Playing Lottery Tickets in Style Transfer Models

Meihao Kong
Nanjing University
Nanjing, China
kongmeihao@smail.nju.edu.cn

Jing Huo*
Nanjing University
Nanjing, China
huojing@nju.edu.cn

Wenbin Li
Nanjing University
Nanjing, China
liwenbin@nju.edu.cn

Jing Wu
Cardiff University
Cardiff, United Kingdom
wuj11@cardiff.ac.uk

Yu-Kun Lai
Cardiff University
Cardiff, United Kingdom
laiy4@cardiff.ac.uk

Yang Gao
Nanjing University
Nanjing, China
gaoy@nju.edu.cn

ABSTRACT

Style transfer has achieved great success and attracted a wide range of attention from both academic and industrial communities due to its flexible application scenarios. However, the dependence on a pretty large VGG-based autoencoder leads to existing style transfer models having high parameter complexities, which limits their applications on resource-constrained devices. Compared with many other tasks, the compression of style transfer models has been less explored. Recently, the lottery ticket hypothesis (LTH) has shown great potential in finding extremely sparse matching subnetworks which can achieve on par or even better performance than the original full networks when trained in isolation. In this work, we for the first time perform an empirical study to verify whether such trainable matching subnetworks also exist in style transfer models. Specifically, we take two most popular style transfer models, *i.e.*, AdaIN and SANet, as the main testbeds, which represent global and local transformation based style transfer methods respectively. We carry out extensive experiments and comprehensive analysis, and draw the following conclusions. (*i*) Compared with fixing the VGG encoder, style transfer models can benefit more from training the whole network together. (*ii*) Using iterative magnitude pruning, we find the matching subnetworks at 89.2% sparsity in AdaIN and 73.7% sparsity in SANet, which demonstrates that *Style transfer models can play lottery tickets too.* (*iii*) The feature transformation module should also be pruned to obtain a much sparser model without affecting the existence and quality of the matching subnetworks. (*iv*) Besides AdaIN and SANet, other models such as LST, MANet, AdaAttN and MCCNet can also play lottery tickets, which shows that LTH can be generalized to various style transfer models.

CCS CONCEPTS

• **Computing methodologies** → **Computational photography**; **Non-photorealistic rendering**.

*Corresponding author

KEYWORDS

style transfer, neural network pruning, lottery ticket hypothesis

ACM Reference Format:
Meihao Kong, Jing Huo, Wenbin Li, Jing Wu, Yu-Kun Lai, and Yang Gao. 2022. Playing Lottery Tickets in Style Transfer Models. In *Proceedings of the 1st International Workshop on Methodologies for Multimedia (M4MM '22), October 14, 2022, Lisboa, Portugal.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3552487.3556440

1 INTRODUCTION

Recent years have witnessed rapid development in the area of neural style transfer, which aims at composing a content image with new styles from reference images. Extensive research has focused on improving visual quality [2, 3, 28, 30, 33, 39], efficiency [25, 26, 29, 41, 47, 49] and flexibility [1, 17, 21, 22, 24, 43]. Although great success has been achieved in these aspects, the memory and computational footprints required for these style transfer models are large owing to the widely adopted autoencoder-based architecture. In particular, they usually consist of a pre-trained VGG encoder [46], a feature transformation module and a corresponding decoder, with a large number of parameters, which makes these models infeasible to be used in resource-constrained scenarios. This naturally raises a question: *"Can we prune a large style transfer model while preserving its performance?"*

In this paper, we aim to answer the above question via the lens of lottery ticket hypothesis (LTH) [13]. LTH states that there exist small subnetworks in dense neural networks that can be trained in isolation from initialization to match the performance of the original network after training for at most the same number of iterations. Existence of LTH has been successfully shown in various fields [5, 16, 42, 50], and its property has been widely studied [6, 14, 27, 37]. Nonetheless, to the best of our knowledge, no prior work exists on understanding the lottery ticket hypothesis in style transfer, which otherwise could be a powerful tool to understand the parameter redundancy in the current prevailing style transfer models. This will be the focus of our work.

Specifically, we mainly investigate the existence of LTH in two representative style transfer models – AdaIN (a representative *global transformation based* method) [23] and SANet (a representative *local transformation based* method) [39]. AdaIN applies mean and standard deviation to transform the features globally, and is the basis for many global style transfer methods, while SANet is the pioneering work that introduces attention mechanism to consider local feature matching.

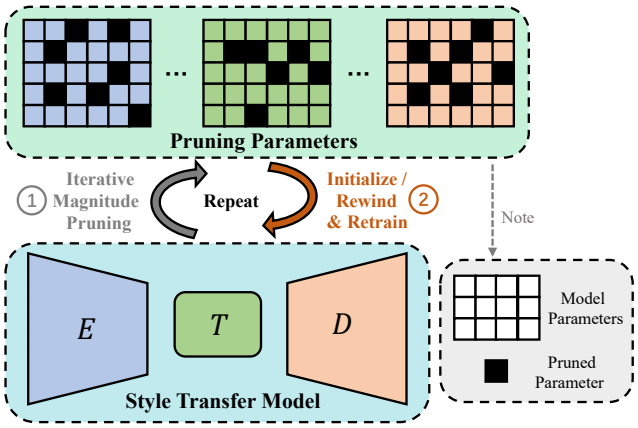

Figure 1: Overview of the overall process for playing lottery tickets in style transfer models. Winning tickets can be found by IMP. After that, we initialize or rewind the parameters of the original full model and then retrain the subnetwork to verify the performance of the found ticket.

In our context, a *ticket* means a style transfer subnetwork, a *winning ticket* represents a subnetwork which can match the performance of the original full style transfer model.

In order to enable searching for winning tickets across the whole style transfer networks, we first conduct a comparative experiment to verify whether training the VGG encoder together could be comparable or even outperform the traditional training strategy, *i.e.,* only training the decoder and feature transformation module. This has rarely been explored in previous studies. We evaluate the performance both quantitatively and qualitatively. Table 1 and Figure 2 illustrate the superiority of the strategy of training together. This finding not only benefits to obtain much sparser matching subnetworks (winning tickets), but also establishes a stronger baseline of full models. With the above discovery, we are able to further explore *"Are there winning tickets in the whole style transfer models?"*

The process of finding and verifying winning tickets uses a technique called *Iterative Magnitude Pruning* (IMP), which prunes the model by alternating between network pruning and network re-training. At each iteration of this process, we obtain a sparse subnetwork (a ticket) along with its parameter initialization or rewinding. Figure 1 gives an overview of the overall process. We fully combine the provided *average style transfer test error*, visual results as well as a user study to evaluate the performance of subnetworks qualitatively and quantitatively. Furthermore, extensive experimental results demonstrate that *"Style transfer models can play lottery tickets too."*

In order to gain more insight into LTH of style transfer models, we further conduct widely verified and comparative experiments, including 1) the selection of the pruning strategy and initialization policy, 2) whether to prune feature transformation module, 3) the effect of rewinding late, 4) performance comparison with other pruning methods, and 5) LTH in other style transfer models. Through comprehensive analysis, our main findings can be summarized as follows:

