Spatially Sparse Inference for Generative Image Editing Supplementary Material

A Additional Implementation Details

For all models, we use block size 6 for 3×3 convolutions and block size 4 for 1×1 convolutions. For DDIM [1] and Progressive Distillation [12], we pre-compute and reuse the statistics of the original image for all group normalization layers [84]. For GAN Compression [3], we pre-compute and reuse the statistics of the original image for all instance normalization layers [82] whose resolution is higher than 16×32.

B Kernel Fusion

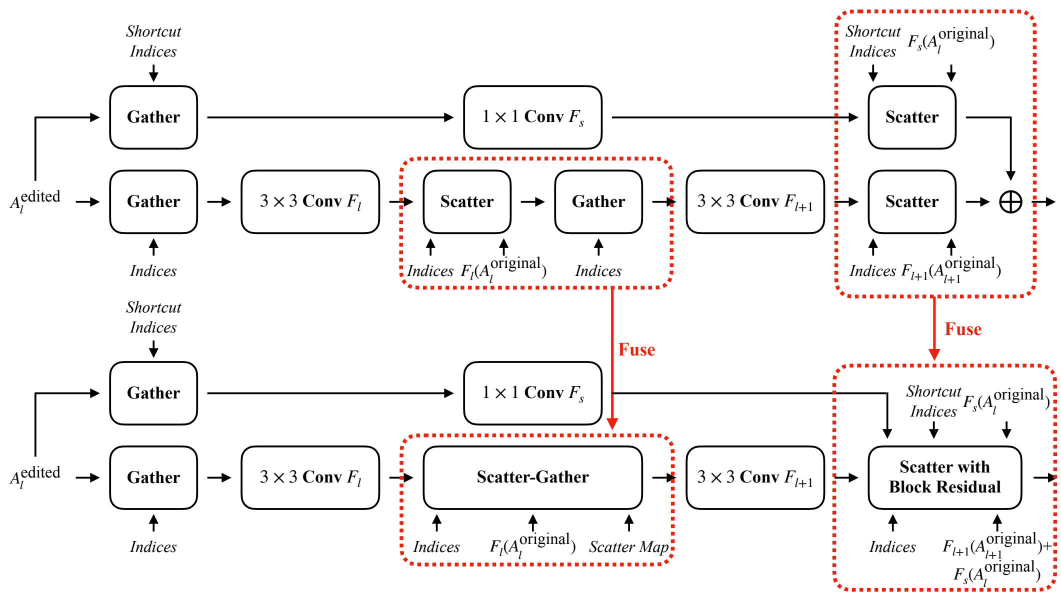


Figure 6: Visualization of kernel fusion in DDIM [1] ResBlock [88]. We omit the element-wise operations for simplicity and follow the notations in Section 3. As the kernel sizes of the convolution in the shortcut branch and main branch are different, their reduced active block indices are different (*Indices* and *Shortcut Indices*). To reduce the tensor copying overheads in `Scatter`, we fuse `Scatter` and the following `Gather` into `Scatter-Gather` and fuse the `Scatter` in the shortcut, main branch and residual addition into `Scatter with Block Residual`. We pre-compute an additional *Scatter Map* for the `Scatter-Gather` kernel.

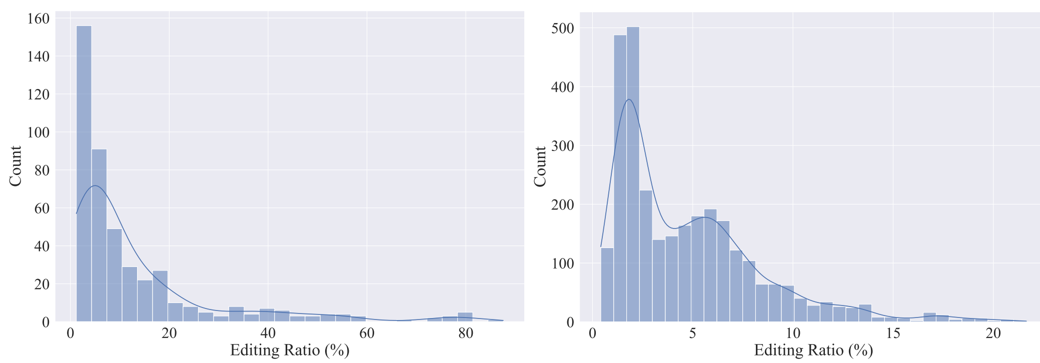
As mentioned in Section 3.2, we fuse `Scatter` and the following `Gather` into a `Scatter-Gather` operator and also fuse `Scatter` in the shortcut, main branch and residual addition together. The detailed fusion pattern is shown in Figure 6. For simplicity, we omit the element-wise operations (*e.g.*, `Nonlinearity` and `Scale+Shift`). Below we include more implementation details of each fusion design.

