DeepPrism: Channel Convolution for Lightweight Generative Models

Changqing Fu, Laurent D. Cohen

DeepPrism: Channel Convolution for Lightweight Generative Models

Changqing Fu
CEREMADE, University Paris Dauphine, PSL Research University, UMR CNRS 7534, Paris, 75016, France
cfu@ceremade.dauphine.fr

Laurent D. Cohen
CEREMADE, University Paris Dauphine, PSL Research University, UMR CNRS 7534, Paris, 75016, France
cohen@ceremade.dauphine.fr

ABSTRACT

In this paper, we introduce DeepPrism, a novel network architecture for generative models, which addresses the redundancy issue of convolutional filters. DeepPrism reaches an unprecedented parameter efficiency improvement of up to 1000 times on generative models. The number of parameters is reduced over conventional CNNs from quadratic to constant dependency with respect to the network's width. The training / inference time and space are also reduced. The main novelty lies in the geometric property namely the translation equivariance on channels, which gives rise to the convolution structure along the channels, and trivially generalizes to the attention mechanism. Compared with Latent Diffusion Model, DeepPrism produces similar qualitative and quantitative results, but the number of parameters can be reduced at a great scale. Other generative models including autoencoders and single-image diffusion models are also experimented to exhibit the generality of this method on generative models.

CCS CONCEPTS

• **Computing methodologies**; • **Image representations**;

KEYWORDS

Image Representations, Diffusion Models, Deep Learning

ACM Reference Format:
Changqing Fu and Laurent D. Cohen. 2023. DeepPrism: Channel Convolution for Lightweight Generative Models. In *2023 the 5th International Conference on Video, Signal and Image Processing (VSIP 2023), November 24–26, 2023, Harbin, China.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3638682.3638709

1 INTRODUCTION

Recent years have witnessed great developments of generative models in computer vision, where the output image is generated under multiple constraints as inputs. Many classical problems can be formulated into this framework. In image processing, examples are denoising, super-resolution, colorization, inpainting, etc., where the input constraint is a corrupted image. In computer vision, examples involve generation under label constraints (classes), spatial constraints (sketches, strokes), style constraints (of reference images), text constraints, etc. However, important issues remain unsolved along the way.

Efficient Generative Models. On one hand, digging out the power of small models is crucial and beneficial to boost the efficiency of large generative models. Modern learning systems are costly, since larger models tend to perform better. The training and inference burdens are both economically expensive and environmentally unsustainable. Even though large models can produce high-quality results, they are computationally inefficient since their marginal performance per parameter drops at scale. Recent works on efficient models like [15, 20, 27, 34, 45] have reduced the size of a classification network to several millions of parameters, but there are few similar work on generative models which achieve this level of efficiency within our scope. Our method even reaches an unprecedented parameter scale even compared to most small recognition models and keeps tremendous performance.

Geometric Deep Learning. On the other hand, new mathematical considerations of the convolution structure have always been a vital issue. Recent works in exploiting symmetry [3, 7, 14, 41] design more efficient network structures from a geometrical perspective. The term group equivariance of a layer means that layer and the group action (such as the translation operation) commutes, and the equivariance property leads to invariant quantities across hidden layers computed by Noether's theorem. Understanding it to find new forms of invariance remedies the redundancy problem as a benefit. Most existing approaches only consider the convolutional shift-invariance along the spatial dimensions, whereas our method study the shift-invariance of the channel domain of the hidden signal, which leads to explainable channel space in the light of this new type of convolution. Since the output depends on adjacent color channels, the proposed convolution can naturally be compared with chromatic dispersion.

Contribution. In this paper, we combine both above perspectives. In summary, we seek better parameter efficiency and more desirable geometrical property while largely preserving model performance. Our approach reduces the number of parameters of convolutional neural networks (CNNs), which results from an enforced weight-sharing technique. We find the usefulness of weight-sharing in 3 dimensions (width, height, channel), compared to conventional convolution in 2 dimensions (width and height only). Therefore we propose a convolution filter with parameters of as few as $3\times3\times3 = 27$ parameters each layer, and it works unreasonably well in generative models. Empirical result on reduction on parameter size with maintained generation performance is shown in figure 1.

