Teachers Do More Than Teach: Compressing Image-to-Image Models

Qing Jin[1*] Jian Ren[2] Oliver J. Woodford[*] Jiazhuo Wang[2]
Geng Yuan[1] Yanzhi Wang[1] Sergey Tulyakov[2]
[1]Northeastern University, USA [2]Snap Inc.

Abstract

Generative Adversarial Networks (GANs) have achieved huge success in generating high-fidelity images, however, they suffer from low efficiency due to tremendous computational cost and bulky memory usage. Recent efforts on compression GANs show noticeable progress in obtaining smaller generators by sacrificing image quality or involving a time-consuming searching process. In this work, we aim to address these issues by introducing a teacher network that provides a search space in which efficient network architectures can be found, in addition to performing knowledge distillation. First, we revisit the search space of generative models, introducing an inception-based residual block into generators. Second, to achieve target computation cost, we propose a one-step pruning algorithm that searches a student architecture from the teacher model and substantially reduces searching cost. It requires no ℓ^1 sparsity regularization and its associated hyper-parameters, simplifying the training procedure. Finally, we propose to distill knowledge through maximizing feature similarity between teacher and student via an index named Global Kernel Alignment (GKA). Our compressed networks achieve similar or even better image fidelity (FID, mIoU) than the original models with much-reduced computational cost, e.g., MACs. Code will be released at https://github.com/snap-research/CAT.

1. Introduction

Generative adversarial networks (GANs), which synthesize images by adversarial training [21], have witnessed tremendous progress in generating high-quality, high-resolution, and photo-realistic images and videos [4, 33, 66]. In conditional setting [54], the generation process is controlled via additional input signals, such as segmentation information [7, 57, 59, 69, 70], class labels [81], and sketches [29, 83]. These techniques have seen applications in commercial image editing tools. However, due to their massive computation complexity and bulky size, applying generative models at scale is less practical, especially on resource-constrained platforms, where low memory footprint, power consumption, and real-time execution are as, and often more, important than performance [36].

To accelerate inference and save storage space for huge models without sacrificing performance, previous works propose to compress models with techniques including weight pruning [24], channel slimming [43, 44], layer skipping [3, 71], patterned or block pruning [17, 35, 40, 42, 49, 50, 51, 52, 55, 56, 80, 82], and network quantization [12, 18, 30, 31, 32, 38, 73]. Specifically, these studies elaborate on compressing discriminative models for image classification, detection, or segmentation tasks. The problem of compressing generative models, on the other hand, is less investigated, despite that typical generators are bulky in memory usage and inefficient during inference. Up till now, only a handful of attempts exist [20, 36, 63, 68], all of which degenerate the quality of synthetic images compared to the original model (Fig. 1).

[*]Work done while at Snap Inc.

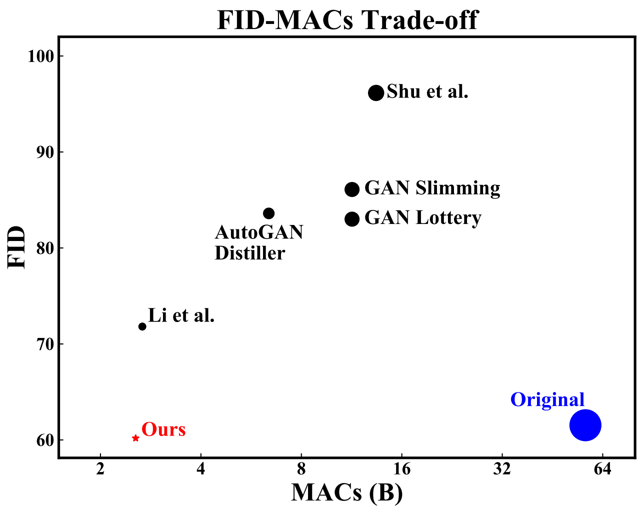

Figure 1: Performance comparison between our and existing GAN compression techniques [11, 20, 36, 63, 68] on CycleGAN [83] for Horse→Zebra dataset. *Smaller* MACs indicates more efficient models. *Lower* FID indicates models can generate more realistic images. Our method (red star) achieves the state-of-the-art performance-efficiency trade-off as it has the lowest FID with the smallest MACs.

In this work, we focus on compressing image-to-image translation networks, such as CycleGAN [83] and GauGAN [57]. Existing compression method [36] obtains an efficient student model and employs two additional networks: teacher and supernet, where the former is for knowledge distillation and the latter for architecture search. However, we argue that the supernet is not necessary, as the teacher can play its role. Specifically, in our proposed framework, the teacher does more than teaching the student (*i.e*. knowledge distillation)—it plays a central role in all aspects of the framework through three key contributions:

1. We introduce a new network design that can be applied to both encoder-decoder architectures such as Pix2pix [29], and decoder-style networks such as GauGAN [57]. It serves as both the teacher network design, *and* the architecture search space of the student.
2. We directly prune the trained *teacher* network using an efficient, one-step technique that removes certain channels in its generators to achieve a target computation budget, *e.g*., the number of Multiply-Accumulate Operations (MACs). This reduces architecture search costs by at least $10,000\times$ than the state-of-the-art compression method for generative models. Furthermore, our pruning method only involves one hyper-parameter, making its application straightforward.
3. We introduce a knowledge distillation technique based on the similarity between teacher and student models' feature spaces, which we call global kernel alignment (*GKA*). GKA directly forces feature representations from the two models to be similar, and avoids extra learnable layers [36] to match the different dimensions of teacher and student feature spaces, which could otherwise lead to information leakage.

We name our method as **CAT** as we show teacher model can and should do **C**ompression **A**nd **T**eaching (distillation) jointly, which we find is beneficial for finding generative networks with smaller MACs, using much lower computational resource than prior work. More importantly, our compressed networks can achieve similar or even better performance than their original counterparts (Tab. 1).

2. Related Work

Due to their high computation cost, running GANs on resource-constrained devices in real-time remains a challenging problem. As a result, GAN compression has garnered attention recently. Existing methods [1, 9, 20, 36, 63, 68] exploit network architecture search/pruning and knowledge distillation (discussed below). Although they can compress the original models (*e.g*., CycleGAN [83]) to a relatively small MACs, all these methods suffers from sacrifice on performance. In contrast, our method finds smaller networks than existing compressed GAN models, whilst *improves* performance over the original models, such as Pix2pix [29], CycleGAN [83], and GauGAN [57].