Scatter-Gather fusion. When a `Scatter` is directly followed by a `Gather`, we could fuse these two operators into a `Scatter-Gather` to avoid copying the original activation $F_l(A_l^{\text{original}})$. We pre-built a *Scatter Map* to indicate the index mapping from the F_l output to the previous `Scatter` output, and directly gather the active blocks from the F_l output and original activation $F_l(A_l^{\text{original}})$ with it. Note that the pre-computation is cheap and only needs to be once for each resolution.

Shortcut Scatter fusion. The 1×1 convolution in the shortcut branch consumes much less computation than the convolutions in the main branch, therefore the overheads of `Gather` and `Scatter` weigh more in the shortcut branch. We fuse the `Scatter` in the shortcut branch and main branch along with residual addition together into `Scatter with Block Residual` to reduce these overheads. Specifically, we first scatter F_{l+1} output in the pre-computed $F_{l+1}(A_l^{\text{original}}) + F_s(A_l^{\text{original}})$

(a) LSUN-Church (b) Cityscapes

Figure 7: Several examples of our synthetic editing dataset on (a) LSUN Church and (b) Cityscapes. On LSUN Church, we view the inpainted image as the original image and generate the editing by quantizing color at the corresponding regions. On Cityscapes, we generate the editing by pasting some foreground objects to the ground-truth semantic maps.



(a) LSUN Church. (b) Cityscapes.

Figure 8: Detailed editing ratio distribution of our synthetic datasets.

and add the original residual $F_s(A_l^{\text{original}})$ only at the scattered locations correspondingly according to *Indices*. Then we calibrate the final output with F_s output by adding the residual difference $F_s(A_l^{\text{edited}}) - F_s(A_l^{\text{original}})$ at the scattered locations inplace according to *Shortcut Indices*.

C Benchmark Datasets

We elaborate more details on how we build the synthetic editing dataset.

LSUN Church. Figure 7(a) shows some examples of our synthetic editing on LSUN Church. The average edited area of the whole dataset is 13.1%. The detailed distribution is shown in Figure 8a.

Cityscapes. We collect 27 foreground object semantic masks from the validation set. The objects include 4 bicycles, 1 motorcycle, 7 cars, 6 trucks, 3 buses, 5 persons, and 1 train. Figure 9 shows some visualization of the collected semantic masks. We generate the editing by randomly pasting one of these objects to the ground-truth semantic maps with augmentation. The augmentation includes random horizontal flip, resize (scale factor in $[0.8, 1.2]$), translation ($[-32, 32]$ for height and $[-64, 64]$ for width). To make the synthetic editing more reasonable, when the scale factor is larger than 1, the height translation can only be positive, otherwise, it can only be negative. Figure 7(b)

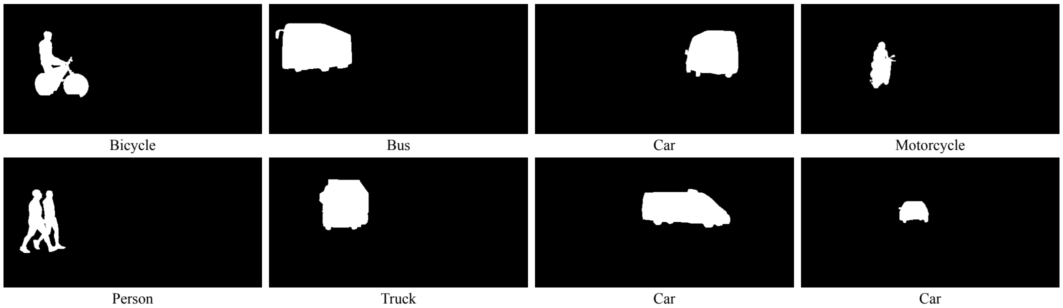


Figure 9: Several examples of our collected foreground object semantic masks.

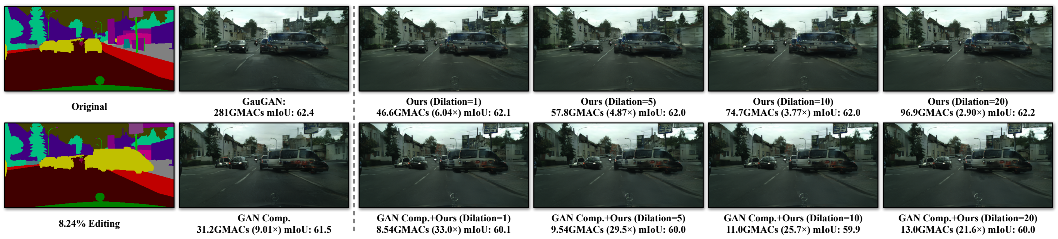


Figure 10: Visualization results of different dilation sizes on GauGAN. Although without mIoU improvement, increasing the dilation could smoothly blend the boundary between the edited region and unedited regions to improve the image quality slightly. Specifically, the shadow boundary of the added car fades when dilation increases. However, it will incur more computations.

shows some editing examples. The average editing area of the entire dataset is 4.77%. The detailed distribution is shown in Figure 8b.