- *Training together gains a lot*: Different from the traditional training strategy *i.e.,* fixing the VGG encoder, we train the encoder together to search for winning tickets in the whole network and obtain much sparser matching subnetworks. We also find that the performance of the original models can be further improved by adopting this training strategy.
- *Style transfer models can play lottery tickets too*: Using iterative magnitude pruning, we are able to identify the matching subnetworks at 89.2% sparsity in AdaIN [23] and 73.7% sparsity in SANet [39]. Moreover, these extreme winning tickets can achieve or even exceed the performance of the original full models.
- *The feature transformation module should also be pruned*: We experimentally find that, not only the autoencoder, the feature transformation module can also be pruned to get a sparser matching subnetwork without affecting the existence and quality of winning tickets.
- *Rewinding has minor impact*: Unlike [14, 42], we find that "late rewinding" technique does not have a notable effect on style transfer subnetworks.
- *Universal presence of LTH in style transfer models*: Besides AdaIN and SANet, we also verify LTH in other multiple style transfer models, including LST [30], MANet [11], AdaAttN [33], and MCCNet [10] and obtain winning tickets with extreme sparsity of 93.1%, 79%, 73.7%, 83.2%, respectively. This indicates the great potential of LTH in style transfer model compression.

2 RELATED WORK

2.1 Neural Style Transfer

We have witnessed a boom of neural style transfer methods in the past few years. Numerous research works have been conducted to improve the visual quality [8, 23, 28, 30, 31, 33, 39, 45], efficiency [25, 26, 29, 41, 47] and flexibility [17, 22, 24, 43] of style transfer models. Nevertheless, most of these approaches have the common problem of large model sizes due to the widespread adoption of the over-parameterized VGG-based backbone in conjunction with its corresponding feature decoder.

Different from those efforts on improving style transfer model capability regardless of model computational complexity, we focus on making style transfer models sparser and smaller. Note that recently, Wang *et al.* [49] have also attempted to train smaller style transfer models based on WCT [31] and AdaIN [23]. However, our focus is different from theirs. Specifically, Wang *et al.* [49] aim to handle the ultra-resolution style transfer task via knowledge distillation. In addition, they have only compressed the encoder without compressing the decoder, hence the overall network is still large. Here, we study the over-parameterization of the whole style transfer networks from the perspective of lottery ticket hypothesis, a popular concept in deep neural network nowadays which has not been introduced into the field of style transfer yet.

2.2 Lottery Ticket Hypothesis

Dating back to 2018, Frankle *et al.* [13] firstly proposed to find winning tickets via Iterative Magnitude Pruning (IMP). The lottery ticket hypothesis (LTH) has attracted widespread attention and has

been evidenced in various traditional computer vision fields, such as image classification [35, 36, 44, 48], and object detection [18]. Recently, the properties of LTH has also been widely studied across other fields, such as natural language processing [5, 15, 40, 50], reinforcement learning [50], graph neural networks [6], life-long learning [7], and generative adversarial networks [4, 9, 27]. Besides, the "rewinding late" rule is found by [14, 42] to scale up LTH to larger networks and datasets.

Although LTH has made pioneering progress in various deep learning fields, to our best knowledge, the research of lottery tickets hypothesis in the style transfer field remains untouched. At present, style transfer has played an important role in image and video processing areas, *e.g.,* movie synthesis and photo art. Therefore, it is critical to understand the parameter redundancy in such models and make them small without sacrificing the performance.

3 PRELIMINARIES

In this section, we describe the techniques that we use to find winning tickets and the metrics to evaluate the performance of the subnetworks. We also present our setup for the empirical study.

3.1 Original Full Networks

We use two representative style transfer networks as the main testbeds: AdaIN [23] and SANet [39].

AdaIN is one of the most popular global transformation based style transfer approaches. The core idea is to adaptively transfer the mean and standard deviation from the style feature map to the content feature map. Specifically, given a content image I_c and a style image I_s, AdaIN first adopts the first few layers (up to $relu4\_1$) of a pre-trained VGG-19 network [46] to encode content features F_c and style features F_s:

$$F_c, F_s = E(I_c, I_s; \theta_E) \tag{1}$$

where E is the encoder with parameters θ_E. Next, the AdaIN module replaces the channel-wise mean and standard deviation from one feature map to the other:

$$AdaIN(F_c, F_s) = \sigma(F_s)\left(\frac{F_c - \mu(F_c)}{\sigma(F_c)}\right) + \mu(F_s) \tag{2}$$

where $\mu(F_c)$ ($\mu(F_s)$) calculates the mean of F_c (F_s) and $\sigma(F_c)$ ($\sigma(F_s)$) calculates the standard deviation of F_c (F_s). For simplicity, we assume $F_{cs} = AdaIN(F_c, F_s)$. Then, target features F_{cs} are fed into the decoder D to obtain the stylized image I_t:

$$I_t = D(F_{cs}; \theta_D) \tag{3}$$

where θ_D denotes the parameters in D.

SANet is a local transformation based method. It is able to flexibly match the local semantically nearest style features onto the content features via a learnable style-attentional transformation module. Similar to AdaIN, SANet also utilizes the pre-trained VGG-based autoencoder architecture. The whole process of SANet can be divided into three stages. Firstly, the feature encoding stage:

$$F_c^{41}, F_s^{41}, F_c^{51}, F_s^{51} = E(I_c, I_s; \theta_E) \tag{4}$$

where F_c^{41} (F_s^{41}) and F_c^{51} (F_s^{51}) denote the corresponding layer VGG feature maps (*i.e.,* $relu4\_1$ and $relu5\_1$) of content (style) images respectively. Secondly, the feature transformation stage:

$$SANet(F_c, F_s) = T(F_c^{41}, F_s^{41}, F_c^{51}, F_s^{51}; \theta_T) \tag{5}$$

where T is the attention based feature transformation module with trainable parameters θ_T (different from the parameter-free transformation in AdaIN). Finally, the stylized output image I_t is synthesized by feeding F_{cs} into the decoder just like Eq. (3).

Actually, almost all the mainstream feed-forward style transfer methods [8, 10, 11, 23, 30, 33, 39, 45] have similar architectures as AdaIN [8, 23, 45] or SANet [10, 11, 30, 33, 39]. It indicates that if LTH exists in AdaIN and SANet, it will also exist in these methods. Our experiments in Section 4.7 demonstrate this point.

3.2 Subnetworks

For a network f that maps samples $x \in \mathcal{X}$ with parameters $\boldsymbol{\theta} \in \mathbb{R}^d$ to $f(x; \boldsymbol{\theta})$, a subnetwork is defined as $f(x; \boldsymbol{m} \odot \boldsymbol{\theta})$, where $\boldsymbol{m} \in \{0, 1\}^d$ is a binary pruning mask indicating which part of the network parameters is set to 0, with $\odot$ denoting element-wise multiplication. For any configuration $\boldsymbol{m}$, the effective parameter space of the induced network $f(x; \boldsymbol{m} \odot \boldsymbol{\theta})$ is $\{\boldsymbol{m} \odot \boldsymbol{\theta} \mid \boldsymbol{\theta} \in \mathbb{R}^d\}$, a $\|\boldsymbol{m}\|_0$-dimensional space, hence we say that the subnetwork has $\|\boldsymbol{m}\|_0$ parameters instead of d. Specifically, for both AdaIN and SANet, two separate masks, $\boldsymbol{m_E}$ and $\boldsymbol{m_D}$, are required for the VGG based encoder and decoder. Moreover, a transformation module mask $\boldsymbol{m_T}$ is needed for SANet. Accordingly, a general subnetwork of style transfer methods consists of: a sparse encoder $E(\cdot; \boldsymbol{m_E} \odot \boldsymbol{\theta_E})$, a sparse transformation module $T(\cdot; \boldsymbol{m_T} \odot \boldsymbol{\theta_T})$ and a sparse decoder $D(\cdot; \boldsymbol{m_D} \odot \boldsymbol{\theta_D})$.

