000×3000 produced by three different models, *i.e.*, the original VGG-19, FP-slimmed VGG-19 and our compressed VGG-19 (best viewed in color and zoomed in).

text. Meanwhile, even though a compressed model tends to be less overfitting, the model pruned by FP [36] has a problem of losing description power, which is embodied in two aspects in Fig. 4. First, as we see, FP-slimmed model tends to produce the stylized images with *less* color diversity. Second, when looked closer, the stylized images by FP-slimmed model have serious checkerboard artifacts.

User study. Style transfer has an open problem of lacking broadly-accepted comparison criteria [28], mainly because stylization is quite subjective. For a more objective comparison, previous works [39] conducted user studies to investigate user preference over different stylized results. Here we adopt this idea by investigating which is the most visually pleasing among the stylized images produced by the three models. We generate 20-pair stylized images using the three models. Among them, 10 pairs are randomly selected for each subject to choose which one is the most visually pleasing. In this part, we received 600 valid votes. The user study results are shown in Tab. 1, where the stylized images by our compressed model are top-rated on average, in line with the qualitative comparison in Fig. 4.



(a) Content (b) Original (c) Ours

Figure 5: Stylization detail comparison of the nuanced textures (*e.g.*, the sky and water area) in content images, using the original model and our compressed model.

Style distance. To further quantitatively evaluate the three models, we explore the style similarity between the stylized image and the style image. Intuitively, a more successfully stylized image should be closer to the style image in the

Table 1: User study of preference over different deep models within three universal artistic style transfer frameworks.

Stylization scheme	Original	FP-slimmed	Ours
WCT [39]	33.0%	24.3%	**42.7%**
AdaIN [24]	**51.1%**	6.5%	42.4%
Gatys [13]	**46.5%**	22.3%	31.2%

Table 2: Style distance comparison among three models. Values are the style distances defined as Eq. (7) based on the 5-stage deep features (smaller is better).

Model	Conv1	Conv2	Conv3	Conv4	Conv5
Original	**25.5**	44.6	**236.3**	465.5	476.5
FP-slimmed	43.8	64.3	416.4	597.8	482.1
Ours	28.1	**43.8**	269.7	**446.4**	**399.0**

style space, so we define *style distance* as

$$\mathcal{D}^{(k)}_{\text{style}} = \|\mathcal{G}(\mathcal{F}^{(k)}(\mathcal{I}_{\text{stylized}})) - \mathcal{G}(\mathcal{F}^{(k)}(\mathcal{I}_{\text{style}}))\|_2, \quad (7)$$

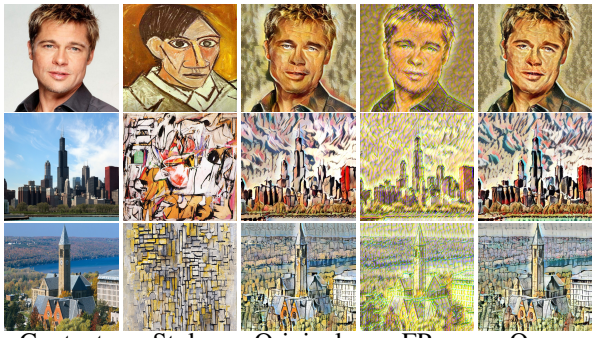
where $\mathcal{G}$ is the Gram matrix. Particularly, we feed the 20-pair stylized images in the user study to the original VGG-19 to extract features of the 5-stage layers (`ReLU_k_1`, $k \in \{1, 2, 3, 4, 5\}$), then the style distances are calculated based on these features. Tab. 2 shows that the stylized images generated by the original and our compressed models are significantly closer to the style images than those by the FP-slimmed model. Our model is fairly comparable with the original one, which agrees with the user preference (Tab. 1).

Aside from the high-resolution stylized images shown above, we show an ultra-resolution stylized image in Fig. 1. The shapes and textures are still very clear even zoomed in on a large scale. To our best knowledge, this is the *first* time that we can obtain ultra-resolution universal style transfer results using a single 12GB GPU. We also report the statistics about the model size and speedup in Tab. 3. Our compressed model is $15.5\times$ smaller and $3.6\times$ faster than the original VGG-19 on GPU. Note that the total size for all the 5-stage models is only 10MB, easy to fit in mobile devices.