VSIP 2023, November 24–26, 2023, Harbin, China

ACM ISBN 979-8-4007-0927-2/23/11
https://doi.org/10.1145/3638682.3638709

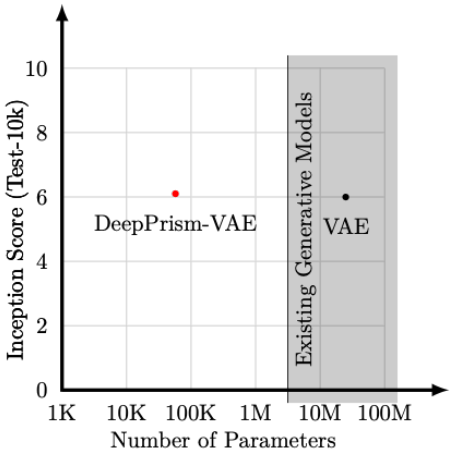

Figure 1: Generation quality of an Autoencoder versus number of parameters tested on CIFAR10 dataset.

2 RELATED WORKS

Generative models. Generative models have been developing at a great speed in the past few years, in terms of higher quality, wider diversity and more flexible controllability. For the quality aspect, both Generative Adversarial Networks (GAN) [13] and Diffusion Models (DM) [18, 37] can produce images of high resolution. In zero-shot settings with no training data (training only on the input), high quality images can also be generated using a CNN [35, 38]. For the diversity aspect, both GAN-based and DM-based models have the capacity to cover the patterns of the whole dataset out of a shared model. For the controllability side, input constraints can be fed into the model in the form of the concatenated feature maps [11, 28], learnable weights in the Adaptive Instance Normalization (AdaIN) [21] or weights in the cross-attention function [31].

Convolution and Attention-based Networks. The mainstream architecture for generative models is the CNN [25]. A basic network is composed of normalization, activation, convolution, spatial resampling (such as pooling and upscaling) layers, and so forth. Through this multi-layer process, the network might contain skip connections, such like residual networks [16]. The full convolution could be replaced by convolution with its variants such as group convolution [23]. Group convolution is one way to sparsify the convolution, which uses a block-diagonal matrix of G blocks instead of a full matrix along the channel dimensions. It reduces the network's parameter size by G folds. The dimensionality of a convolution filter is C^2K^2 where C is the network width and K is a small integer which implies the convolution kernel size. The attention layer, introduced in the Transformer architecture [39], is another successful network structures parallel to the convolution-activation layers, which has variants like multi-head attention [9], sparse attention [5], etc. The multi-head variant reduces the parameter sizes by considering a smaller spatial structure. The parameter sizes of attention layers are C^2. Practical choices of network width C is up to several hundred. Therefore the redundancy lies mostly in the channel dimensions in both cases, which motivates the design of lightweight structures.

Convolution Decompositions. The decomposition of convolution kernels leads to simplifications and reduction of redundancy. Understanding the spectral representation of convolution filters on a graph is the general case of this perspective [1, 4, 8]. Orthogonal convolution [40] can resolve redundancy and improve the performance of inference and generative models, but it neither changes the number of parameters nor reduces the computation overhead. Low-rank decomposition [43] is a way to compress the filter of channel dimensions C_1C_2 into the product of two matrices of dimension C_1r and rC_2 where r is a small integer. Practically speaking, it can achieve a compression rate of one order of magnitude while not losing too much performance for inference models.

Model Compression. Besides low-rank compression, there exists other ad-hoc methods to compress trained models. Pruning [2] is a way to wipe out parameters, which can also compress the model by an order of magnitude for inference models. There exist also other methods to reduce the computation of generative models. Distillation [29] is another method to use a trained model as the teacher (source of training data) for a (usually smaller) student model. Applying to diffusion models, it could accelerate the generation speed by one order of magnitude. What's more, quantization of floating-point numbers in the weights [30] and Neural Architecture Search [32] could also compress the parameters by one order of magnitude. Our method produces a much better parameter-efficient result.

3 METHOD: DEEPPRISM

3.1 Motivation

Our interpretation of the channel space is motivated by the optic dispersion phenomenon. A natural light is composed of "lasers" (pure-color rays) of different frequencies. When a beam of white light passes through a prism, the resulting light is split into a "rainbow", since different color has different refraction angles. The arrangement of pure colors on the rainbow is monotone according to the light frequency. In this way, the mixed color beam is disentangled by the prism, and we propose a physical interpretation of the channel dimension as an analogy of the light spectrum, illustrated in figure 2. The frequency values form the space of light spectrum, which is a real line or 1D Euclidian space. Besides the light frequency, other color spaces also motivate this understanding. Under the scenario of biological vision, the visible spectrum is a finite interval with minimum and maximum frequencies. Under the scenario of digital representation of colors using HSV space, the hue takes the role of the light frequency, taking value on a circle $[0, 2\pi]$, as is summarized in table 1.