Algorithm 1 Iterative Magnitude Pruning for Style Transfer Winning Tickets

Input: Total training iteration N; Rewind iteration $r \geq 0$; Desired sparsity s
Output: A sparse style transfer model $E(\cdot; \boldsymbol{m_E} \odot \boldsymbol{\theta_E})$, $D(\cdot; \boldsymbol{m_D} \odot \boldsymbol{\theta_D})$ and $T(\cdot; \boldsymbol{m_T} \odot \boldsymbol{\theta_T})$

Set $\boldsymbol{\theta_E}^{(r)}$, $\boldsymbol{\theta_D}^{(r)}$ and $\boldsymbol{\theta_T}^{(r)}$ as initial weights of
  $E(\cdot)$, $D(\cdot)$ and $T(\cdot)$ respectively.
Set $\boldsymbol{m_E} = \mathbf{1} \in \mathbb{R}^{\|\boldsymbol{\theta_E}^{(r)}\|_0}$, $\boldsymbol{m_D} = \mathbf{1} \in \mathbb{R}^{\|\boldsymbol{\theta_D}^{(r)}\|_0}$,
  and $\boldsymbol{m_T} = \mathbf{1} \in \mathbb{R}^{\|\boldsymbol{\theta_T}^{(r)}\|_0}$, assume $\boldsymbol{m} = \{\boldsymbol{m_E}, \boldsymbol{m_D}, \boldsymbol{m_T}\}$.
while the sparsity of $\boldsymbol{m} < \boldsymbol{s}$ **do**
  Train $E(\cdot; \boldsymbol{m_E} \odot \boldsymbol{\theta_E}^{(r)})$ and $D(\cdot; \boldsymbol{m_D} \odot \boldsymbol{\theta_D}^{(r)})$ for N iterations to get parameters $\boldsymbol{\theta_E}^N$ and $\boldsymbol{\theta_D}^N$.
  if pruning the transformation module $T(\cdot)$ **then**
    Prune 20% of the parameters in $\boldsymbol{\theta_E}^N$, $\boldsymbol{\theta_D}^N$ and $\boldsymbol{\theta_T}^N$, calculating three mask $\boldsymbol{m_E}'$, $\boldsymbol{m_D}'$ and $\boldsymbol{m_T}'$.
  else
    Prune 20% of the parameters in $\boldsymbol{\theta_E}^N$ and $\boldsymbol{\theta_D}^N$, calculating two mask $\boldsymbol{m_E}'$ and $\boldsymbol{m_D}'$. $\boldsymbol{m_T}'$ remains $\mathbf{1} \in \mathbb{R}^{\|\boldsymbol{\theta_T}^{(r)}\|_0}$.
  end if
  Update $\boldsymbol{m_E} = \boldsymbol{m_E}'$, $\boldsymbol{m_D} = \boldsymbol{m_D}'$ and $\boldsymbol{m_T} = \boldsymbol{m_T}'$.
end while

3.3 Matching Subnetworks

For a network f and randomly-initialized parameters $\boldsymbol{\theta^{(0)}}$, a matching subnetwork f^* is given by a configuration $\boldsymbol{m} \in \{0, 1\}^d$, which

is trained in isolation from $\boldsymbol{\theta}^{*(0)} = \boldsymbol{\theta}^{(k)} \odot \boldsymbol{m}$, where $\boldsymbol{\theta}^{(k)}$ is the collection of parameter values obtained by training f from $\boldsymbol{\theta}^{(0)}$ for k iterations. Furthermore, to be a matching subnetwork, f^* should reach or even surpass the performance of a trained f given the same training iterations.

Winning Ticket. A matching subnetwork f^* is a *winning ticket* if it can be trained in isolation from initialization. In other words, a winning ticket is a matching subnetwork such that $k = 0$ in the definition above. Ticket search is to identify such a subnetwork, given an unpruned dense network f and randomly initialized parameters $\boldsymbol{\theta}^{(0)}$.

3.4 Identifying Subnetworks

Identifying subnetworks is to find three masks $\boldsymbol{m}_E$, $\boldsymbol{m}_D$ and $\boldsymbol{m}_T$ for the encoder, decoder and transformation module, respectively. Note that $\boldsymbol{m}_T$ is not needed for AdaIN. To achieve this, we utilize the Iterative Magnitude Pruning (IMP) algorithm [19]. In particular, we determine the pruning mask $\boldsymbol{m} = (\boldsymbol{m}_E, \boldsymbol{m}_D, \boldsymbol{m}_T)$ by training the full unpruned style transfer network. Then, we prune individual weights with the lowest-magnitudes throughout the network globally. In detail, the position of a remaining weight in $\boldsymbol{m}$ is marked as 1, and the position of a pruned weight is marked as 0. We set the weights of this subnetwork to $\boldsymbol{\theta}^{(i)}$ for a specific *rewinding* step i during training. For instance, to set the weights of the subnetwork to their values from the initialization, we set $\boldsymbol{\theta} = \boldsymbol{\theta}^{(0)}$. As previous work has shown, to find the smallest possible matching subnetworks, it is better to repeat this pruning process iteratively. Intuitively, we prune a certain amount (*e.g.,* 20%) of non-zero parameters each step and retrain the network several times to reach the desired sparsity rather than pruning the network only once to meet the sparsity requirement. Algorithm 1 presents details of the IMP procedure to find matching subnetworks. In addition, Figure 1 also provides a flowchart to illustrate the process.

3.5 Evaluation of Subnetworks

To evaluate whether the subnetwork is a matching subnetwork or not, after obtaining the subnetwork $E(\cdot; \boldsymbol{m}_E \odot \boldsymbol{\theta}_E)$, $D(\cdot; \boldsymbol{m}_D \odot \boldsymbol{\theta}_D)$ and $T(\cdot; \boldsymbol{m}_T \odot \boldsymbol{\theta}_T)$, we reset the weights to $\boldsymbol{\theta}_E^{(r)}$, $\boldsymbol{\theta}_D^{(r)}$ and $\boldsymbol{\theta}_T^{(r)}$ ($r > 0$ if rewinding strategy is used, where r is the rewind iteration). We then re-train the subnetwork, and evaluate whether it can still achieve the performance as the original full network. Note that, all the test content images, as well as all the test style images have *never* been seen during the training process.