D Additional Results

Dilation hyper-parameter. We show the results of our method with different dilation sizes on GauGAN in Figure 10. Increasing the dilation brings more computations but also slightly improves the image quality. Specifically, the shadow boundary of the added car fades when increasing the dilation. We choose dilation 1 as the image quality is almost the same as 20 while delivering the best speed.

Large editing. In Table 4 and Figure 11, we show the results of large editing ($\sim 35\%$) using our method. Specifically, we could achieve at most $1.7\times$ speedup on DDIM, $1.5\times$ speedup on PD256 and $1.7\times$ speedup on GauGAN without losing visual fidelity. Furthermore, in many practical cases, users can decompose a large edit into several small edits. Our method could incrementally update the results instantly when the edit is being created.

Sequential editing. In Figure 12, we show the results of sequential editing with our method. Specifically, *One-time Pre-computation* performs as well as the *Full Model*, demonstrating that our method can be applied to multiple sequential editing with only one-time pre-computation in most cases. Moreover, for extremely large edited regions, we could use SIGE to incrementally update the pre-computed features (*Incremental Pre-computation*) and condition the later editing on the recomputed one. Its results are also as good as the full model. Therefore, our method could well address the sequential editing.

Additional visualization. In Figure 13, we show additional synthetic editing visual results of DDIM [1] and Progressive Distillation [12] on LSUN Church [10]. In Figure 14, we show additional synthetic editing visual results of GauGAN on Cityscapes [11].

E License & Computation Resources

Here we show all the licenses of our used assets. The model DDIM [1], Progressive Distillation [12], GauGAN [2] and GAN Compression [3] is under MIT license, Apache license, Creative Commons license and BSD license, respectively. SDEdit is under MIT license. The license of Cityscapes [11] is here. LSUN Church [10] does not have explicit license.

Model	Editing Size	Method	MACs		3090		2080Ti		Intel Core i9-10920X		Apple M1 Pro	
			Value	Ratio	Value	Ratio	Value	Ratio	Value	Ratio	Value	Ratio
DDIM	–	Original	248G	–	37.5ms	–	54.6ms	–	609ms	–	12.9s	–
	32.9%	Ours	115G	2.2×	26.0ms	1.4×	36.9ms	1.5×	449ms	1.4×	7.53s	1.7×
PD256	–	Original	119G	–	35.1ms	–	51.2ms	–	388ms	–	6.18s	–
	32.9%	Ours	64.3G	1.9×	25.3ms	1.4×	35.1ms	1.5×	334ms	1.2×	4.47s	1.4×
GauGAN	–	Original	281G	–	45.4ms	–	49.5ms	–	682ms	–	14.1s	–
		GAN Compression	31.2G	9.0×	17.0ms	2.7×	25.0ms	2.0×	333ms	2.1×	2.11s	6.7×
	38.7%	Ours	148G	1.9×	27.9ms	1.6×	41.7ms	1.2×	512ms	1.3×	8.37s	1.7×
		GAN Comp.+Ours	18.3G	15×	15.3ms	3.0×	22.2ms	2.2×	169ms	4.0×	1.25s	11×

Table 4: Measured latency speedup of large editing on different devices. The detailed editing examples are shown in Figure 11. Our method could reduce up to 2.2× MACs, and 1.4×, 1.5×, 1.4× and 1.7× latency on NVIDIA RTX 3090, 2080Ti, Intel Core i9-10920X and M1 Pro. With GAN Compression, we could further speedup GauGAN by 4.0× on Intel Core-i9 and 11× on Apple M1 Pro.

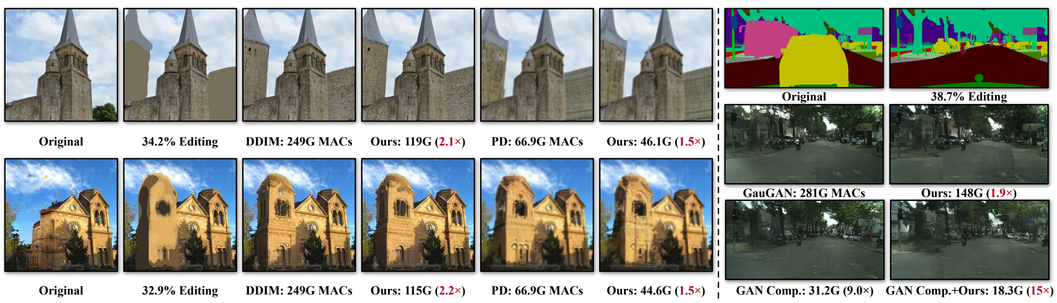


Figure 11: Qualitative results of our method under large editing. Our method could still well preserve the visual fidelity of the original model without losing global context while reducing the computation by $1.5 \sim 1.9\times$.