3.2 Topology on Channel

Our geometrical understanding of the channel dimension originates from a signal processing perspective. We regard the channel index as a discrete sampling from a continuous space. On this space we endow a topological structure. Note that the term topology in our context refer to the mathematical tool of a standard classification of spaces. The core idea is the continuity concept: two spaces are the same class if they are continuously mapped to each other with an invertible function. To characterize continuity, the basic building block is the definition of "neighborhoods", or open sets. In short,

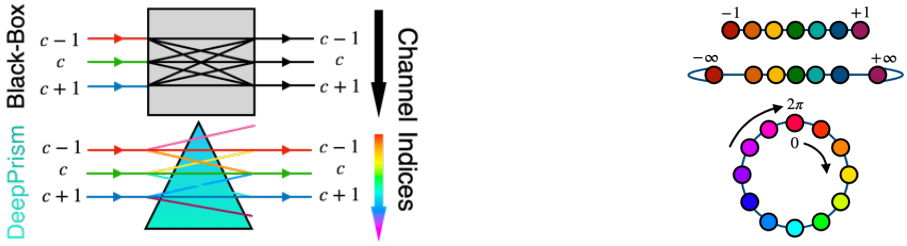

Figure 2: Top Left: "black-box" fully connected layer. Bottom Left: light dispersion through a prism. Right: Different types of topologies to model the light spectrum. From top to bottom: interval, line, circle.

Table 1: Topology of different padding mode of channel convolutions, with relations to the the boundary types of the filtering process, and their motivation.

Padding	Zero	Replicate	Circular
Parameter	Interval	Line	Circle
Training	Direchlet	Neumann	Periodic

neighborhoods are continuously mapped from neighborhoods. On the spatial dimensions of an image, this motivates convolution. The neighborhoods correspond to image patches, defined by adjacent pixels. The discrete counterpart of a topological space is a discrete graph, where the neighborhood of a node is defined by those nodes connected by edges. Under the perspective of graph neural networks, we can define the topology of a convolution layer. Take a 3-by-3 convolution layer as an example. The pixel grid is a graph whose nodes are the homogeneous 2D lattice, and the edge is the adjacency of pixels.

Our key insight is that in the channel dimension, neighborhoods can also be characterized by the adjacency of channel indices. As a result, the channel indices now obtain their unique topological structure (in the circular case, the uniqueness is valid modulo channel rotation). Learning this type of convolution on the database automatically grasp features which are shift-invariant along the channel dimension.

3.3 Channel Convolution

Given the input feature maps u with shape [B, C, H, W], with batch size B, channel size C, spatial resolution H × W, and the convolutional filter with shape [K], where K is the kernel size, The channel convolution on an image is defined as

$$Prism\,(u; w) = Conv1D(w, u^{\top})^{\top} \tag{1}$$

where Conv1D stands for the 1D per-channel convolution on the last dimension (or number of groups equals channel size), $\top$ is the transpose operator between spatial dimensions and the channel dimension. We only show the astonishing efficiency of the case when K is as small as 3.

Different padding modes of the 1D convolution corresponds to different channel topology, as is illustrated in table 1. Zero padding corresponds to the image signal with channel values fixed to zero at both ends, or at minimum and maximum light frequency. The fixed terminal value of a continuous signal is referred to as the Dirichlet boundary. Correspondingly, the replicative padding mode characterizes the free boundary or Neumann boundary, whereas the circular padding mode corresponds to the periodic boundary. These three cases correspond to the three cases discussed section 3.1.

3.4 Separable Convolution

Separable convolutions [6] is a combination of channel-wise and space-wise convolutions. It gives rise to an alternating convolution on both the space and channel directions. For a channel filter w_c with shape [K], with K being the kernel size, and a spatial filter of shape w_s = [K, K], the filter w = $\{w_c, w_s\}$ parameterizes a separable convolution:

$$PrismConv\,(u; w) = Conv1D(w_c, Conv2D(w_s, u)^{\top})^{\top} \tag{2}$$

This pseudo-3D convolution is successfully applied in video generation architectures [12, 19, 36]. The difference is that in our case, Conv2D is also a per-channel convolution, to produce an even more lightweight design for image generation. That is, taking K as 3, the w_s takes the form of a 3 × 3 matrix, and the whole convolution w is a tensor direct sum of the spatial filter w_s and channel filter w_c, or $(w_{ij} + w'_h)_{i,j=1,2,3}$. Denoting this 3D filter as w, we obtain an alternative form of the convolutionn, the extension is deduced as $PrismAttn(u; w_q, w_k, w_v, w_o) = Prism(Attn(q, k, v); w_o)$

This structure is not capable of representing the correlation between the space and channel dimensions.