Quantitative evaluations. We compare the subnetworks and the original full network via the *average style transfer test error* $\mathcal{E}$ calculated from numerous stylized results based on test images.

$$\mathcal{E} = \mathcal{E}_{content} + \mathcal{E}_{style} \tag{6}$$

For both AdaIN and SANet models, the average style error $\mathcal{E}_{style}$ is calculated as:

$$\mathcal{E}_{style} = \frac{1}{N}\sum_{n=1}^{N}\Big(\sum_{i=1}^{L}\|\mu(\Phi_i(I_t)) - \mu(\Phi_i(I_s))\|_2 \quad + \sum_{i=1}^{L}\|\sigma(\Phi_i(I_t)) - \sigma(\Phi_i(I_s))\|_2\Big) \tag{7}$$

where N denotes the number of test stylized images and each Φ_i denotes the i-layer in VGG-19. The average content error $\mathcal{E}_c$ of AdaIN is calculated as:

$$\mathcal{E}_{content}^{AdaIN} = \frac{1}{N}\sum_{n=1}^{N}\|F_t^{41} - F_c^{41'}\|_2 \tag{8}$$

where F_t^{41} denotes the *relu4_1* layer feature maps of the content image and $F_c^{41'}$ denotes the content features after AdaIN transformation. Similarly, for SANet:

$$\begin{aligned}\mathcal{E}_{content}^{SANet} = \frac{1}{N}\sum_{n=1}^{N}\big(&\|F_t^{41} - F_c^{41'}\|_2 + \|F_t^{51} - F_c^{51'}\|_2 \\ &+\|I_{cc} - I_c\|_2 + \|I_{ss} - I_s\|_2 + \sum_{i=1}^{L}\big(\|\Phi_i(I_{cc}) \\ &-\Phi_i(I_c)\|_2 + \|\Phi_i(I_{ss}) - \Phi_i(I_s)\|_2\big)\big)\end{aligned} \tag{9}$$

where I_{cc} (or I_{ss}) denotes the generated results using a common natural image (or painting) as content and style images simultaneously.

Overall, the average style transfer test error is consistent with the optimization loss function for training AdaIN and SANet networks (more details can be found in [23, 39]). Note that in the absence of ideal evaluation metrics for assessing the style transfer models' performance, the average style transfer test error is a feasible alternative to accomplish the measurement task quantitatively. For example, combining the results of Table 1 and Figure 2 (or Figure **??** and Figure 8), we can see that there is a strong correlation between visual quality and the test error, *i.e.,* better performance obtained in terms of content preservation and stylization degree where there is a smaller average style transfer test error.

Qualitative evaluations. We also conduct a user study to qualitatively evaluate the subnetworks. Specifically, 15 content images and 20 style images are randomly picked to form 300 images pairs in total. Then we randomly sample 100 content-style pairs and synthesize stylized images using both the original full networks and the corresponding subnetworks. Results are presented side-by-side in a random order and we ask subjects to select the most visually pleasant one from three views: content preservation, stylization degree, and overall preference. We collect 2000 votes from 20 users and present the statistical results in Figure 5.

Datasets and settings. In the training phase, we use MS-COCO dataset [32] and WikiArt dataset [38] as our content image set and style image set, respectively. Each dataset contains roughly 80,000 training examples. Besides, we follow the same settings (*e.g.,* hyperparameters, image resolution, *etc.*) as described in [23, 39] to train the original full model or its corresponding subnetworks. In the testing phase, 40 content images and 100 style images are randomly selected from the test set of the MS-COCO [32] and WikiArt dataset [38] to calculate the average style transfer test error. All models are trained on a GeForce RTX 2080 Ti GPU.

4 EXPERIMENTAL RESULTS AND ANALYSIS

4.1 Training Together Gains A Lot

Before starting the experimental verification of the LTH, we perform some preparatory experiments. As we all know, most of the

Table 1: Quantitative performance of different training strategies. AdaIN+: training encoder together based on AdaIN, SANet+: training encoder together based on SANet.

Methods	Avg content error	Avg style error	Avg error
AdaIN [23]	2.284	6.013	8.297
AdaIN+	**2.199**	**3.486**	**5.685**
SANet [39]	10.597	3.841	14.438
SANet+	**5.618**	**3.009**	**8.627**

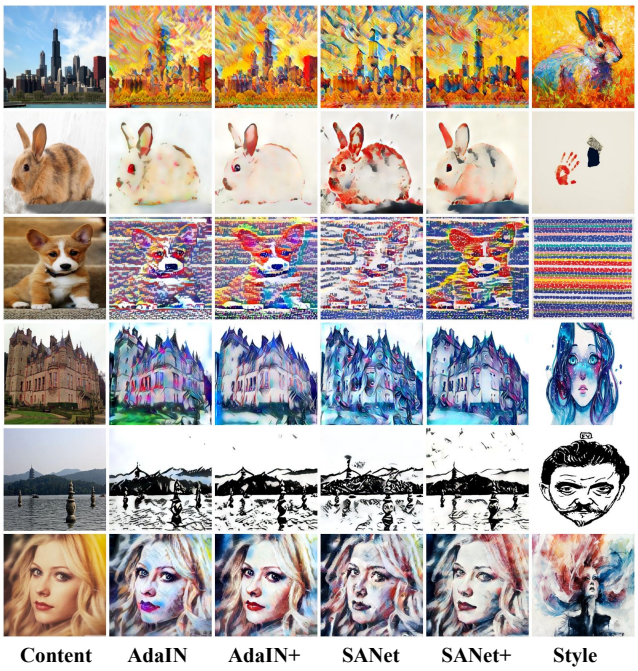

Figure 2: Image style transfer results of different model training strategies. Zoom in to have a better view.

existing style transfer methods adopt a common training strategy that fixes the VGG-19 encoder while only training the decoder and the feature transformation module. However, on the one hand, we are not able to search for winning tickets across the whole model while adopting the above training strategy, hence the VGG encoder is not considered during the training phase. On the other hand, it overly relies on the content and style patterns representation ability of the pre-trained VGG network which is not trained for this purpose and may not always be reliable. Therefore, we perform experiments to compare the above training strategy and training together strategy. For simplicity, we name the training together strategy as a *plus strategy*. The quantitative and qualitative results are shown in Table 1 and Figure 2, respectively.

From Table 1, we can see that the plus strategy outperforms the original strategy overall in terms of content, style and total errors. In Figure 2, by comparing stylized results of the 2nd (4th) and 3rd (5th) columns, we can further determine that the plus strategy is generally better than the traditional training strategy, combined with the aspect of content preservation and stylization degree. Therefore, we claim that *"Training Together Gains A Lot"*, which not only makes it possible to search for winning tickets across the whole network, but also establishes a stronger baseline of the original full model.

4.2 Style Transfer Models Can Play Lottery Tickets Too

To prove this point, we follow the same procedure in [13]. We first conduct experiments on AdaIN by pruning the VGG encoder and decoder with the following steps: 1) Run IMP to obtain the sparsity pattern $\{\boldsymbol{m_E}', \boldsymbol{m_D}'\}$, with $s_i\%$ sparsity; 2) Initialize the resulting subnetwork to ${\boldsymbol{\theta_E}}^{(0)}$ and ${\boldsymbol{\theta_D}}^{(0)}$. This produces a subnetwork $\{E(\cdot; \boldsymbol{m_E}' \odot {\boldsymbol{\theta_E}}^{(0)}), D(\cdot; \boldsymbol{m_D}' \odot {\boldsymbol{\theta_D}}^{(0)})\}$; 3) Train this subnetwork again to evaluate whether it is a winning ticket. In detail, we set $s_i\% = (1 - 0.8^i) \times 100\%$, which we use for all the experiments that involve iteratively pruning hereinafter. For SANet, besides $\boldsymbol{m_E}'$ and $\boldsymbol{m_D}'$, $\boldsymbol{m_T}'$ is also required to produce the subnetwork $\{E(\cdot; \boldsymbol{m_E}' \odot {\boldsymbol{\theta_E}}^{(0)}), \{T(\cdot; \boldsymbol{m_T}' \odot {\boldsymbol{\theta_T}}^{(0)}), D(\cdot; \boldsymbol{m_D}' \odot {\boldsymbol{\theta_D}}^{(0)})\}$. The overall process is similar to AdaIN. More specific details can be found in Algorithm 1.