Since our method does not involve any model training, all our generated results are obtained on a single NVIDIA RTX 3090, which only takes $1 \sim 2$ hours to process all the test images ($\sim 7,000$ in total) including both the original models and our method. We measure the model latency on NVIDIA RTX 3090, 2080Ti, Intel Core i9-10920X CPU, and Apple M1 Pro.

F Discussion

Limitations. As discussed in Section 4.2, our method needs some additional memory to store the original activations, even though this only increases the peak GPU memory usage slightly. It may not work on some memory-constrained devices, especially for the diffusion models (*e.g.*, DDIM [1] and Progressive Distillation [12]), since our method requires storing activations of all iteration steps.

Our engine has limited speedup on convolution with low resolution. When the input resolution is low, the sparse block size needs to be even smaller to get a good sparsity, such as 1 or 2. However, such extremely small block sizes have worse memory locality and will result in low hardware efficiency.

Besides, we sometimes observe noticeable boundary between the edited region and unedited region in our generated samples of GauGAN [2]. This is because, for GauGAN model, the unedited region will also change slightly when we perform normal inference. However, since our method does not update the unedited region, there may be some color gaps between the edited and unedited region, even though the semantic is coherent. Dilating the difference mask would help reduce the gap.

Societal impact. In this paper, we investigate how to update user editing locally without losing global coherence to enable smoother interaction with the generative models. In real-world scenarios, people could use an interactive interface to edit an image, and our method could provide a quick and high-quality preview for their editing, which eases the process of visual content creation and saves energy.

However, our method can also be utilized by some malicious users to generate fake content, deceive people, and spread misinformation, which may lead to potential negative social impacts. Following previous work [9], we will also explicitly specify the usage permission of our engine with proper licenses.