Network architecture search & pruning. To determine the structure of a pruned model, previous work employs neural architecture search (NAS) [6, 10, 37, 40, 39, 41, 42, 46, 47, 53, 60, 65, 72, 78, 84] and pruning techniques [3, 16, 17, 35, 42, 43, 44, 49, 50, 51, 52, 55, 56, 62, 71, 76, 77, 79, 80, 82], where the number of channels and/or operations can be optimized automatically. Applying these methods directly on generative models can lead to inferior performance of compressed models than their original counterparts. For example, Shu *et al*. [63] employ an evolutionary algorithm [58] and Fu *et al*. [20] engage differentiable network design [39], while Li *et al*. [36] train a supernet with random sampling technique [5, 23, 77, 78] to select the optimal architecture. The common key drawback of these methods is the slow searching process. In contrast, directly pruning on a pre-trained model is much faster. Following previous methods of network slimming [43, 44], Wang *et al*. [68] apply ℓ^1 regularization to generative models for channel pruning. However, they report performance degradation compared to the original network. Besides, these pruning methods require tuning additional hyper-parameters for ℓ^1 regularization to encourage channel-wise sparsity [43, 44] and even more hyper-parameters to decide the number of channels to be pruned [53], making the process tedious. Additionally, GAN training involves optimizing multiple objective functions, and the associated hyper-parameters make the training process even harder. Recently, lottery ticket hypothesis [19] is also investigated on GAN problem [11], while the performance is not satisfactory.

Knowledge distillation [26] is a technique to transfer knowledge from a larger, teacher network to a smaller, student network, and has been used for model compression in various computer vision tasks [8, 9, 48, 74, 45]. A recent survey [22] categorizes knowledge distillation as response-based, feature-based, or relation-based. Most GAN compression methods [1, 9, 20] use response-based distillation, enforcing the synthesized images from the teacher and student networks to be the same. Li *et al*. [36] apply feature-based distillation by introducing extra layers to match feature sizes between the teacher and student, and minimizing the differences of these embeddings using mean squared error (MSE) loss. However, this has the potential problem that some information can be stored in those extra layers, without being passed on to the student. Here, we propose to distill knowledge by directly maximizing the similarity between features from teacher and student models.

3. Methods

In this section, we show our method for searching a compressed student generator from a teacher generator. We revisit the network design of conditional image genera-

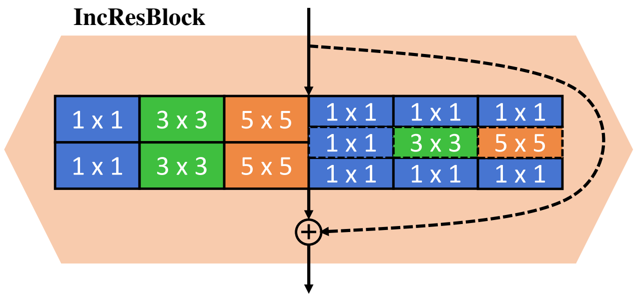

Figure 2: IncResBlock includes three conventional convolution blocks and three depth-wise convolution blocks (dashed border), both with kernels sizes of 1, 3, 5. Normalization layers (*e.g.*, BN), and ReLU, are applied between each two consecutive convolution layers. A normalization layer that can be inserted after summing features from the six blocks and the residual connection are optional. Unless otherwise stated, both are applied by default.

tion models and introduce inception-based residual blocks (Sec. 3.1). The teacher model is built upon the proposed block design and can serve two purposes. First, we show that the teacher model can be viewed as a large search space that enables *one-shot* neural architecture search without training an extra supernet. With the proposed one-step pruning method, a computationally efficient network that satisfies a given computational budget can be found instantly (Sec. 3.2). Second, we show the teacher model itself is sufficient for knowledge distillation, without necessity of introducing extra layers. By maximizing the similarity between intermediate features of teacher and student network directly, where features of the two networks contain different numbers of channels, we can effectively transfer knowledge from teacher to student (Sec. 3.3).

3.1. Design of Teacher Generator

Existing efforts leverage supernet to introduce search space that contains more efficient networks [5, 23, 36]. The optimization of supernet can lead to extra training costs. However, as we already have a teacher network in hand, searching efficient student from the teacher model should be more straightforward, as long as the teacher network contains a large searching space. In this way, the teacher network can perform both knowledge distillation and provide search space. Therefore, the goal of obtaining a good supernet can be changed to design a teacher generator that can synthesize high fidelity images; and itself contains a reasonable search space.

Inception-based residual block. With the above goal bearing in mind, we design a new architecture for the image generation tasks so that a pre-trained teacher generator with such architecture can serve as a large search space. We aim to search for a smaller student network that can have different operations (*e.g.*, convolution layers with various kernel size) and different numbers of channels than the teacher network through pruning. Towards this end, we adopt the widely used inception module on discriminative models [53, 64, 85] to the image generators and propose the inception-based residual block (IncResBlock). A conventional residual block in generators only contains convolution layers with one kernel size (*e.g.*, 3×3), while in IncResBlock, as shown in Fig. 2, we introduce convolution layers with different kernel sizes, including 1×1, 3×3, and 5×5. Additionally, we incorporate depth-wise blocks [27] into IncResBlock as depth-wise convolution layers typically require less computation cost without sacrificing the performance, and are particularly suitable for models deployed on mobile devices [61]. Specifically, the IncResBlock includes six types of operations, with two types of convolution layers and three different kernel-sizes. To achieve similar total computation cost, we set the number of output channels for the first convolution layers of each operations to that of the original residual blocks divided by six, which is the number of different operations in the IncResBlock. We find the performance is maintained thanks to the architecture design.

To get our teacher networks, for Pix2pix and CycleGAN, we replace all residual blocks in original models with the IncResBlock. For GauGAN, we apply IncResBlock in both the SPADE modules and the residual blocks. More details are illustrated in the supplementary materials.

3.2. Search from Teacher Generator via Pruning

With the teacher network introduced, we search a compressed student network from it. Our searching algorithm includes two parts. The first one is deciding a threshold based on the given computational budget, and the second one is pruning channels with a scale less than a threshold. Compared with existing iterative pruning methods [43, 53], we only perform pruning once, and we name our searching algorithm as *one-step pruning*.

Automatically threshold searching. Following existing efforts [43, 44], we prune the channels through the magnitudes of scaling factors in normalization layers, such as Batch Normalization (BN) [28] and Instance Normalization (IN) [67]. To this end, a threshold is required to choose channels to prune. As we train the teacher model without regularization, there is no constraint to force the teacher model to be sparse. The magnitude of scaling factors from the normalization layers is not guaranteed to be small. Thus, the previous iterative pruning methods, which remove channels using a manually designed threshold, are not suitable for our network.

To solve this, we determine the threshold by a given computation budget, which can be MACs or latency. All channels with scale smaller than the threshold are pruned until the final model achieves the target computation budget.