Figure 3 shows the quantitative performance of the subnetworks generated by IMP with different sparsity levels. As can be seen, we are still able to find the winning tickets by iteratively pruning the entire networks to very high sparsity, around 90% in AdaIN+, and around 74% in SANet+, where the test errors of these subnetworks are comparable or even less than the full networks.

To give a more intuitive impression, we visualize the stylized results in Figures 4. For AdaIN+, we can see that the extremely sparse subnetwork performs very well even with only 10.8% parameters of the full network. Surprisingly, the found winning tickets in AdaIN+ can even yield more pleasant stylized results with fewer artifacts and better content preservation in detail, *e.g.,* less artifacts at the edges of the buildings (row 2), and in the sky (row 6), more complete bridge structure (row 5). We speculate that this phenomenon may be explained since a model with fewer parameters tends to be less overfitting. Meanwhile, from the stylized results generated by winning tickets found in SANet+, we can also observe that in the case of high sparsity, *i.e.,* 73.7%, the matching subnetwork can still achieve comparable performance to the original full model.

Moreover, we conduct a user study to evaluate the quality of the winning tickets more objectively. The statistical results are presented in Figure 5. We can see that no matter AdaIN+ or SANet+, the found winning tickets can achieve the performance that is comparable or even outperforms the full models, *i.e.,* 38.4% *vs.* 25.7% for AdaIN+ and 35.5% *vs.* 32.4% for SANet+. Besides AdaIN and SANet, the existence of the LTH can also be confirmed in many other style transfer models as shown in Section 4.7. In summary, through the comparison and analysis above, we can conclude that *"Style Transfer Models Can Play Lottery Tickets Too"*.

4.3 Feature Transformation Module Should Also be Pruned

Actually, for a range of feature transformation based style transfer models with learnable parameters [10, 11, 30, 33, 39], the intermediate transformation modules also contain a large number of

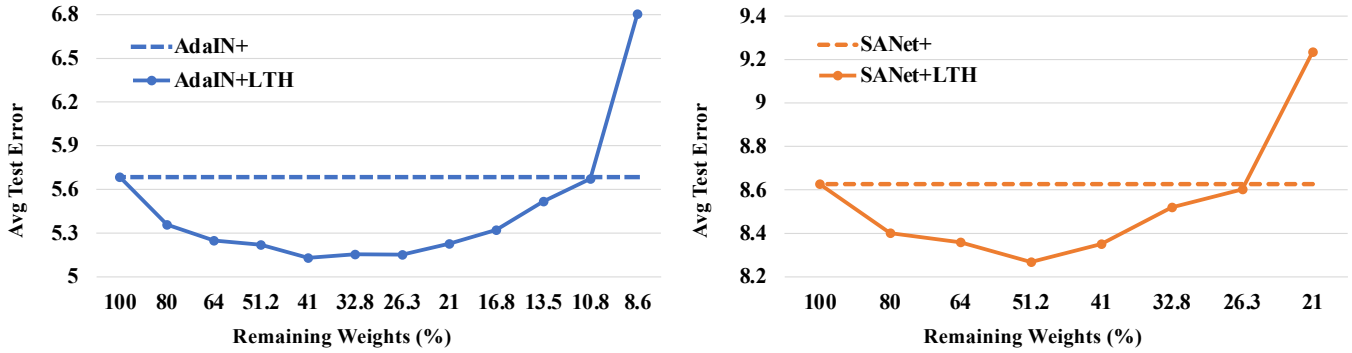

Figure 3: The average style transfer test error $\mathcal{E}$ curves of the subnetworks in AdaIN+ (left) and SANet+ (right) generated by IMP (average of three trials). The dashed line indicates the performance of full models, both trained via plus strategy as described in Section 4.1. Note that the x-axis charts the *percent of remaining weights*, where *remaining weights (%) = 1 - sparsity (%).*