3.5 DeepPrism

We propose to use instead a relaxed full 3D tensor as the convolution kernel (w_{hij}) where h,i,j=1,2,3, h is the channel dimension and i, j are the spatial dimensions. Therefore, the proposed DeepPrism convolution is defined as

$$DeepPrism\,(u; w) = Conv3D\,(w, u) \tag{3}$$

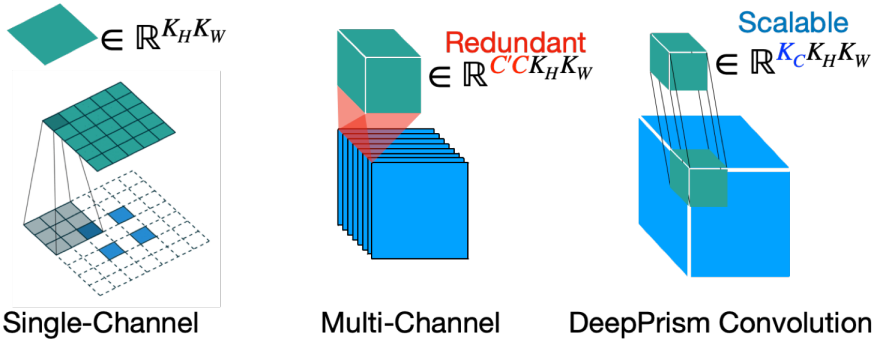

Figure 3: Illustration of Single-Channel, Multi-Channel Convolution and DeepPrism. The green cube is the convolution kernel whereas the blue cube is the neuron signal or the hidden layers of the neural network. The dimensionality of DeepPrism's convolution kernel greatly reduces from $C^{'}CK_HC_W$ of a common convolution kernel to $K_CK_HK_W$, where $K_C \ll C$ and we take $K_C = K_H = K_W = 3$.

The whole architecture of the DeepPrism backbone is therefore replacing every convolution layer except the input and output convolution layers. Regarding the two layers as demodulation and modulation between RGB colors and C-channel features, the rest of the network is a channel-dependent convolution on the feature signal over a fixed 3D field. A DeepPrism convolution is defined as single-channel 3D convolution, by regarding the input layer as a 3D signal with dimensions of height, width and channel size, as is illustrated in figure 3.

A concise pytorch implementation of the 3×3×3 `DeepPrism` function is the following.

```
# x, y: Layer input and output
# x and y's size: (B,C,H,W)
class DeepPrism(torch.nn.Conv3d):
    def __init__(self,**kwargs):
        super(DeepPrism, self).__init__(in_channels=1,out_channels
        =1, **kwargs)
    def forward(self,input):
        return super(DeepPrism, self).forward(input.unsqueeze(1)).
        squeeze(1)
layer = DeepPrism(kernel_size=(3,3,3),padding=1)
y = layer(x)
```

Extension: PrismAttn. Especially for the attention layer, suppose q, k, v stand for the query, key and value, which are the output of the C-channel convolution with kernel size 1×1, the attention function is defined as

$$Attn(q,k,v)_{hij} = \sum_{h',i',j'=1}^{C,H,W} Softmax\left(\frac{q_{h'ij}k_{h'i'j'}}{\sqrt{C}}\right) v_{hi'j'} \tag{4}$$

The attention block is a ResNet layer whose residual is the attention function convolved by a 1×1 convolution filter, Conv2D(wo, Attn(q, k, v)). Suppose the convolution kernels are w_q, w_k, w_v, w_o with size [K], the Prism Attention is therefore defined as

$$PrismAttn\left(u; w_q, w_k, w_v, w_o\right) = Prism\left(Attn\left(q,k,v\right); w_o\right) \tag{5}$$

with the query, value and key being q = Prism(u; w_q), k = Prism(u; w_k) and v = Prism(u; w_v)).

4 EXPERIMENTS

DeepPrism replaces all the hidden 3D convolution layers of width C with DeepPrism in the baseline model, and saves the number of parameters by at least two to three orders of magnitude. The complexity of the proposed method improved over conventional convolutions is analyzed in table 2, whereas the practical number of parameters is compared in table 3.

4.1 Autoencoder

The first task we experiment on is the autoencoder. Its goal is to compress an image into a latent code of smaller dimensions.

Results. The compression rates compared with the baseline is summarized in table 3. The quantitative result of the image reconstruction evaluated by Inception Score was shown in figure 1. It indicates that our model surpasses state-of-the-art generative models by a large margin in terms of parameter size. Detailed quantitative evaluations are detailed in table 4. Visualizations of the reconstructions are illustrated in figure 4. Note that other lightweight convolution methods never reach such a small scale on parameter sizes as is mentioned in the related work section, and thus they are not comparable to our approach.