We find the scale threshold by binary search on the scaling factors of normalization layers from the pre-trained teacher model. Specifically, we temporarily prune all channels with a scaling factor magnitude smaller than the threshold and measure the computational cost of the pruned model. If it is smaller than the budget, the model is pruned too much and we search in the lower interval to get a smaller threshold; otherwise, we search in the upper interval to get a larger value. During this process, we also keep the number of output channels for convolution layers outside the IncResBlock larger than a pre-defined value to avoid an invalid model. Details of the algorithm are illustrated in Algorithm 1.

Channel pruning. With the threshold decided, we perform network searching via pruning. Given an IncResBlock, it is possible to change both the number of channels in each layer and modify the operation, such that, *e.g.*, one IncResBlock may only include layers with kernel sizes 1×1 and 3×3. Similar to Mei *et al.* [53], we prune channels of the normalization layers together with the corresponding convolution layers. Specifically, we prune the first normalization layers for each operation in IncResBlock, namely the ones after the first $k \times k$ convolution layers for conventional operations and the ones after the first 1×1 convolution layers for depth-wise operations.

Algorithm 1 Searching via One-Step Pruning.

```
Require: Computational budget $T_{\mathrm{b}}$, teacher model $\mathrm{G_T}$, scaling factors $\gamma_i^{(l)}$ (used for pruning) of the $i$-th channel in normalization layers $N^{(l)} \in \mathrm{G_T}$, minimum # output channels $c_{lb}$ for convolution layers (outside the IncResBlock).
Ensure: pruned student architecture $\mathrm{G_S}$.
1:  Initialize scale lower bound $\gamma_{lo}$: $\gamma_{lo} \leftarrow \min_{i,l} |\gamma_i^{(l)}|$.
2:  Initialize scale upper bound $\gamma_{hi}$: $\gamma_{hi} \leftarrow \max_{i,l} |\gamma_i^{(l)}|$.
3:  while $\gamma_{lo} < \gamma_{hi}$ do
4:      $\gamma_{th} \leftarrow (\gamma_{lo} + \gamma_{hi})/2$
5:      Prune channels satisfying $|\gamma_i^{(l)}| < \gamma_{th}$ on $\mathrm{G_T}$ while keep $c_{lb}$ to get $\mathrm{G_S}$
6:      $T \leftarrow$ computational cost of $\mathrm{G_S}$
7:      if $T > T_{\mathrm{b}}$ then
8:          $\gamma_{lo} \leftarrow \gamma_{th}$
9:      else
10:         $\gamma_{hi} \leftarrow \gamma_{th}$
11:     end if
12: end while
```

Discussion. Our searching algorithm is different from previous works that focus on compressing generative models in the following three perspectives. First, we search an efficient network from a *pre-trained* teacher model without utilizing an extra supernet [36]. Second, we show the scales of the normalization layers in the *pre-trained* teacher network are sufficient for pruning, therefore, weight regularization for iterative pruning [53, 68] might not be necessary for the generation tasks. Third, the teacher network can be compressed to several different architectures, and we can find the student network that satisfies an arbitrary type of computational cost, *e.g.*, MACs, under any value of predefined budget during the searching directly. Such differences bring us three advantages. First, searching cost is significantly reduced without introducing extra network. Second, removing the weight regularization, *e.g.*, ℓ^1-norm, eases the searching process as a bunch of hyper-parameters are reduced, which we find are hard to tune in practice. Third, we have more flexibility to choose a student network with required computational cost.

3.3. Distillation from Teacher Generator

After obtaining a student network architecture, we train it from scratch, leveraging the teacher model for knowledge distillation. In particular, we transfer knowledge between the two networks' *feature* spaces, since this has been shown [36] to achieve better performance than reconstructing images synthesized by the teacher [20]. With different numbers of channels between teacher and student layers, Li *et al.* [36] introduce auxiliary, learnable layers that project the student features into the same dimensional space as the teacher, as shown in Fig. 3. Whilst equalizing the number of channels between the two networks, these layers can also impact the efficacy of distillation, since some information can be stored in these extra layers. To avoid information loss, we propose to encourage similarity between the two feature spaces directly.

3.3.1 Similarity-based Knowledge Distillation

We develop our distillation method based on centered kernel alignment (CKA) [14, 15], a similarity index between two matrices, $X \in \mathbb{R}^{n \times p_1}$ and $Y \in \mathbb{R}^{n \times p_2}$, where after centering the kernel alignment (KA) is calculated, which is defined as[1]

$$\mathrm{KA}(X, Y) = \frac{\|Y^{\mathrm{T}} X\|_{\mathrm{F}}^2}{\|X^{\mathrm{T}} X\|_{\mathrm{F}} \|Y^{\mathrm{T}} Y\|_{\mathrm{F}}}. \tag{1}$$

It is invariant to an orthogonal transform and isotropic scaling of the rows, but is sensitive to an invertible linear transform. Importantly, p_1 and p_2 can differ. Kornblith *et al.* [34] use this index to compute the similarity between different learned feature representations of varying lengths ($p_1 = hwc_1$ & $p_2 = hwc_2$, where h, w and c. are the height, width and number of channels of the respective layer tensors; n is the batch size).

[1]The identity $\|Y^{\mathrm{T}} X\|_{\mathrm{F}}^2 = \langle \mathrm{vec}(XX^{\mathrm{T}}), \mathrm{vec}(YY^{\mathrm{T}}) \rangle$ is used to achieve computational complexity of $\mathcal{O}(n^2 hw \max(c_1, c_2))$ [34].

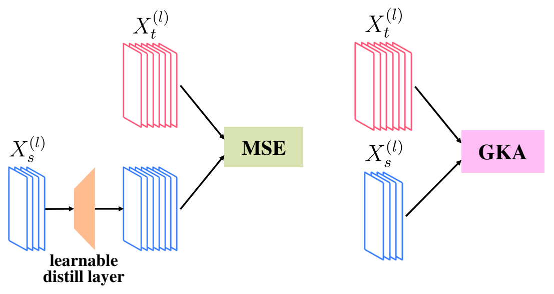

Figure 3: **Left**: Knowledge distillation with MSE loss requires extra learnable layers to map features into the same number of channels. **Right**: Our proposed GKA maximizes similarity between features directly.

Global-KA. To compare similarity between teacher and student features, we introduce a similar metric called Global-KA (*GKA*), where for the same two tensors X and Y defined in Eqn. 1, GKA is defined as follows:

$$\mathrm{GKA}(X,Y) = \mathrm{KA}(\rho(X),\rho(Y)), \tag{2}$$

where $\rho : \mathbb{R}^{n\times hwc} \to \mathbb{R}^{nhw\times c}$ is a simple reshape operation on the input matrix. Unlike CKA, which sums similarity between two batches of features over channels and spatial pixels, and describes batch-wise similarity, GKA sums feature similarity over channels, characterizing both batch-wise and spatial-wise similarity. The computational complexity of this operation is $\mathcal{O}(nhw \max(c_1, c_2)^2)$, which is lower than CKA if the batch size is much larger than the channel numbers. To perform distillation, we maximize the similarity between features of teacher and student networks by maximizing GKA.