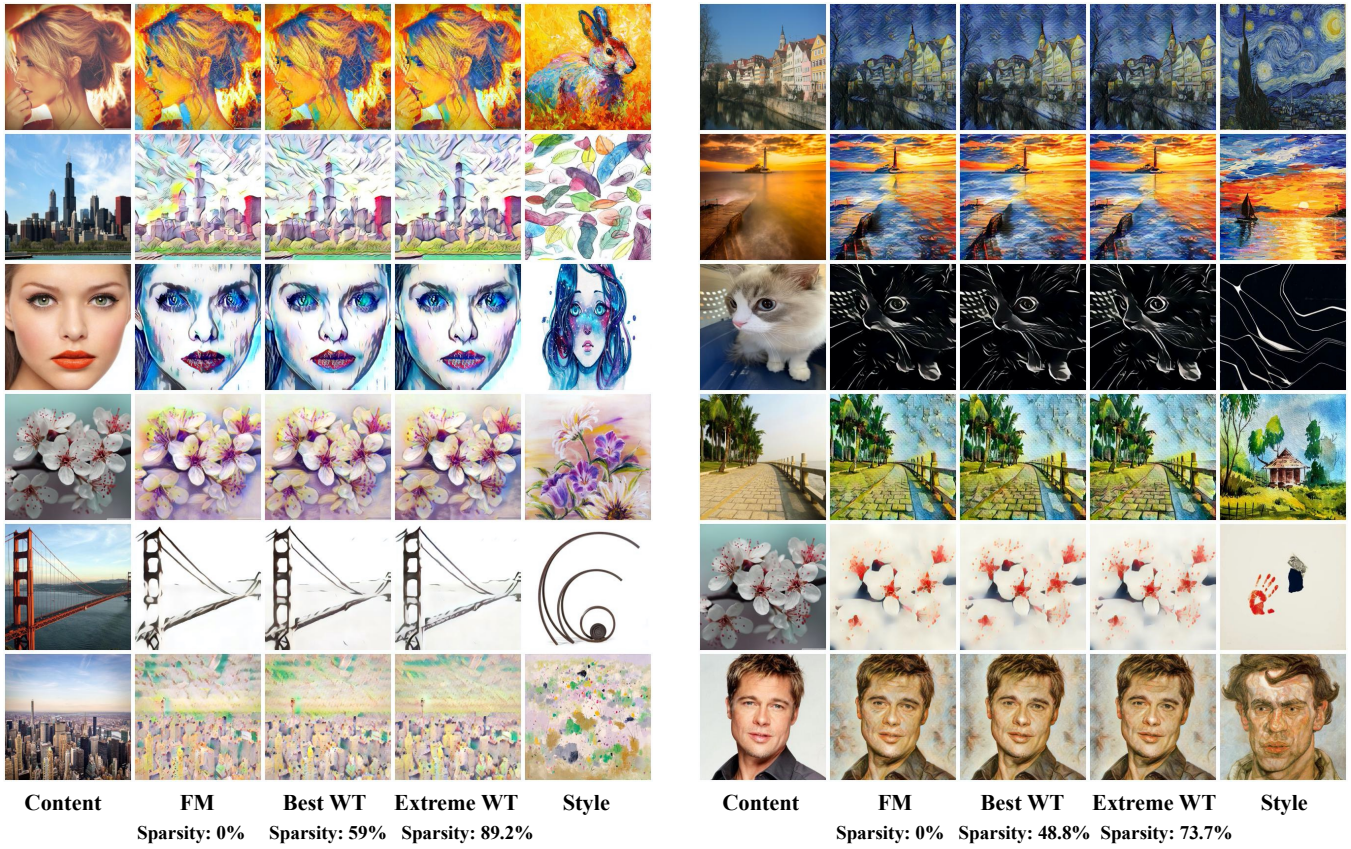

Figure 4: Image style transfer results of Winning Tickets found by IMP. Left: winning tickets of AdaIN+; Right: winning tickets of SANet+; FM: Full AdaIN+ (SANet+) model; Best WT: Winning Tickets with best performance, *i.e.,* lowest test error (sparsity of 59% for AdaIN+ and sparsity of 48.8% for SANet+); Extreme WT: Winning Tickets with highest sparsity (89.2% for AdaIN+ and 73.7% for SANet+). Zoom in to have a better view.

parameters, sometimes even more than the total parameters of the decoder. Taking SANet as an example, the parameter sizes of the encoder, transformation module and decoder are 49.38MB, 17.02MB and 13.37MB respectively, which means that even all parameters of autoencoder are pruned, only up to 78.7% sparsity can be achieved. Therefore, a natural question that came to our mind is: *Can we prune the feature transformation module to get a much sparser style transfer network without compromising performance?*

To answer this question, we compare two different iterative pruning settings for both SANet [39] and LST [30]: 1) Prune the

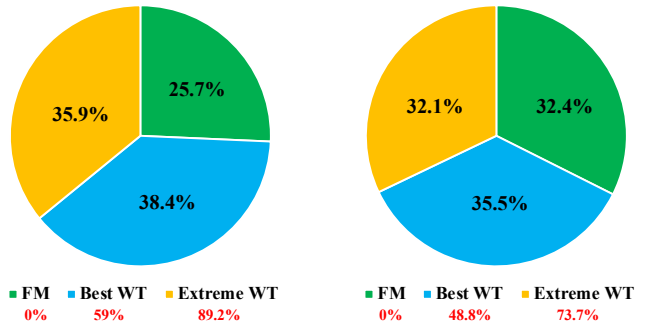

Figure 5: User study of Winning Tickets found by IMP. Left: winning tickets of AdaIN+; Right: winning tickets of SANet+. The percentage in red shown underneath each method indicates the sparsity of corresponding network.

autoencoder only (noPT); 2) Prune both the autoencoder and the transformation module (PT). All parts of the network are reset to the same random initialization $\boldsymbol{\theta}^{(0)}$ after the masks are obtained.

The experimental results are reported in Figure 6 and Figure 7, respectively. Both graphs suggest that the two settings share similar patterns when subnetworks are not too sparse (*i.e.,* 0%–50% of sparsity). However, as the total number of parameters of the model is further reduced to 30% or less, PT strategy shows enormous advantage compared to noPT strategy. The explanation is probably that adopting the noPT strategy, the parameters in the encoder and decoder will be severely reduced to get higher sparsity overall, hence damaging the quality of the subnetworks. While the PT strategy can achieve the overall trade-off of the whole model parameters and still perform well with extremely high sparsity, as the transformation module is also pruned.

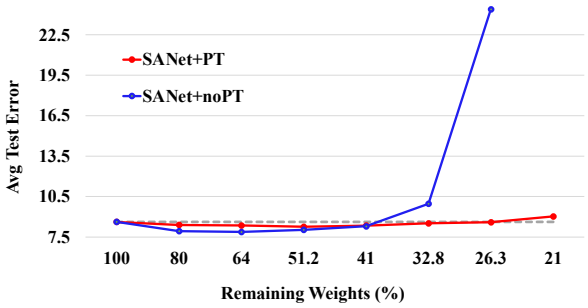

Figure 6: Results of whether to prune the feature transformation module of SANet. The dashed line indicates the performance of the full SANet+ model.

4.4 IMP Winning Tickets Are Sparser than OMP, Random Pruning, and Random Tickets

Previous work describes winning tickets as a "combination of weights and connections capable of learning" [13], which means both the specific pruned weights and the specific initialization are necessary for a winning ticket to achieve this performance. To extend such a statement in the context of style transfer models, we compare IMP with several other benchmarks, one-shot

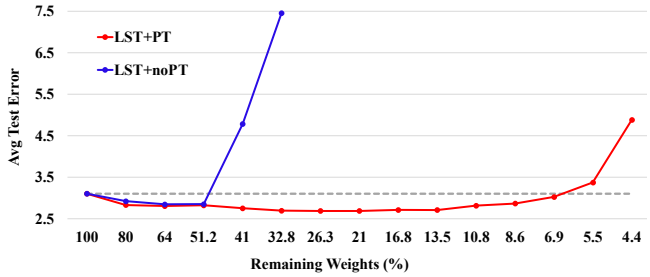

Figure 7: Results of whether to prune the feature transformation module of LST. The dashed line indicates the performance of the full LST+ model.

magnitude pruning (OMP), random pruning (RP), and random tickets (RT). Specifically, we train a subnetwork $\{E(\cdot; \boldsymbol{m}_E^{OMP} \odot \boldsymbol{\theta}_E{}^{(0)}), D(\cdot; \boldsymbol{m}_D^{OMP} \odot \boldsymbol{\theta}_D{}^{(0)})\}$ with a one-hot magnitude pruning mask $\boldsymbol{m}^{OMP}$ (which evaluates the importance of the iterative pruning strategy), a subnetwork $\{E(\cdot; \boldsymbol{m}_E^{RP} \odot \boldsymbol{\theta}_E{}^{(0)}), D(\cdot; \boldsymbol{m}_D^{RP} \odot \boldsymbol{\theta}_D{}^{(0)})\}$ with a random pruning mask $\boldsymbol{m}^{RP}$ (which evaluates the importance of the pruning masks) and a subnetwork $\{E(\cdot; \boldsymbol{m}_E^{IMP} \odot \boldsymbol{\theta}_E{}^{(0)'}), D(\cdot; \boldsymbol{m}_D^{IMP} \odot \boldsymbol{\theta}_D{}^{(0)'})\}$ with a random initialization $\boldsymbol{\theta}'$ (which evaluates the importance of the pre-trained initialization) based on AdaIN+ to see if IMP can obtain the best performance.

Table 2: Results of the best subnetworks and the extreme sparsity of matching networks found by different pruning settings. $\mathcal{E}_{Best}$: Minimal test error of all subnetworks. $\mathcal{S}_{Extreme}$: Extreme sparsity where matching subnetworks exist.