Ablation study. Here we explore the effect of the two proposed sub-schemes, collaborative distillation ($\mathcal{L}_{\text{collab}}$) and linear embedding ($\mathcal{L}_{\text{embed}}$). The ablation results in Fig. 6 show that, both the linear embedding and collaboration losses can transfer considerable knowledge from the teacher to the student. Generally, both schemes can independently produce fairly comparable results as the original models even if $15.5\times$ smaller. Between the two losses, $\mathcal{L}_{\text{embed}}$ only transfers the information of feature domain rather than image domain, so its results have more distortion and also some checkerboard effects (please zoom in and see the details of Fig. 6(c)). Meanwhile, $\mathcal{L}_{\text{collab}}$ mainly focuses on the image reconstruction, so it produces sharper results with

(a) C/S (b) Original (c) $\mathcal{L}_{\text{embed}}$ (d) $\mathcal{L}_{\text{collab}}$ (e) Both

Figure 6: Ablation study of the two proposed losses on WCT. (a) Content and style. (b) Use the original models. (c) Use $\mathcal{L}_{\text{embed}}$. (d) Use $\mathcal{L}_{\text{collab}}$. (e) Both losses are used.



Content Style Original FP Ours

Figure 7: Comparison of the three models with AdaIN.

less distortion, which confirms our intuition that the decoder has considerable knowledge which we can leverage to train a small encoder. When both losses employed, we obtain the sweet spot of the two: The results do not have checkerboards, and also properly maintain the artistic distortion.

4.2. Style transfer on AdaIN

We further evaluate the proposed method on AdaIN, where the encoder-decoder collaboration task is style transfer. The training process and network architectures are set the same way as Sec. 4.1, except that the $\mathcal{L}_{\text{st}}$ (Eq. 2) is now utilized as the collaborator loss. λ_s is set to 10 and β is set to 10. The results are shown in Fig. 7, where our compressed encoder produces visually comparable results with the original one, while FP-slimmed model degrades the visual effect significantly. This visual evaluation is further justified by the user study in Tab. 1, where ours and the original model receive similar votes, significantly more than those of the model compressed by the FP algorithm [36].

4.3. Optimization-based style transfer

Although the work of [13] is not encoder-decoder based, it is worthwhile to check whether the small encoder compressed by our method can still perform well in this case. Layers of `Conv_k_1` ($k \in \{1, 2, 3, 4, 5\}$) are selected for style matching and layer `Conv4_2` selected for content matching. The L-BFGS [43] is used as the optimization solver. Fig. 8 shows that the compressed model can achieve

Table 3: Summary of the original models and our compressed models. Storage is measured in PyTorch model. GFLOPs and Inference time are measured when the content and style are both RGB images of 3000×3000 pixels. Max input resolution is measured on GPU using PyTorch implementation, with content and style are *square* RGB images of the same size.

Model	# Params (10^6)	Storage (MB)	# GFLOPs	Inference time (GPU/CPU, s)	Max resolution
Original	17.1	66.6	6961.7	31.2/937.7	3000×3000
Ours	1.1(15.5×)	5.2(12.8×)	451.6(15.4×)	8.6(3.6×)/366.0(2.6×)	6000×6000

Figure 8: Comparison of stylized images using the method by Gatys *et al*. [13], with the original model and our compressed model (best viewed in color and zoomed in).

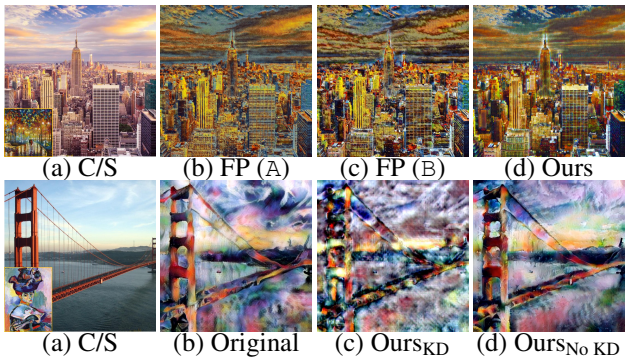


Figure 9: Row 1: Stylization comparison of FP-pruned model A and B. Row 2: Stylization comparison between decoders using and not using KD with the WCT framework.

fairly comparable results with those generated by the original VGG-19. Similar to the experiments of artistic style transfer in Sec. 4.1, we also conduct a user study for a more objective comparison. Results in Tab. 1 show that our compressed model is slightly less popular than the original model (which is reasonable considering that the compressed model has $15.5\times$ fewer parameters), while still more preferred than the model compressed by the FP method.