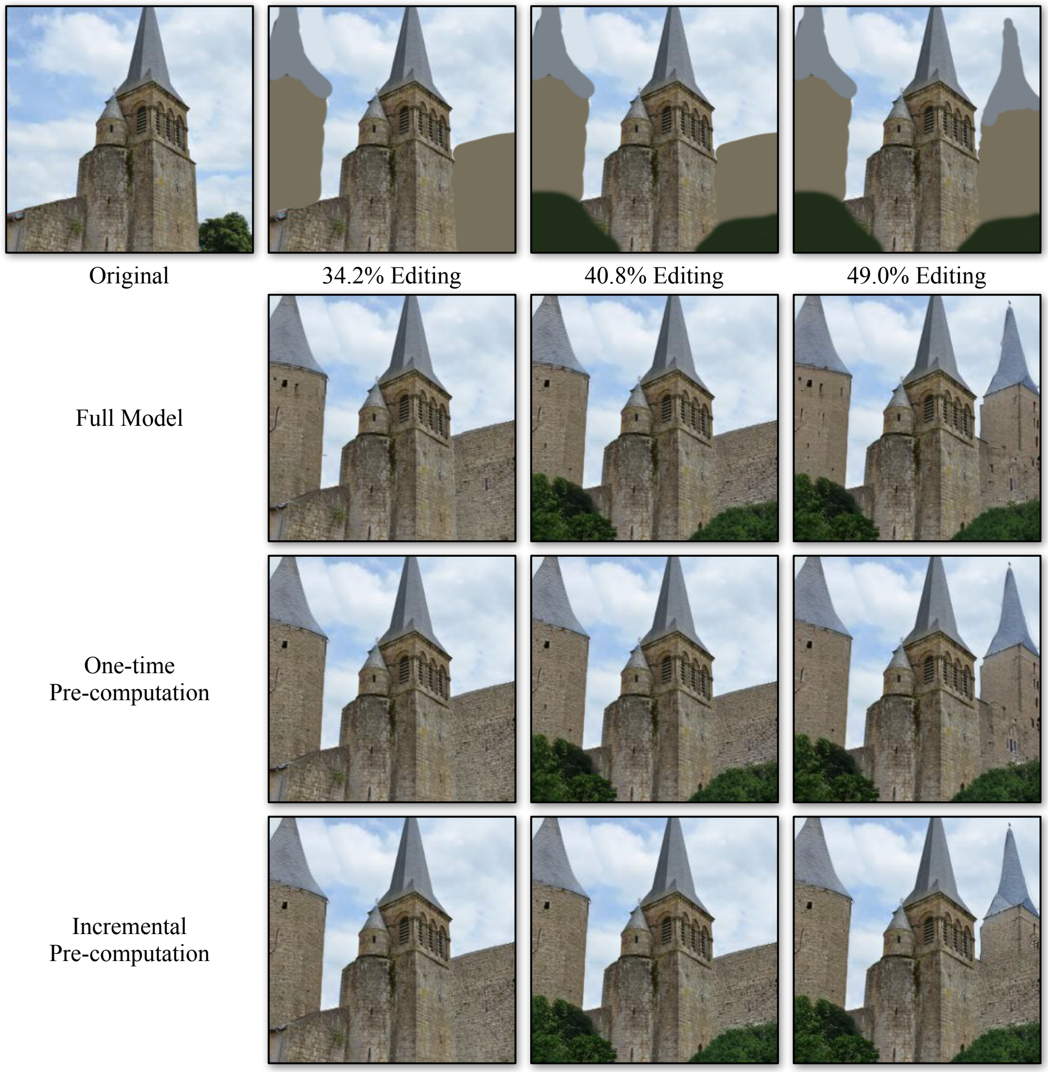


Figure 12: Sequential editing results with SIGE. *Full Model* means the results with the full model. *One-time Pre-computation* means we only pre-compute the original image features for all the editing steps. *Incremental Pre-computation* means we incrementally update the pre-computed features with SIGE before the next editing step. The image quality of all methods are quite similar.

References

[1] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2020. 1, 2, 5, 6, 9, 3, 4

[2] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019. 1, 2, 5, 6, 9, 3, 4, 7

[3] Muyang Li, Ji Lin, Yaoyao Ding, Zhijian Liu, Jun-Yan Zhu, and Song Han. Gan compression: Efficient architectures for interactive conditional gans. In *CVPR*, 2020. 1, 2, 3, 6, 7, 9

[4] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *NeurIPS*, 2014. 1, 2

[5] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015. 1, 2

[6] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020. 1, 2, 6

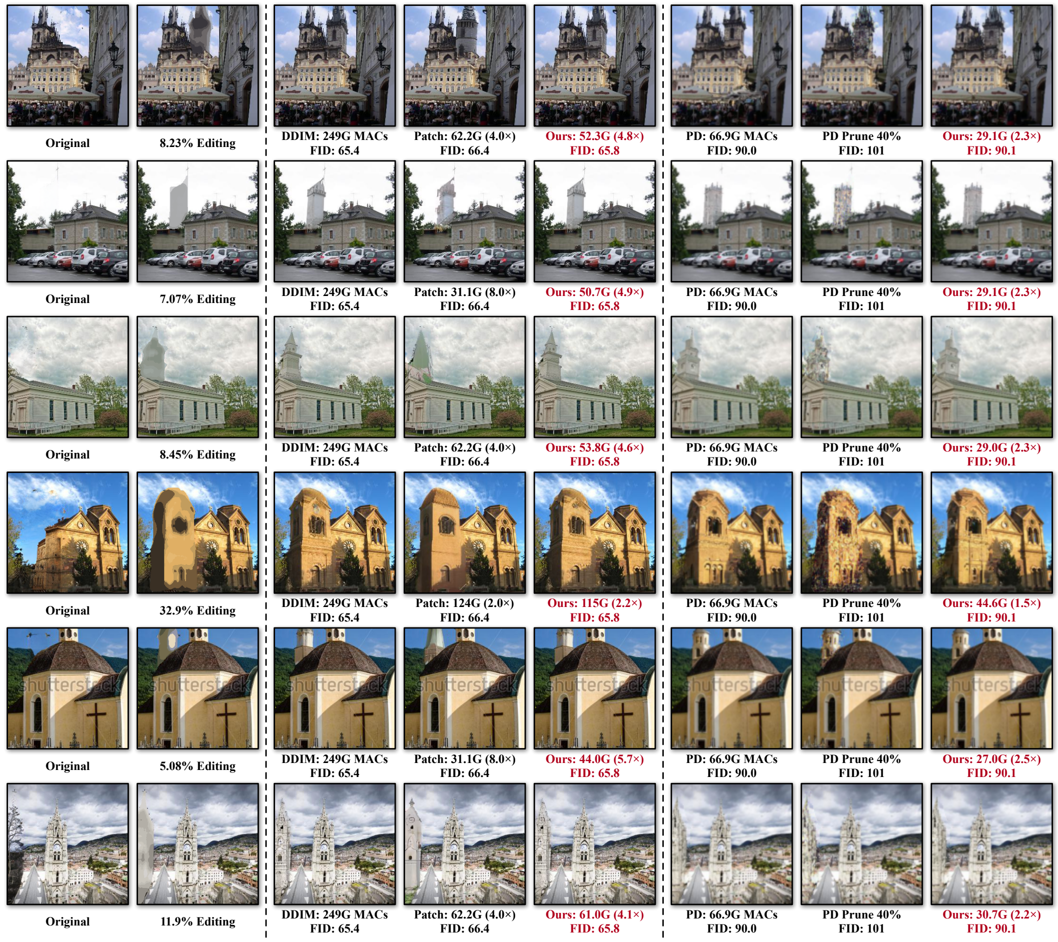


Figure 13: More visualization results on LSUN Church of DDIM [1] and Progressive Distillation. *Prune 40%*: Uniformly pruning 40% weights of the model without fine-tuning. *Patch*: Cropping the smallest image patch that covers all the edited region of the model input and blend the model output back to the original output image. Our method achieves lower FID with less MACs for both DDIM and progressive distillation.