Methods	$\mathcal{E}_{Best}$(Sparsity)	$\mathcal{S}_{Extreme}$
Full Model	5.685(0.0%)	–
IMP	**5.134(59.0%)**	**89.2%**
OMP	5.255(59.0%)	59.0%
Random Pruning	5.413(20.0%)	30.0%
Random Tickets	–	0.0%

As shown in Table 2, the extreme winning ticket found by IMP is significantly sparser than that found by other pruning settings, and the minimal test error is smaller as well, which confirms the superiority of iterative pruning. Moreover, when adopting random tickets, we cannot obtain any matching subnetworks. This observation defends that specific initialization is essential for finding winning tickets. In Figure 8, we also show the visual performance of subnetworks with sparsity of 89.2% obtained by different pruning strategies. We observe that all other subnetworks have poor performance in style transfer with such a high sparsity, except for the IMP winning ticket.

4.5 Whether Rewinding Improves the Performance

In previous paragraphs, we demonstrate that we are able to find winning tickets in both AdaIN and SANet at non-trivial sparsities (*i.e.,* sparsities where random pruning cannot find winning tickets).

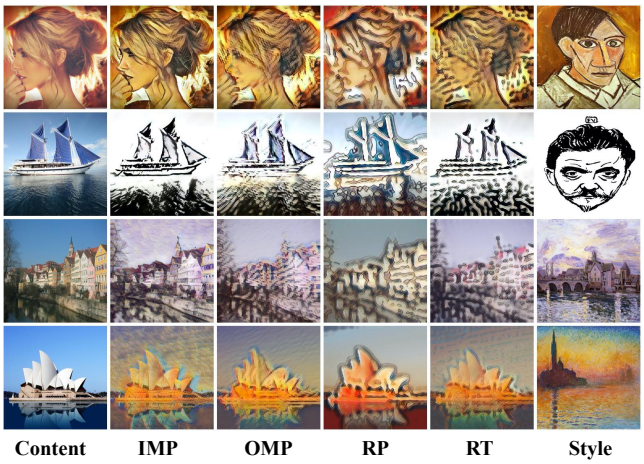

Figure 8: Image style transfer results of IMP, OMP, RP and RT with *sparsity of 89.2%*. Zoom in to have a better view.

Considering that the work [42] shows the rewinding paradigm is necessary to identify winning tickets for large networks, we would like to examine whether rewinding is helpful in the context of style transfer models. We perform experiments at different rewinding ratios. Specifically, after IMP training, we obtain the masks. Then, instead of resetting the weights to $\theta^{(0)}$, we rewind the weights to $\theta^{(i)}$, *i.e.*, the weights after i steps of training. The rewinding ratio = i/N, where N is the total training iteration. The results are shown in Table 3, we can see that rewinding does not have a notable effect. In particular, subnetworks trained at different rewinding ratios have the similar highest sparsity and best performance.

Table 3: Rewinding results of the best subnetworks and the extreme sparsity of matching networks found by IMP.

Rewinding ratios	AdaIN		SANet	
	$\mathcal{E}_{Best}$	$\mathcal{S}_{Extreme}$	$\mathcal{E}_{Best}$	$\mathcal{S}_{Extreme}$
Rewind 0%	5.134	89.2%	8.268	73.7%
Rewind 10%	5.114	89.2%	8.166	73.7%
Rewind 20%	5.097	89.2%	8.274	73.7%
Rewind 30%	5.052	89.2%	8.012	73.7%
Rewind 40%	5.083	89.2%	8.249	73.7%

4.6 IMP Winning Tickets vs. Other Pruning Methods

We further perform experiments to compare the IMP Winning Tickets with other mainstream pruning methods, *i.e.,* the structured channel pruning method [34] (network slimming) and finetuning based magnitude pruning method [20]. Results are shown in Figure 9. It can be seen that the IMP winning tickets are significantly better than subnetworks found by network slimming and finetuning based magnitude pruning at high sparsity.

4.7 Experiments on Other Style Transfer Models

To verify the universal presence of the lottery ticket hypothesis in diverse style transfer models, we also conduct experiments on

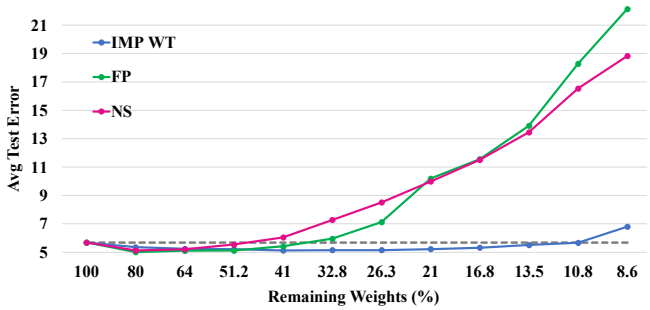

Figure 9: Results of different pruning methods. IMP WT: IMP Winning Tickets; FP: Finetuning based magnitude Pruning; NS: Network Slimming. The dashed line indicates the performance of the original AdaIN+ model.

LST [30], MANet [11], AdaAttN [33] and MCCNet [10]. Table 4 shows that LTH can be generalized to various style transfer models despite the different extreme sparsities. Besides, we are surprised to find that the smallest matching subnetwork of LST has only 6.9% of the parameters of the original full model.

Table 4: Results on other style transfer models. $\mathcal{E}_{Full}$: The test error of the full model. $\mathcal{E}_{Best}$: The minimal test error of all subnetworks. $\mathcal{S}_{Extreme}$: Extreme sparsity where matching subnetworks exist. (·%) denotes the sparsity of the corresponding network.

Model	$\mathcal{E}_{Full}$	$\mathcal{E}_{Best}$	$\mathcal{S}_{Extreme}$
LST [30]	3.103(0%)	2.686(79.0%)	**93.1%**
MANet [11]	17.176(0%)	11.227(20.0%)	**79.0%**
AdaAttN [33]	24.467(0%)	22.873(48.8%)	**73.7%**
MCCNet [10]	7.875(0%)	7.574(59.0%)	**83.2%**

5 CONCLUSION

In this paper, the LTH has been extended to the style transfer field for the first time. Through extensive experiments and comprehensive analysis, we verify the existence of winning tickets in a range of style transfer models. In future work, we plan to examine the speedup results on a hardware platform that is friendly to unstructured pruning. For instance, XNNPACK [12] has shown significant speedups over dense baselines on smartphone processors at 70%-90% unstructured sparsity. Besides, we will further explore the application of LTH in more diverse style transfer scenarios, *e.g.,* video synthesis, caricature generation, *etc.*

6 ACKNOWLEDGMENTS

This work is supported by Science and Technology Innovation 2030 New Generation Artificial Intelligence Major Project (2018AAA0100905, 2021ZD0113303), the National Natural Science Foundation of China (61806092, 62106100), the CAAI-Huawei MindSpore Open Fund and the Collaborative Innovation Center of Novel Software Technology and Industrialization.

REFERENCES

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

[2] Prashanth Chandran, Gaspard Zoss, Paulo Gotardo, Markus Gross, and Derek Bradley. 2021. Adaptive Convolutions for Structure-Aware Style Transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7972–7981.