3.3.2 Distillation Loss

We conduct distillation on the feature space. Let $\mathcal{S}_{\mathrm{KD}}$ denote the set of layers for performing knowledge distillation, whereas $X_t^{(l)}$ and $X_s^{(l)}$ denote feature tensors of layer l from the teacher and student networks, respectively. We minimize the distillation loss $\mathcal{L}_{\mathrm{dist}}$ as follows:

$$\mathcal{L}_{\mathrm{dist}} = -\sum_{l\in\mathcal{S}_{\mathrm{KD}}} \mathrm{GKA}(X_t^{(l)}, X_s^{(l)}), \tag{3}$$

where the minus sign is introduced as we intend to maximize feature similarity between student and teacher models.

3.4. Learning

We train teacher networks using the original loss functions, which includes an adversarial loss $\mathcal{L}_{\mathrm{adv}}$ as follows:

$$\mathcal{L}_{\mathrm{adv}} = \mathbb{E}_{\mathbf{x},\mathbf{y}}\left[\log D(\mathbf{x},\mathbf{y})\right] + \mathbb{E}_{\mathbf{x}}\left[\log(1 - D(\mathbf{x}, G(\mathbf{x})))\right], \tag{4}$$

where $\mathbf{x}$ and $\mathbf{y}$ denote the input and real images, and D and G denote the discriminator and generator, respectively.

Full objective for student. For the training of student generator for CycleGAN, we adopt the setting from [36] where we use the data generated from teacher network to form paired data and train the student the same way as Pix2pix with a reconstruction loss $\mathcal{L}_{\mathrm{recon}}$. Therefore, for CycleGAN and Pix2pix, the overall loss function for student training is:

$$\mathcal{L}_{\mathrm{T}} = \lambda_{\mathrm{adv}}\mathcal{L}_{\mathrm{adv}} + \lambda_{\mathrm{recon}}\mathcal{L}_{\mathrm{recon}} + \lambda_{\mathrm{dist}}\mathcal{L}_{\mathrm{dist}}. \tag{5}$$

For the training of GauGAN, there is an additional feature matching loss $\mathcal{L}_{\mathrm{fm}}$ [70], and the overall loss function is as follows:

$$\mathcal{L}_{\mathrm{T}} = \lambda_{\mathrm{adv}}\mathcal{L}_{\mathrm{adv}} + \lambda_{\mathrm{recon}}\mathcal{L}_{\mathrm{recon}} + \lambda_{\mathrm{fm}}\mathcal{L}_{\mathrm{fm}} + \lambda_{\mathrm{dist}}\mathcal{L}_{\mathrm{dist}}. \tag{6}$$

λ_{adv}, λ_{recon}, λ_{dist} and λ_{fm} in Eqn. 5 and Eqn. 6 indicate the hyper-parameters that balance the losses.

4. Experiments

In this section, we show the results of compressing image-to-image models. We introduce more details about network training and architectures, together with more qualitative results in the *supplementary materials*.

4.1. Basic Setting

Models. We conduct experiments on generation models, including Pix2pix [29], CycleGAN [83], and GauGAN [57]. Following [36], we inherit the teacher discriminator by using the same architecture and the pre-trained weights, and finetune it with the student generator for student training.

Datasets. We examine our method on the following datasets. *Horse→Zebra* and *Zebra→Horse* are two datasets from CycleGAN [83], which converts horse images to zebra and vice versa. There are $1,187$ horse images and $1,474$ zebra images. *Cityscapes* [13] is a dataset for mapping semantic inputs to images of street scenes. There are $2,975$ training and 500 validation data, and we apply Pix2pix and GauGAN models on it. *Map→Aerial photo* contains $2,194$ images [29], and we apply Pix2pix model on it.

Evaluation metrics. We adopt two standard metrics for the evaluation of generative models. For the Cityspaces dataset, we follow existing works [29, 57] to use a semantic segmentation metric to evaluate the quality of synthetic images. We run an image segmentation model, which is DRN-D-105 [75], on the generated images to calculate mean Intersection over Union (mIoU). A higher value of mIoU indicates better quality of generated images. For other datasets, we apply commonly used Fréchet Inception Distance (FID) [25], as it estimates the distribution between real and generated images. We also adopt a recent proposed metric named Kernel Inception Distance (KID) [2] for more thorough comparison. A lower FID or KID value indicates better model performance.

Table 1: Quantitative comparison between different compression techniques for Image-to-Image models. We use mIoU to evaluate the generation quality of Cityspaces and FID for other datasets. Higher mIoU or lower FID indicates better performance.

Model	Dataset	Method	MACs	FID↓	mIoU↑
CycleGAN	Horse→Zebra	Original [83, 36]	56.8B	61.53	-
		Shu *et al.* [63]	13.4B	96.15	-
		AutoGAN Distiller [20]	6.39B	83.60	-
		GAN Slimming [68]	11.25B	86.09	-
		GAN Lottery [11]	~11.35B†	~83.00†	-
		Li *et al.* [36]	2.67B	71.81	-
		CAT (Ours)	**2.55B**	**60.18**	-
	Zebra→Horse	Original [83, 68]	56.8B	148.81	-
		GAN Slimming [68]	11.81B	120.01	-
		CAT (Ours)	2.59B	142.68	-
Pix2pix	Cityscapes	Original [29, 36]	56.8B	-	42.06
		Li *et al.* [36]	5.66B	-	40.77
		CAT (Ours)	**5.57B**	-	**42.53**
	Map→Aerial photo	Original [29, 36]	56.8B	47.76	-
		Li *et al.* [36]	4.68B	48.02	-
		CAT (Ours)	**4.59B**	**44.96**	-
GauGAN	Cityscapes	Original [57, 36]	281B	-	62.18
		Li *et al.* [36]	31.7B	-	61.22
		CAT-A (Ours)	**29.9B**	-	**62.35**
		CAT-B (Ours)	5.52B	-	54.71

† Estimated from Fig. 11 in [**?**].

Table 2: Further quantitative comparison on KID between different compression techniques for Image-to-Image models, where lower KID indicates better performance.