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

[8] Patsorn Sangkloy, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. Scribbler: Controlling deep image synthesis with sketch and color. In *CVPR*, 2017. 2

[9] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *ICLR*, 2022. 2, 6, 4

[10] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 2, 6, 3

[11] 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 *CVPR*, 2016. 2, 6, 3

[12] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *ICLR*, 2021. 2, 3, 5, 6, 8, 9, 1, 4

[13] Liang Hou, Zehuan Yuan, Lei Huang, Huawei Shen, Xueqi Cheng, and Changhu Wang. Slimmable generative adversarial networks. In *AAAI*, 2021. 2, 3

[14] Yonggan Fu, Wuyang Chen, Haotao Wang, Haoran Li, Yingyan Lin, and Zhangyang Wang. Autogan-distiller: Searching to compress generative adversarial networks. In *ICML*, 2020. 2, 3

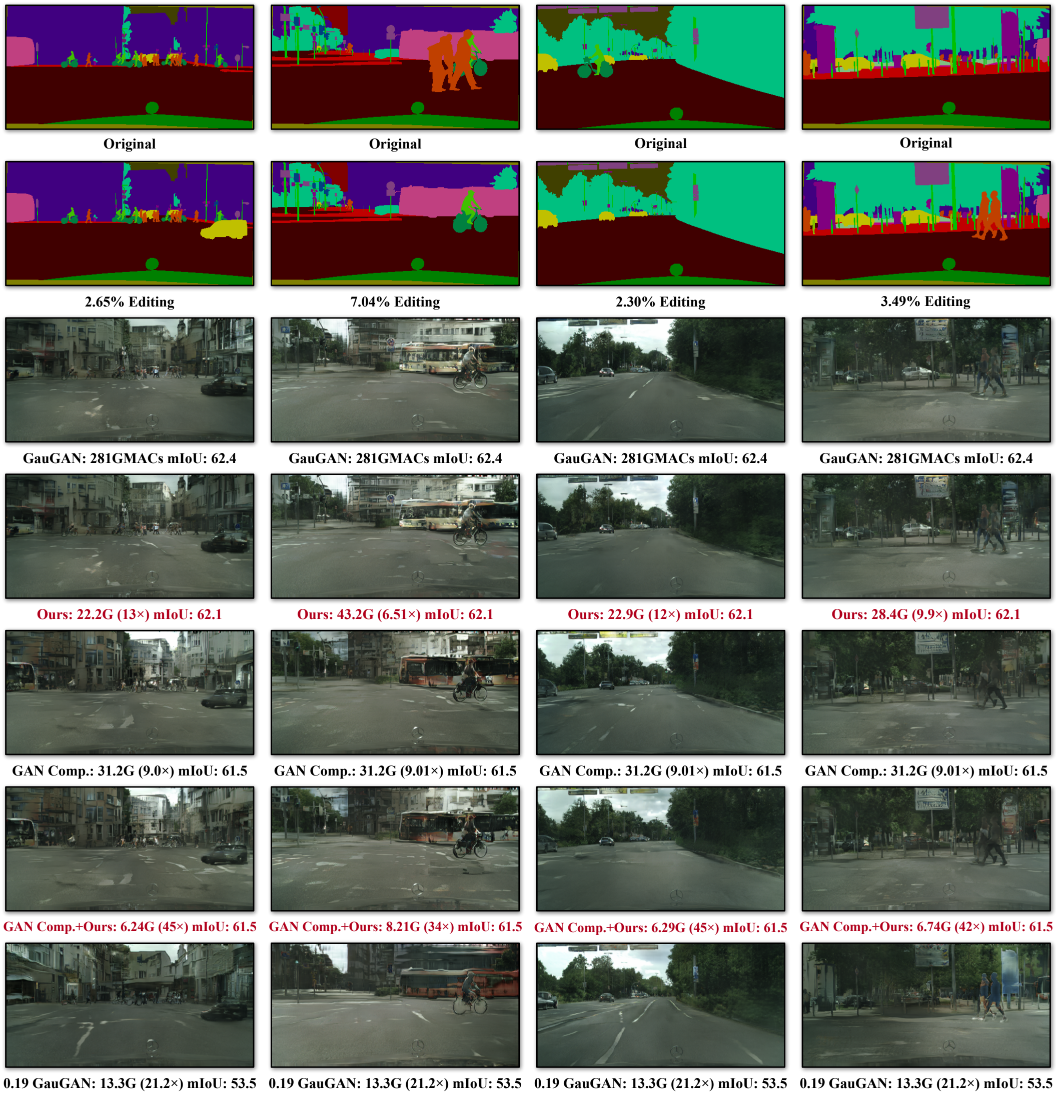


Figure 14: More visualization results on Cityscapes of GauGAN [2]. *0.19 GauGAN*: Uniformly reducing each layer of GauGAN to 19% channels and training from scratch. Our method could achieve higher mIoU than GAN Compression with less MACs. When applying to GAN Compression, our method achieves $34 \sim 45\times$ MACs reduction with minor mIoU drop.