[3] Haibo Chen, Lei Zhao, Zhizhong Wang, Huiming Zhang, Zhiwen Zuo, Ailin Li, Wei Xing, and Dongming Lu. 2021. Dualast: Dual style-learning networks for artistic style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 872–881.

[4] Tianlong Chen, Yu Cheng, Zhe Gan, Jingjing Liu, and Zhangyang Wang. 2021. Ultra-data-efficient gan training: Drawing a lottery ticket first, then training it toughly. *arXiv preprint arXiv:2103.00397* 3 (2021).

[5] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020. The lottery ticket hypothesis for pre-trained bert networks. *Advances in neural information processing systems* 33 (2020), 15834–15846.

[6] Tianlong Chen, Yongduo Sui, Xuxi Chen, Aston Zhang, and Zhangyang Wang. 2021. A unified lottery ticket hypothesis for graph neural networks. In *International Conference on Machine Learning*. PMLR, 1695–1706.

[7] Tianlong Chen, Zhenyu Zhang, Sijia Liu, Shiyu Chang, and Zhangyang Wang. 2020. Long live the lottery: The existence of winning tickets in lifelong learning. In *International Conference on Learning Representations*.

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

[9] Xuxi Chen, Zhenyu Zhang, Yongduo Sui, and Tianlong Chen. 2021. Gans can play lottery tickets too. *arXiv preprint arXiv:2106.00134* (2021).

[10] Yingying Deng, Fan Tang, Weiming Dong, Haibin Huang, Chongyang Ma, and Changsheng Xu. 2021. Arbitrary Video Style Transfer via Multi-Channel Correlation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 1210–1217.

[11] Yingying Deng, Fan Tang, Weiming Dong, Wen Sun, Feiyue Huang, and Changsheng Xu. 2020. Arbitrary style transfer via multi-adaptation network. In *Proceedings of the 28th ACM International Conference on Multimedia*. 2719–2727.

[12] Erich Elsen, Marat Dukhan, Trevor Gale, and Karen Simonyan. 2020. Fast sparse convnets. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 14629–14638.

[13] Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635* (2018).

[14] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. 2020. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learning*. PMLR, 3259–3269.

[15] Trevor Gale, Erich Elsen, and Sara Hooker. 2019. The state of sparsity in deep neural networks. *arXiv preprint arXiv:1902.09574* (2019).

[16] Zhe Gan, Yen-Chun Chen, Linjie Li, Tianlong Chen, Yu Cheng, Shuohang Wang, and Jingjing Liu. 2021. Playing lottery tickets with vision and language. *arXiv preprint arXiv:2104.11832* (2021).

[17] Leon A Gatys, Alexander S Ecker, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. 2017. Controlling perceptual factors in neural style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3985–3993.

[18] Sharath Girish, Shishira R Maiya, Kamal Gupta, Hao Chen, Larry S Davis, and Abhinav Shrivastava. 2021. The lottery ticket hypothesis for object recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 762–771.

[19] Song Han, Huizi Mao, and William J Dally. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149* (2015).

[20] Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems* 28 (2015).

[21] Kibeom Hong, Seogkyu Jeon, Huan Yang, Jianlong Fu, and Hyeran Byun. 2021. Domain-Aware Universal Style Transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 14609–14617.

[22] Zhiyuan Hu, Jia Jia, Bei Liu, Yaohua Bu, and Jianlong Fu. 2020. Aesthetic-aware image style transfer. In *Proceedings of the 28th ACM International Conference on Multimedia*. 3320–3329.

[23] Xun Huang and Serge Belongie. 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE international conference on computer vision*. 1501–1510.

[24] Jing Huo, Shiyin Jin, Wenbin Li, Jing Wu, Yu-Kun Lai, Yinghuan Shi, and Yang Gao. 2021. Manifold alignment for semantically aligned style transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 14861–14869.

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

[26] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 2016. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*. Springer, 694–711.

[27] Neha Mukund Kalibhat, Yogesh Balaji, and Soheil Feizi. 2020. Winning lottery tickets in deep generative models. *arXiv preprint arXiv:2010.02350* (2020).

[28] Nicholas Kolkin, Jason Salavon, and Gregory Shakhnarovich. 2019. Style transfer by relaxed optimal transport and self-similarity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10051–10060.

[29] Chuan Li and Michael Wand. 2016. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *European conference on computer vision*. Springer, 702–716.

[30] Xueting Li, Sifei Liu, Jan Kautz, and Ming-Hsuan Yang. 2019. Learning linear transformations for fast image and video style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3809–3817.

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

[32] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*. Springer, 740–755.

[33] Songhua Liu, Tianwei Lin, Dongliang He, Fu Li, Meiling Wang, Xin Li, Zhengxing Sun, Qian Li, and Errui Ding. 2021. Adaattn: Revisit attention mechanism in arbitrary neural style transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 6649–6658.

[34] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. 2017. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*. 2736–2744.

[35] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. 2018. Rethinking the value of network pruning. *arXiv preprint arXiv:1810.05270* (2018).

[36] Haoyu Ma, Tianlong Chen, Ting-Kuei Hu, Chenyu You, Xiaohui Xie, and Zhangyang Wang. 2021. Good students play big lottery better. *arXiv preprint arXiv:2101.03255* 3 (2021).

[37] Eran Malach, Gilad Yehudai, Shai Shalev-Schwartz, and Ohad Shamir. 2020. Proving the lottery ticket hypothesis: Pruning is all you need. In *International Conference on Machine Learning*. PMLR, 6682–6691.

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

[39] Dae Young Park and Kwang Hee Lee. 2019. Arbitrary style transfer with style-attentional networks. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 5880–5888.

[40] Sai Prasanna, Anna Rogers, and Anna Rumshisky. 2020. When bert plays the lottery, all tickets are winning. *arXiv preprint arXiv:2005.00561* (2020).

[41] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[42] Alex Renda, Jonathan Frankle, and Michael Carbin. 2020. Comparing rewinding and fine-tuning in neural network pruning. *arXiv preprint arXiv:2003.02389* (2020).

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

[44] Pedro Savarese, Hugo Silva, and Michael Maire. 2020. Winning the lottery with continuous sparsification. *Advances in Neural Information Processing Systems* 33 (2020), 11380–11390.

[45] Lu Sheng, Ziyi Lin, Jing Shao, and Xiaogang Wang. 2018. Avatar-net: Multi-scale zero-shot style transfer by feature decoration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 8242–8250.

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

[47] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor S Lempitsky. 2016. Texture networks: Feed-forward synthesis of textures and stylized images.. In *ICML*, Vol. 1. 4.

[48] Chaoqi Wang, Guodong Zhang, and Roger Grosse. 2020. Picking winning tickets before training by preserving gradient flow. *arXiv preprint arXiv:2002.07376* (2020).

[49] Huan Wang, Yijun Li, Yuehai Wang, Haoji Hu, and Ming-Hsuan Yang. 2020. Collaborative distillation for ultra-resolution universal style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1860–1869.

[50] Haonan Yu, Sergey Edunov, Yuandong Tian, and Ari S Morcos. 2019. Playing the lottery with rewards and multiple languages: lottery tickets in rl and nlp. *arXiv preprint arXiv:1906.02768* (2019).