Model	Dataset	Method	MACs	KID↓
CycleGAN	Horse→Zebra	Original [83]	56.8B	0.020±0.002
		CAT (Ours)	**2.55B**	**0.017±0.002**
	Zebra→Horse	Original [83]	56.8B	0.030±0.002
		CAT (Ours)	2.59B	0.036±0.002
Pix2pix	Map→Aerial	Original [29]	56.8B	0.154±0.010
		CAT (Ours)	**4.6B**	**0.009±0.002**
GauGAN	Cityscapes	Original [57]	281B	0.026±0.003
		CAT-A (Ours)	**29.9B**	**0.014±0.002**
		CAT-B (Ours)	**5.5B**	**0.013±0.002**

4.2. Comparison Results

Quantitative results. We compare our method with existing studies for image generation tasks on various datasets. The results are summarized in Tab. 1 and Tab. 2. We can see that for all datasets included, our models consume the smallest MACs while achieving comparable and mostly the best performance. Particularly, we achieve better performance than the original models for almost all datasets while reducing computational cost significantly. For example, on CycleGAN, our method results in a large compression ratio as the MACs is saved from 56.8B to 2.55B (22.3×) or 2.59B (21.9×), while at the same time, the model gets better performance as FID is reduced from 61.53 to 60.18 for Horse→Zebra and from 148.8 to 142.7 for Zebra→Horse. For the Cityscapes dataset with Pix2pix model, we compress the model from 56.8B to 5.57B MACs, which is 10.2× smaller, while increase the mIoU from 42.06 to 42.53. Again, for Pix2pix on the Map→Aerial photo dataset, the MACs is reduced from 56.8B to 4.59B by our method, with a compression ratio of 12.4×, whereas the FID is improved and reduced from 47.76 to 44.94.

To further verify the effectiveness of our method for compressing generative models, we experiment on GauGAN with two target MACs: 30B and 5.6B. We choose 5.6B as it is similar to our compressed Pix2pix model on Cityscapes. We find that with 30B MACs, which is 9.4× smaller than GauGAN, the mIoU of our model is better

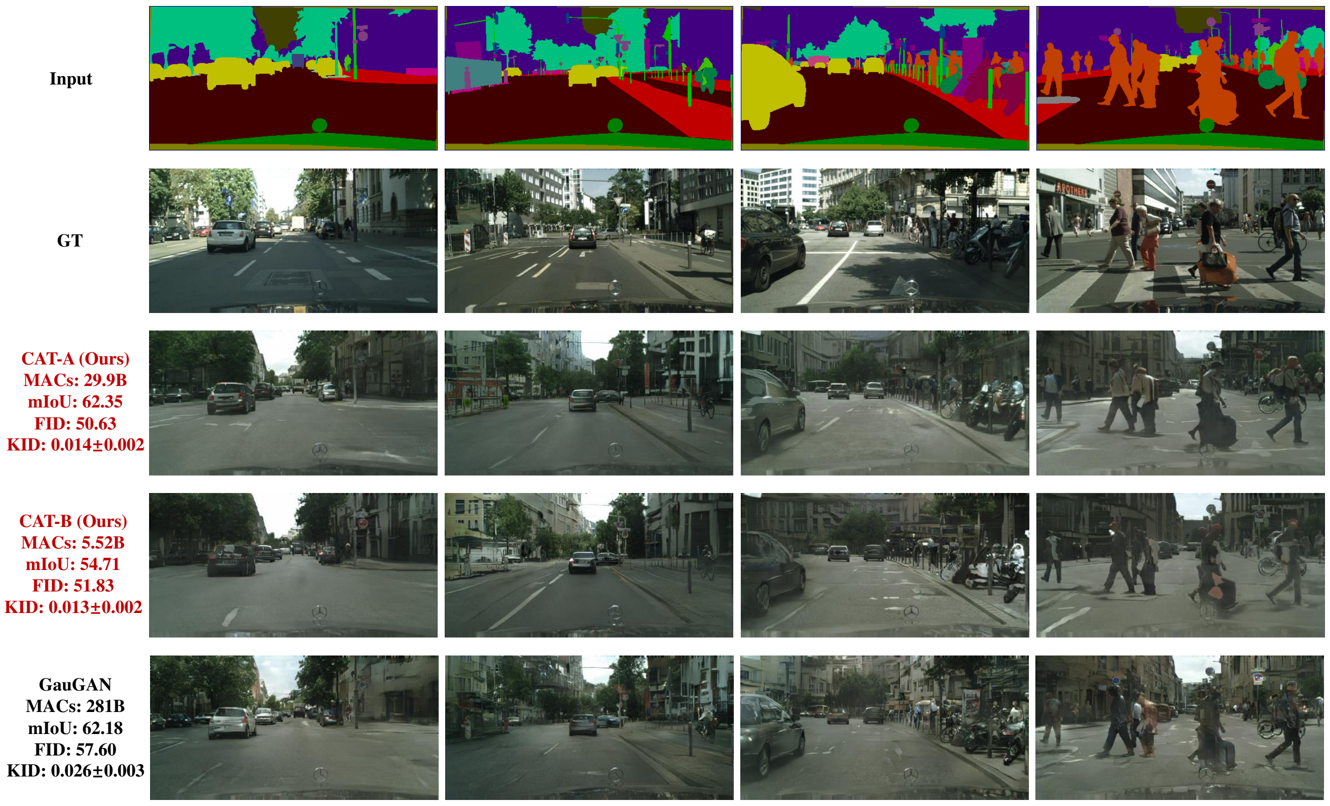

Figure 4: Qualitative results on Cityscapes dataset. Images generated by our compressed model (CAT-A, third row) have higher mIoU and lower FID than the original GauGAN model (fifth row), even with much reduced computational cost. For our CAT-B model (fourth row, $50.9\times$ compressed than GauGAN), although it has lower mIoU, the CAT-B model can synthesize higher fidelity images (lower FID) than GauGAN.

than the original, which is increased from 62.18 to 62.35. We further compress the model to less than 5.6B with a compression ratio of $50.9\times$, and the mIoU is reduced to 54.71. However, it is still much better than that from the Pix2pix model. These demonstrate that our method is a sound technique for compressing image-to-image models, and provides the state-of-the-art trade-off between computation complexity and image generation performance.

Qualitative results. We further show qualitative results to illustrate the effectiveness of our method. Fig. 4 provides samples on Cityspaces, including input segmentation maps, ground-truth (GT), and generated images by different methods. Our compressed model (CAT-A) achieves better quality (higher mIoU and lower FID) than GauGAN. For example, for the leftmost image in Fig. 4, the back of the car synthesized by CAT-A is clearer than GauGAN, and CAT-A generates less blurry human images than GauGAN for the rightmost image. CAT-B, which has much-reduced MACs than GauGAN ($50.9\times$), can also achieve better image fidelity (lower FID) than GauGAN. For Map→Aerial photo with Pix2pix (Fig. 5), our method generates images with better quality for the river and buildings than the original Pix2pix model. For Horse→Zebra on CycleGAN, our method can synthesize better zebra images for challenging input horse images, where the CycleGAN fails to generate.