[15] Shaojie Li, Mingbao Lin, Yan Wang, Chao Fei, Ling Shao, and Rongrong Ji. Learning efficient gans for image translation via differentiable masks and co-attention distillation. *IEEE Transactions on Multimedia*, 2022. 2, 3

[16] Qing Jin, Jian Ren, Oliver J Woodford, Jiazhuo Wang, Geng Yuan, Yanzhi Wang, and Sergey Tulyakov. Teachers do more than teach: Compressing image-to-image models. In *CVPR*, 2021. 2, 3

[17] Tamar Rott Shaham, Michaël Gharbi, Richard Zhang, Eli Shechtman, and Tomer Michaeli. Spatially-adaptive pixelwise networks for fast image translation. In *CVPR*, 2021. 2, 3

[18] Haotao Wang, Shupeng Gui, Haichuan Yang, Ji Liu, and Zhangyang Wang. Gan slimming: All-in-one gan compression by a unified optimization framework. In *ECCV*, 2020. 2, 3

[19] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 2

[20] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020. 2

[21] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019. 2

[22] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34, 2021. 2

[23] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2021. 2

[24] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. In *NeurIPS*, volume 32, 2019. 2

[25] Chitwan Saharia, William Chan, Huiwen Chang, Chris A Lee, Jonathan Ho, Tim Salimans, David J Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. *arXiv preprint arXiv:2111.05826*, 2021. 2

[26] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 2

[27] Peihao Zhu, Rameen Abdal, Yipeng Qin, and Peter Wonka. Sean: Image synthesis with semantic region-adaptive normalization. In *CVPR*, 2020. 2

[28] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021. 2

[29] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. Ilvr: Conditioning method for denoising diffusion probabilistic models. *arXiv preprint arXiv:2108.02938*, 2021. 2

[30] Gwanghyun Kim and Jong Chul Ye. Diffusionclip: Text-guided image manipulation using diffusion models. *arXiv preprint arXiv:2110.02711*, 2021. 2

[31] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *ECCV*, 2016. 2

[32] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *ICCV*, 2021. 2

[33] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *ICCV*, 2019. 2

[34] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? In *CVPR*, pages 8296–8305, 2020. 2

[35] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *ICCV*, 2019. 2

[36] 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. 2

[37] 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*, 2018. 2

[38] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 2, 6

[39] Zhifeng Kong and Wei Ping. On fast sampling of diffusion probabilistic models. *arXiv preprint arXiv:2106.00132*, 2021. 2

[40] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion GANs. In *ICLR*, 2022. 3

[41] Song Han, Huizi Mao, and William J Dally. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. In *ICLR*, 2016. 3

[42] Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. AMC: AutoML for Model Compression and Acceleration on Mobile Devices. In *ECCV*, 2018. 3

[43] Ji Lin, Yongming Rao, Jiwen Lu, and Jie Zhou. Runtime neural pruning. In *NeurIPS*, 2017. 3

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

[45] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *ICCV*, 2017. 3

[46] Zechun Liu, Haoyuan Mu, Xiangyu Zhang, Zichao Guo, Xin Yang, Kwang-Ting Cheng, and Jian Sun. MetaPruning: Meta Learning for Automatic Neural Network Channel Pruning. In *ICCV*, 2019. 3

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

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

[49] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. HAQ: Hardware-Aware Automated Quantization with Mixed Precision. In *CVPR*. 3

[50] 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. 3

[51] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *CVPR*, pages 2704–2713, 2018. 3

[52] Barret Zoph and Quoc V Le. Neural Architecture Search with Reinforcement Learning. In *ICLR*, 2017. 3

[53] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning Transferable Architectures for Scalable Image Recognition. In *CVPR*, 2018. 3

[54] Haoxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable Architecture Search. In *ICLR*, 2019. 3

[55] Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct Neural Architecture Search on Target Task and Hardware. In *ICLR*, 2019. 3

[56] 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 *CVPR*, 2019. 3

[57] 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 *CVPR*, 2019. 3

[58] Ji Lin, Wei-Ming Chen, Yujun Lin, John Cohn, Chuang Gan, and Song Han. Mcunet: Tiny deep learning on iot devices. In *NeurIPS*, 2020. 3