5. Discussions

We briefly explain why existing knowledge distillation methods [44, 46, 60, 47] generally work less effectively for style transfer. The FP-pruned VGG-19 (called A) in our paper achieves a top-5 accuracy of 83.47% on ImageNet. We had an intermediate model (called B) in the middle of the fine-tuning process of A, with a lower top-5 accuracy 80.55%. We compare the stylization quality of A and B with WCT and find that the results of A does *not* show any advantage over those of B as the unpleasant messy textures still remain (Fig. 9, Row 1). This implies that a small accuracy gain in classification (typically less than 3%, which is what the state-of-the-art distillation methods [44, 46, 60, 47] can achieve at most) cannot really translate to the *perceptual* improvement in neural style transfer. In addition, we only apply the proposed method to distilling the encoder, *not the decoder*, because it will simply degrade the visual quality otherwise, as shown in Fig. 9 (Row 2). The reason is that, the decoder is responsible for image reconstruction, which is already trained with proper supervisions, *i.e*., the pixel and perceptual losses in Eq. (1). When applying distillation to the small decoder, the extra supervision from the original decoder does not help but undermining the effect of loss (1), thus deteriorating the visual quality of stylized results.

6. Conclusion

Input resolution is an important limitation for universal neural style transfer due to the large model size of CNNs. In this work, we present a new knowledge distillation method (*i.e*., Collaborative Distillation) to reduce the model size of VGG-19, which exploits the phenomenon that encoder-decoder pairs in universal style transfer construct an exclusive collaborative relationship. To resolve the feature size mismatch problem, a linear embedding scheme is further proposed. Extensive experiments show the merits of our method on two universal stylization approaches (WCT and AdaIN). Further experiments within the Gatys stylization framework demonstrate the generality of our approach on the optimization-based style transfer paradigm. Although we mainly focus on neural style transfer in this work, the encoder-decoder scheme is also generally utilized in other low-level vision tasks like super-resolution and image inpainting. The performance of our method on these tasks is worth exploring, which we leave as the future work.

Acknowledgment

We thank Wei Gao and Lixin Liu for helpful discussions. This work is supported in part by the Natural Key R&D Program of China under Grant No.2017YFB1002400 and US National Science Foundation CAREER Grant No.1149783.

References

[1] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *NeurIPS*, 2014. 2, 3

[2] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *SIGKDD*, 2006. 2, 3

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

[4] Tian Qi Chen and Mark Schmidt. Fast patch-based style transfer of arbitrary style. *arXiv preprint arXiv:1612.04337*, 2016. 2, 3

[5] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or -1. *arXiv preprint arXiv:1602.02830*, 2016. 3

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 5

[7] Emily L Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In *NeurIPS*, 2014. 3

[8] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In *ICLR*, 2017. 3

[9] Alexei A Efros and Thomas K Leung. Texture synthesis by non-parametric sampling. In *ICCV*, 1999. 3

[10] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *JMLR*, 20(55):1–21, 2019. 3

[11] Wei Gao, Yijun Li, Yihang Yin, and Ming-Hsuan Yang. Fast video multi-style transfer. In *WACV*, 2020. 3

[12] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. In *NeurIPS*, 2015. 4

[13] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *CVPR*, 2016. 2, 3, 4, 5, 7, 8

[14] Leon A Gatys, Alexander S Ecker, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. Controlling perceptual factors in neural style transfer. In *CVPR*, 2017. 3

[15] Bruce Gooch and Amy Gooch. *Non-photorealistic rendering*. AK Peters/CRC Press, 2001. 3

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 3

[17] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In *ICLR*, 2016. 2, 3

[18] Song Han, Jeff Pool, John Tran, and William J Dally. Learning both weights and connections for efficient neural network. In *NeurIPS*, 2015. 2, 3

[19] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *ICCV*, 2017. 2, 3

[20] Aaron Hertzmann, Charles E Jacobs, Nuria Oliver, Brian Curless, and David H Salesin. Image analogies. In *SIGGRAPH*, 2001. 3

[21] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2, 3

[22] Miliha Hossain. Whitening and coloring transforms for multivariate gaussian random variables. *Project Rhea*, 2016. 3

[23] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 3

[24] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017. 2, 3, 4, 5, 7

[25] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *JMLR*, 18(1):6869–6898, 2017. 3

[26] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and $<$ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016. 3

[27] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. Speeding up convolutional neural networks with low rank expansions. In *BMVC*, 2014. 3

[28] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, and Mingli Song. Neural style transfer: A review. *arXiv preprint arXiv:1705.04058*, 2017. 6