The examples shown in Fig. 4 & 5 demonstrate that our compression technique is an effective method for saving the computational cost of generative models. Besides, the compressed models can surpass the original models, even though they require much reduced computational cost and, thus, are more efficient during inference. These results indicate significant redundancy in the original large generators, and it is worth further studying the extreme of these generative models in terms of performance-efficiency trade-off.

Analysis of searching cost. Here we show the analysis of searching costs for finding a student network. Our method can search the architecture under a pre-defined computational budget with a much reduced searching cost compared with previous state-of-the-art compressing method [36]. Tab. 3 provides the searching cost of the two methods on various datasets and models. As can be seen, our method is at least $10,000\times$ times faster for searching. The searching time for the previous method [36] is estimated by only including the time for training a supernet, which is designed

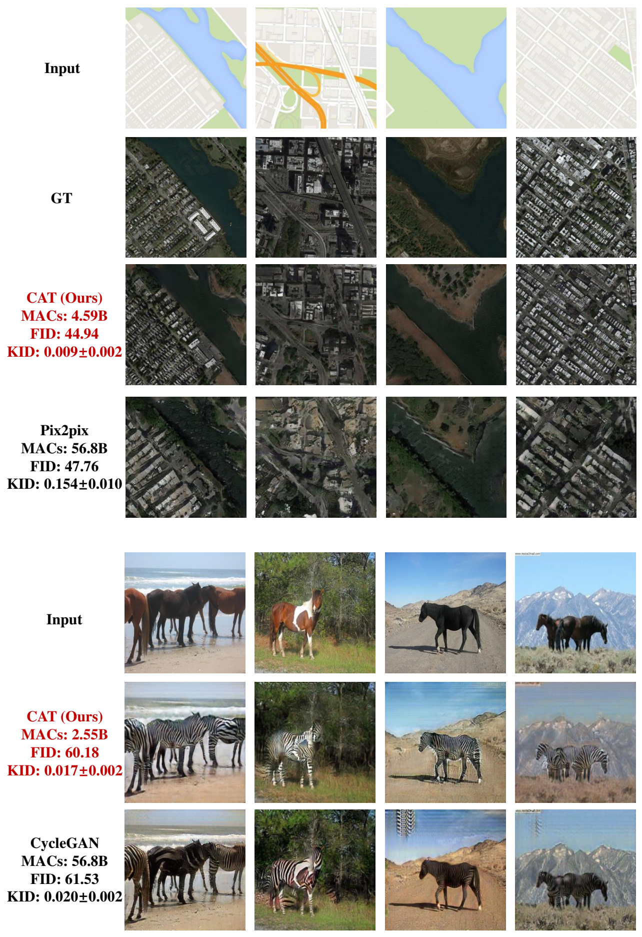

Figure 5: Qualitative results on Map→Aerial photo (top four rows) and Horse→Zebra datasets (bottom three rows). Compared with original networks (Pix2pix and CycleGAN), our models have much reduced MACs and can generate images with higher fidelity (lower FID) by synthesizing textures that are not well-handled by the original large models.

for architecture search. We estimate it as 20 hours with 1 GPU for the CycleGAN and Pix2pix models and 40 hours with 8 GPUs for the GauGAN model, both of which are much shorter than those required in practice and thus serves as a lower bound. Besides, we have ignored the time required for searching a student network from the supernet for [36], which is also non-negligible. For example, for Cityscapes with Pix2pix model, the supernet includes more than $5,000$ possible architectures, and each requires around 3 minutes with 1 GPU for evaluation, resulting in several days of architecture search. Despite, we do not take this process of [36] into account for time-estimation in Tab. 3.

5. Conclusion

In this paper, we study the problem of compressing generative models, especially the generators for image-to-image tasks. We show the problem can be tackled by using a powerful teacher model, which is not restricted to teach a student through knowledge distillation, but can serve as a supernet to search efficient architecture (for student) under pre-defined computational budgets.

Table 3: Architecture search cost, measured in seconds of GPU computation, for our method vs. Li *et al.* [36], across different models.

Model	Dataset	Method	Search Cost (GPU Seconds)
CycleGAN	Horse→Zebra	Li *et al.* [36]	$\gtrsim 7.2 \times 10^4$
		CAT (Ours)	**3.81**
	Zebra→Horse	Li *et al.* [36]	$\gtrsim 7.2 \times 10^4$
		CAT (Ours)	**3.62**
Pix2pix	Cityscapes	Li *et al.* [36]	$\gtrsim 7.2 \times 10^4$
		CAT (Ours)	**4.28**
	Map→Aerial photo	Li *et al.* [36]	$\gtrsim 7.2 \times 10^4$
		CAT (Ours)	**4.33**
GauGAN	Cityscapes	Li *et al.* [36]	$\gtrsim 1.2 \times 10^6$
		CAT-A (Ours)	**8.22**
		CAT-B (Ours)	**6.20**

Specifically, our framework is built upon a newly designed teacher model, which incorporates the proposed IncResBlock. We show such teacher model contains a large search space where efficient student architecture can be determined through network searching. The searching process is implemented with our proposed one-step pruning algorithm, which can be conducted with negligible efforts. We also introduce a similarity-based knowledge distillation technique to train student network, where feature similarity between student and teacher is measured directly by the proposed GKA index. With our method, we can obtain networks that have similar or even better performance than original Pix2pix, CycleGAN, and GauGAN models on various datasets. More importantly, our networks have much reduced MACs than their original counterparts.

Our work demonstrates that there remains redundancy in existing generative models, and we can achieve improved performance, *e.g.*, synthesizing images with better fidelity, with much reduced computational cost. It is worth further investigating the ability of generative models to synthesize images with high quality under an extremely constrained computational budget, which we leave for future study.

6. Acknowledgement

The Authors would like to appreciate Jieru Mei from John Hopkins University for invaluable technical discussion. Also this research is partially supported by National Science Foundation CNS-1909172.