[59] Ji Lin, Richard Zhang, Frieder Ganz, Song Han, and Jun-Yan Zhu. Anycost gans for interactive image synthesis and editing. In *CVPR*, 2021. 3

[60] 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 *ICCV*, 2019. 3

[61] Yuchen Liu, Zhixin Shu, Yijun Li, Zhe Lin, Federico Perazzi, and Sun-Yuan Kung. Content-aware gan compression. In *CVPR*, 2021. 3

[62] Ruixin Ma and Junying Lou. Cpgan: An efficient architecture designing for text-to-image generative adversarial networks based on canonical polyadic decomposition. *Scientific Programming*, 2021. 3

[63] 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. 3

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

[65] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *ICLR*, 2016. 3

[66] Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, and Marianna Pensky. Sparse convolutional neural networks. In *CVPR*, 2015. 3

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

[68] Haotian Tang, Zhijian Liu, Xiuyu Li, Yujun Lin, and Song Han. Torchsparse: Efficient point cloud inference engine. In *MLSys*, 2022. 3

[69] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *CVPR*, 2017. 3

[70] Mengye Ren, Andrei Pokrovsky, Bin Yang, and Raquel Urtasun. Sbnet: Sparse blocks network for fast inference. In *CVPR*, 2018. 3, 5

[71] Patrick Judd, Alberto Delmas, Sayeh Sharify, and Andreas Moshovos. Cnvlutin2: Ineffectual-activation-and-weight-free deep neural network computing. *arXiv preprint arXiv:1705.00125*, 2017. 3

[72] Shaohuai Shi and Xiaowen Chu. Speeding up convolutional neural networks by exploiting the sparsity of rectifier units. *arXiv preprint arXiv:1704.07724*, 2017. 3

[73] Xuanyi Dong, Junshi Huang, Yi Yang, and Shuicheng Yan. More is less: A more complicated network with less inference complexity. In *CVPR*, 2017. 3

[74] Bowen Pan, Wuwei Lin, Xiaolin Fang, Chaoqin Huang, Bolei Zhou, and Cewu Lu. Recurrent residual module for fast inference in videos. In *CVPR*, 2018. 3

[75] Xiaoxiao Li, Ziwei Liu, Ping Luo, Chen Change Loy, and Xiaoou Tang. Not all pixels are equal: Difficulty-aware semantic segmentation via deep layer cascade. In *CVPR*, 2017. 3

[76] Thomas Verelst and Tinne Tuytelaars. Dynamic convolutions: Exploiting spatial sparsity for faster inference. In *CVPR*, 2020. 3

[77] Longguang Wang, Xiaoyu Dong, Yingqian Wang, Xinyi Ying, Zaiping Lin, Wei An, and Yulan Guo. Exploring sparsity in image super-resolution for efficient inference. In *CVPR*, 2021. 3

[78] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Yitian Zhang, and Haojun Jiang. Spatially adaptive feature refinement for efficient inference. *TIP*, 2021. 3

[79] Yulin Wang, Yang Yue, Yuanze Lin, Haojun Jiang, Zihang Lai, Victor Kulikov, Nikita Orlov, Humphrey Shi, and Gao Huang. Adafocus v2: End-to-end training of spatial dynamic networks for video recognition. *arXiv preprint arXiv:2112.14238*, 2021. 3

[80] Mathias Parger, Chengcheng Tang, Christopher D Twigg, Cem Keskin, Robert Wang, and Markus Steinberger. Deltacnn: End-to-end cnn inference of sparse frame differences in videos. *arXiv preprint arXiv:2203.03996*, 2022. 3

[81] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 5

[82] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 5, 1

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

[84] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, 2018. 5, 1

[85] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *ICML*, pages 8162–8171, 2021. 5

[86] Yaoyao Ding, Ligeng Zhu, Zhihao Jia, Gennady Pekhimenko, and Song Han. Ios: Inter-operator scheduler for cnn acceleration. *MLSys*, 2021. 5

[87] Zhihao Jia, Oded Padon, James Thomas, Todd Warszawski, Matei Zaharia, and Alex Aiken. Taso: optimizing deep learning computation with automatic generation of graph substitutions. In *SOSP*, 2019. 5

[88] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5, 1

[89] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015. 5

[90] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *arXiv preprint arXiv:2201.02605*, 2021. 6

[91] Shengyu Zhao, Jonathan Cui, Yilun Sheng, Yue Dong, Xiao Liang, Eric I Chang, and Yan Xu. Large scale image completion via co-modulated generative adversarial networks. In *ICLR*, 2021. 6

[92] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 6

[93] 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. *NeurIPS*, 2017. 6

[94] Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On aliased resizing and surprising subtleties in gan evaluation. In *CVPR*, 2022. 6

[95] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *CVPR*, pages 472–480, 2017. 6