[29] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016. 3, 4

[30] Andrej Junginger, Markus Hanselmann, Thilo Strauss, Sebastian Boblest, Jens Buchner, and Holger Ulmer. Unpaired high-resolution and scalable style transfer using generative adversarial networks. *arXiv preprint arXiv:1810.05724*, 2018. 3

[31] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 5

[32] Vadim Lebedev, Yaroslav Ganin, Maksim Rakhuba, Ivan Oseledets, and Victor Lempitsky. Speeding-up convolutional neural networks using fine-tuned cp-decomposition. *arXiv preprint arXiv:1412.6553*, 2014. 3

[33] Yann LeCun, John S Denker, and Sara A Solla. Optimal brain damage. In *NeurIPS*, 1990. 3

[34] Chuan Li and Michael Wand. Combining markov random fields and convolutional neural networks for image synthesis. In *CVPR*, 2016. 3

[35] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *ECCV*, 2016. 3

[36] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *ICLR*, 2017. 2, 3, 5, 6, 7

[37] Xueting Li, Sifei Liu, Jan Kautz, and Ming-Hsuan Yang. Learning linear transformations for fast arbitrary style transfer. In *CVPR*, 2019. 2, 3

[38] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Diversified texture synthesis with feed-forward networks. In *CVPR*, 2017. 3

[39] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. In *NeurIPS*, 2017. 2, 3, 4, 5, 6, 7

[40] Yijun Li, Ming-Yu Liu, Xueting Li, Ming-Hsuan Yang, and Jan Kautz. A closed-form solution to photorealistic image stylization. In *ECCV*, 2018. 2

[41] Jing Liao, Yuan Yao, Lu Yuan, Gang Hua, and Sing Bing Kang. Visual attribute transfer through deep image analogy. *ACM Transactions on Graphics*, 36(4):1–15, 2017. 3

[42] 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*, 2014. 5

[43] Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528, 1989. 7

[44] Yufan Liu, Jiajiong Cao, Bing Li, Chunfeng Yuan, Weiming Hu, Yangxi Li, and Yunqiang Duan. Knowledge distillation via instance relationship graph. In *CVPR*, 2019. 2, 8

[45] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *ECCV*, 2018. 3

[46] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *CVPR*, 2019. 2, 8

[47] Baoyun Peng, Xiao Jin, Jiaheng Liu, Dongsheng Li, Yichao Wu, Yu Liu, Shunfeng Zhou, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In *ICCV*, 2019. 2, 8

[48] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV*, 2016. 2, 3

[49] Artsiom Sanakoyeu, Dmytro Kotovenko, Sabine Lang, and Björn Ommer. A style-aware content loss for real-time hd style transfer. In *ECCV*, 2018. 3

[50] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018. 3

[51] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2, 3

[52] Thomas Strothotte and Stefan Schlechtweg. *Non-photorealistic computer graphics: modeling, rendering, and animation*. Morgan Kaufmann, 2002. 3

[53] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. 3

[54] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *ICML*, 2016. 3

[55] Huan Wang, Xinyi Hu, Qiming Zhang, Yuehai Wang, Lu Yu, and Haoji Hu. Structured pruning for efficient convolutional neural networks via incremental regularization. *JSTSP*, 2019. 3

[56] Huan Wang, Qiming Zhang, Yuehai Wang, and Haoji Hu. Structured probabilistic pruning for convolutional neural network acceleration. In *BMVC*, 2018. 2, 3

[57] 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. 3

[58] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *NeurIPS*, 2016. 2, 3

[59] Pierre Wilmot, Eric Risser, and Connelly Barnes. Stable and controllable neural texture synthesis and style transfer using histogram losses. *arXiv preprint arXiv:1701.08893*, 2017. 3

[60] Lu Yu, Vacit Oguz Yazici, Xialei Liu, Joost van de Weijer, Yongmei Cheng, and Arnau Ramisa. Learning metrics from teachers: Compact networks for image embedding. In *CVPR*, 2019. 2, 8

[61] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017. 3

[62] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *CVPR*, 2018. 3

[63] X. Zhang, J. Zou, K. He, and J. Sun. Accelerating very deep convolutional networks for classification and detection. *TPAMI*, 38(10):1943–1955, 2016. 2, 3

[64] Yulun Zhang, Chen Fang, Yilin Wang, Zhaowen Wang, Zhe Lin, Yun Fu, and Jimei Yang. Multimodal style transfer via graph cuts. In *ICCV*, 2019. 3

[65] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016. 3