References

[1] Angeline Aguinaldo, Ping-Yeh Chiang, Alex Gain, Ameya Patil, Kolten Pearson, and Soheil Feizi. Compressing gans using knowledge distillation. *arXiv preprint arXiv:1902.00159*, 2019. 2

[2] Mikołaj Bińkowski, Dougal J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018. 5

[3] Tolga Bolukbasi, Joseph Wang, Ofer Dekel, and Venkatesh Saligrama. Adaptive neural networks for efficient inference. *arXiv preprint arXiv:1702.07811*, 2017. 1, 2

[4] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018. 1

[5] Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once-for-all: Train one network and specialize it for efficient deployment. *arXiv preprint arXiv:1908.09791*, 2019. 2, 3

[6] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. *arXiv preprint arXiv:1812.00332*, 2018. 2

[7] Menglei Chai, Jian Ren, and Sergey Tulyakov. Neural hair rendering. In *Eur. Conf. Comput. Vis.*, 2020. 1

[8] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. In *Advances in Neural Information Processing Systems*, pages 742–751, 2017. 2

[9] Hanting Chen, Yunhe Wang, Han Shu, Changyuan Wen, Chunjing Xu, Boxin Shi, Chao Xu, and Chang Xu. Distilling portable generative adversarial networks for image translation. *arXiv preprint arXiv:2003.03519*, 2020. 2

[10] Xin Chen, Lingxi Xie, Jun Wu, and Qi Tian. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1294–1303, 2019. 2

[11] Xuxi Chen, Zhenyu Zhang, Yongduo Sui, and Tianlong Chen. {GAN}s can play lottery tickets too. In *Submitted to International Conference on Learning Representations*, 2021. under review. 1, 2, 6

[12] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018. 1

[13] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 5

[14] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment. *The Journal of Machine Learning Research*, 13(1):795–828, 2012. 4

[15] Nello Cristianini, Jaz Kandola, Andre Elisseeff, and John Shawe-Taylor. On kernel target alignment. In *Innovations in machine learning*, pages 205–256. Springer, 2006. 4

[16] Caiwen Ding, Siyu Liao, Yanzhi Wang, Zhe Li, Ning Liu, Youwei Zhuo, Chao Wang, Xuehai Qian, Yu Bai, Geng Yuan, et al. Circnn: accelerating and compressing deep neural networks using block-circulant weight matrices. In *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 395–408, 2017. 2

[17] Caiwen Ding, Ao Ren, Geng Yuan, Xiaolong Ma, Jiayu Li, Ning Liu, Bo Yuan, and Yanzhi Wang. Structured weight matrices-based hardware accelerators in deep neural networks: Fpgas and asics. In *Proceedings of the 2018 on Great Lakes Symposium on VLSI*, pages 353–358, 2018. 1, 2

[18] Caiwen Ding, Shuo Wang, Ning Liu, Kaidi Xu, Yanzhi Wang, and Yun Liang. Req-yolo: A resource-aware, efficient quantization framework for object detection on fpgas. In *Proceedings of the 2019 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pages 33–42, 2019. 1

[19] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018. 2

[20] Yonggan Fu, Wuyang Chen, Haotao Wang, Haoran Li, Yingyan Lin, and Zhangyang Wang. Autogan-distiller: Searching to compress generative adversarial networks. *arXiv preprint arXiv:2006.08198*, 2020. 1, 2, 4, 6

[21] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 1

[22] Jianping Gou, Baosheng Yu, Stephen John Maybank, and Dacheng Tao. Knowledge distillation: A survey. *arXiv preprint arXiv:2006.05525*, 2020. 2

[23] Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun. Single path one-shot neural architecture search with uniform sampling. In *European Conference on Computer Vision*, pages 544–560. Springer, 2020. 2, 3

[24] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015. 1

[25] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPs*, 2017. 5

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

[27] 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

[28] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 3

[29] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017. 1, 2, 5, 6

[30] Qing Jin, Linjie Yang, and Zhenyu Liao. Towards efficient training for neural network quantization. *arXiv preprint arXiv:1912.10207*, 2019. 1

[31] Qing Jin, Linjie Yang, and Zhenyu Liao. Adabits: Neural network quantization with adaptive bit-widths. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2146–2156, 2020. 1

[32] Qing Jin, Linjie Yang, Zhenyu Liao, and Xiaoning Qian. Neural network quantization with scale-adjusted training. In *The British Machine Vision Conference (BMVC)*, 2020. 1

[33] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8110–8119, 2020. 1

[34] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. *arXiv preprint arXiv:1905.00414*, 2019. 4

[35] Hongjia Li, Ning Liu, Xiaolong Ma, Sheng Lin, Shaokai Ye, Tianyun Zhang, Xue Lin, Wenyao Xu, and Yanzhi Wang. Admm-based weight pruning for real-time deep learning acceleration on mobile devices. In *Proceedings of the 2019 on Great Lakes Symposium on VLSI*, pages 501–506, 2019. 1, 2

[36] Muyang Li, Ji Lin, Yaoyao Ding, Zhijian Liu, Jun-Yan Zhu, and Song Han. Gan compression: Efficient architectures for interactive conditional gans. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5284–5294, 2020. 1, 2, 3, 4, 5, 6, 7, 8

[37] Yingwei Li, Xiaojie Jin, Jieru Mei, Xiaochen Lian, Linjie Yang, Cihang Xie, Qihang Yu, Yuyin Zhou, Song Bai, and Alan L Yuille. Neural architecture search for lightweight non-local networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10297–10306, 2020. 2

[38] Sheng Lin, Xiaolong Ma, Shaokai Ye, Geng Yuan, Kaisheng Ma, and Yanzhi Wang. Toward extremely low bit and lossless accuracy in dnns with progressive admm. *arXiv preprint arXiv:1905.00789*, 2019. 1

[39] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018. 2

[40] Ning Liu, Xiaolong Ma, Zhiyuan Xu, Yanzhi Wang, Jian Tang, and Jieping Ye. Autoslim: An automatic dnn structured pruning framework for ultra-high compression rates. *arXiv preprint arXiv:1907.03141*, 2019. 1, 2

[41] Ning Liu, Xiaolong Ma, Zhiyuan Xu, Yanzhi Wang, Jian Tang, and Jieping Ye. Autocompress: An automatic dnn structured pruning framework for ultra-high compression rates. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4876–4883, 2020. 2

[42] Shaoshan Liu, Bin Ren, Xipeng Shen, and Yanzhi Wang. Cocopie: Making mobile ai sweet as pie–compression-compilation co-design goes a long way. *arXiv preprint arXiv:2003.06700*, 2020. 1, 2

[43] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2736–2744, 2017. 1, 2, 3

[44] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. *arXiv preprint arXiv:1810.05270*, 2018. 1, 2, 3

[45] David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. Unifying distillation and privileged information. *arXiv preprint arXiv:1511.03643*, 2015. 2

[46] Zhichao Lu, Kalyanmoy Deb, Erik Goodman, Wolfgang Banzhaf, and Vishnu Naresh Boddeti. Nsganetv2: Evolutionary multi-objective surrogate-assisted neural architecture search. In *European Conference on Computer Vision*, pages 35–51. Springer, 2020. 2

[47] Zhichao Lu, Gautam Sreekumar, Erik Goodman, Wolfgang Banzhaf, Kalyanmoy Deb, and Vishnu Naresh Boddeti. Neural architecture transfer. *arXiv preprint arXiv:2005.05859*, 2020. 2

[48] Ping Luo, Zhenyao Zhu, Ziwei Liu, Xiaogang Wang, Xiaoou Tang, et al. Face model compression by distilling knowledge from neurons. In *AAAI*, pages 3560–3566, 2016. 2

[49] Xiaolong Ma, Fu-Ming Guo, Wei Niu, Xue Lin, Jian Tang, Kaisheng Ma, Bin Ren, and Yanzhi Wang. Pconv: The missing but desirable sparsity in dnn weight pruning for real-time execution on mobile devices. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5117–5124, 2020. 1, 2

[50] Xiaolong Ma, Sheng Lin, Shaokai Ye, Zhezhi He, Linfeng Zhang, Geng Yuan, Sia Huat Tan, Zhengang Li, Deliang Fan, Xuehai Qian, et al. Non-structured dnn weight pruning–is it beneficial in any platform? *arXiv preprint arXiv:1907.02124*, 2019. 1, 2

[51] Xiaolong Ma, Wei Niu, Tianyun Zhang, Sijia Liu, Sheng Lin, Hongjia Li, Wujie Wen, Xiang Chen, Jian Tang, Kaisheng Ma, et al. An image enhancing pattern-based sparsity for real-time inference on mobile devices. In *European Conference on Computer Vision*, pages 629–645. Springer, 2020. 1, 2

[52] Xiaolong Ma, Geng Yuan, Sheng Lin, Zhengang Li, Hao Sun, and Yanzhi Wang. Resnet can be pruned 60×: Introducing network purification and unused path removal (p-rm) after weight pruning. In *2019 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH)*, pages 1–2. IEEE, 2019. 1, 2

[53] Jieru Mei, Yingwei Li, Xiaochen Lian, Xiaojie Jin, Linjie Yang, Alan Yuille, and Jianchao Yang. Atomnas: Fine-grained end-to-end neural architecture search. *arXiv preprint arXiv:1912.09640*, 2019. 2, 3, 4

[54] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. 1

[55] Wei Niu, Xiaolong Ma, Sheng Lin, Shihao Wang, Xuehai Qian, Xue Lin, Yanzhi Wang, and Bin Ren. Patdnn: Achieving real-time dnn execution on mobile devices with pattern-based weight pruning. In *Proceedings of the Twenty-Fifth*

International Conference on Architectural Support for Programming Languages and Operating Systems, pages 907–922, 2020. 1, 2

[56] Wei Niu, Xiaolong Ma, Yanzhi Wang, and Bin Ren. 26ms inference time for resnet-50: Towards real-time execution of all dnns on smartphone. *arXiv preprint arXiv:1905.00571*, 2019. 1, 2

[57] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2337–2346, 2019. 1, 2, 5, 6

[58] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc Le, and Alex Kurakin. Large-scale evolution of image classifiers. *arXiv preprint arXiv:1703.01041*, 2017. 2

[59] Jian Ren, Menglei Chai, Sergey Tulyakov, Chen Fang, Xiaohui Shen, and Jianchao Yang. Human motion transfer from poses in the wild. *arXiv preprint arXiv:2004.03142*, 2020. 1

[60] Jian Ren, Zhe Li, Jianchao Yang, Ning Xu, Tianbao Yang, and David J Foran. Eigen: Ecologically-inspired genetic approach for neural network structure searching from scratch. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9059–9068, 2019. 2

[61] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 3

[62] Runbin Shi, Peiyan Dong, Tong Geng, Yuhao Ding, Xiaolong Ma, Hayden K-H So, Martin Herbordt, Ang Li, and Yanzhi Wang. Csb-rnn: A faster-than-realtime rnn acceleration framework with compressed structured blocks. *arXiv preprint arXiv:2005.05758*, 2020. 2

[63] Han Shu, Yunhe Wang, Xu Jia, Kai Han, Hanting Chen, Chunjing Xu, Qi Tian, and Chang Xu. Co-evolutionary compression for unpaired image translation. In *Int. Conf. Comput. Vis.*, pages 3235–3244, 2019. 1, 2, 6

[64] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261*, 2016. 3

[65] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019. 2

[66] Yu Tian, Jian Ren, Menglei Chai, Kyle Olszewski, Xi Peng, Dimitris N. Metaxas, and Sergey Tulyakov. A good image generator is what you need for high-resolution video synthesis. In *International Conference on Learning Representations*, 2021. 1

[67] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 3

[68] Haotao Wang, Shupeng Gui, Haichuan Yang, Ji Liu, and Zhangyang Wang. Gan slimming: All-in-one gan compression by a unified optimization framework. In *Eur. Conf. Comput. Vis.*, 2020. 1, 2, 4, 6

[69] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. *arXiv preprint arXiv:1808.06601*, 2018. 1

[70] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 1, 5

[71] Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 409–424, 2018. 1, 2

[72] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10734–10742, 2019. 2

[73] Linjie Yang and Qing Jin. Fracbits: Mixed precision quantization via fractional bit-widths. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence*, 2021. 1

[74] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4133–4141, 2017. 2

[75] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 472–480, 2017. 5

[76] Jiahui Yu and Thomas Huang. Autoslim: Towards one-shot architecture search for channel numbers. *arXiv preprint arXiv:1903.11728*, 2019. 2

[77] Jiahui Yu and Thomas S Huang. Universally slimmable networks and improved training techniques. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1803–1811, 2019. 2

[78] Jiahui Yu, Pengchong Jin, Hanxiao Liu, Gabriel Bender, Pieter-Jan Kindermans, Mingxing Tan, Thomas Huang, Xiaodan Song, Ruoming Pang, and Quoc Le. Bignas: Scaling up neural architecture search with big single-stage models. *arXiv preprint arXiv:2003.11142*, 2020. 2

[79] Qihang Yu, Yingwei Li, Jieru Mei, Yuyin Zhou, and Alan L Yuille. Cakes: Channel-wise automatic kernel shrinking for efficient 3d network. *arXiv preprint arXiv:2003.12798*, 2020. 2

[80] Geng Yuan, Xiaolong Ma, Caiwen Ding, Sheng Lin, Tianyun Zhang, Zeinab S Jalali, Yilong Zhao, Li Jiang, Sucheta Soundarajan, and Yanzhi Wang. An ultra-efficient memristor-based dnn framework with structured weight pruning and quantization using admm. In *2019 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, pages 1–6. IEEE, 2019. 1, 2

[81] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks.

In *International Conference on Machine Learning*, pages 7354–7363. PMLR, 2019. 1

[82] Tianyun Zhang, Shaokai Ye, Xiaoyu Feng, Xiaolong Ma, Kaiqi Zhang, Zhengang Li, Jian Tang, Sijia Liu, Xue Lin, Yongpan Liu, et al. Structadmm: Achieving ultrahigh efficiency in structured pruning for dnns. *IEEE Transactions on Neural Networks and Learning Systems*, 2021. 1, 2

[83] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networkss. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017. 1, 2, 5, 6

[84] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016. 2

[85] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018. 3