Baseline. The baseline model follows the configurations in [33]. The network is composed of four parts: the encoder E, the decoder G, the discriminator D and a pre-trained classification network V. Given an input x, the latent code is encoded with the encoder E, and we assume that the latent code follows a Gaussian distribution z ∼ N (μz, σz). The mean and variance of z are predicted by the encoder μz = E(x), σz = E'(x), where E, E' share the same network except for the output convolution layer. The output is then decoded as y = G(z).

Loss Function. The training objective contains four parts: the reconstruction term, the perceptual term, the entropy term and the adversarial term. The loss and regularization terms are explained as follows.

The reconstruction term is the per-pixel loss for rough reconstruction.

$$\| x - y\|^2 = \frac{1}{CHW} \sum_{h,i,j=1}^{C,H,W} |x_{hij} - y_{hij}|^2 \tag{6}$$

Table 2: Complexity of different types of convolution

Operator	Conv [a]	Group	Prism	Attention	FlashAttn [b]	MultiHeadAttn	SparseAttn	PrismAttn
Parameter	C^2	C^2/G	1	C^2	C^2	C^2	C^2	1
Training	BC^2N	BC^2N/G	BCN	$B(CN^2+C^2N)$	$B(CN^2+C^2N)$	$B(CN^2/P+C^2N)$	$B(CN^{3/2}+C^2N)$	$B(N^2+CN)$
Space	BCN	BCN	BCN	$B(N^2+CN)$	BCN	$B(N^2/P+CN)$	$B(CN^{3/2}+CN)$	$B(N^2+CN)$
Inference	C^2N	C^2N/G	CN	CN^2+C^2N	CN^2+C^2N	$CN^2/P+C^2N$	$CN^{3/2}+C^2N$	CN^2

[a] The time complexity of the convolution function could be further optimized through Fast Fourier Transform [17].

[b] The improvement of space complexity of FlashAttention [46] or other methods over vanilla Attention could be combined.

Table 3: Number of parameters of DeepPrism network vs baseline models for image compression with variational autoencoder. Our method gives similar results but is up to thousands of times smaller.

Width C	VAE (LDM)	VAE (DeepPrism)	Boost
128	6.36M	29.5K	216 ×
256	25.3M	57.8K	438 ×
512	101M	114K	886 ×
1024	405M	227K	1784 ×

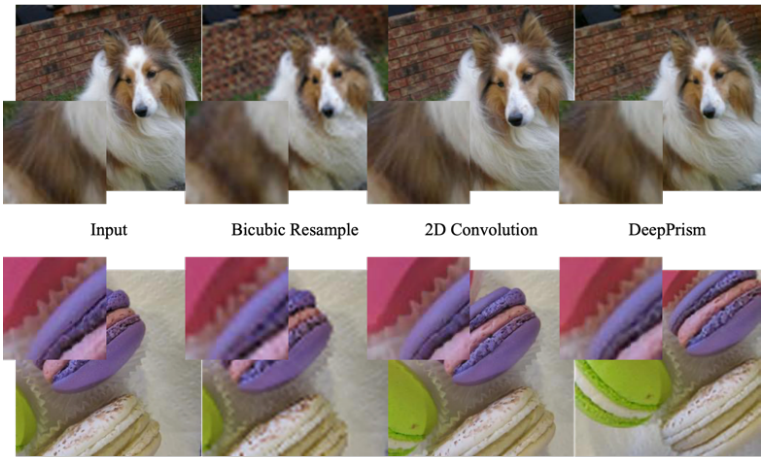

Figure 4: Result on the Imagenet dataset.

Table 4: Quantitative results of the DeepPrism autoencoder on CIFAR10 dataset.

Method	PSNR	IS	KID	PS	SSIM	Parameter
Baseline	18.8 ± 2.4	5.99 ± 0.12	0.0146 ± 0.0011	2.07 ± 0.49	0.609 ± 0.106	25.3M
DeepPrism	**24.1 ± 2.5**	**6.08 ± 0.15**	**0.0179 ± 0.0011**	**1.03 ± 0.27**	**0.850 ± 0.053**	**57.8K**

Using this loss alone is desired when one wishes the network to match the target only. Otherwise, regularization terms help to avoid over-fitting.

- The perceptual term [44] is to take the hidden layers along the forward pass of V , with both the generation and the ground truth as inputs, and then compute a weighted sum of the Euclidian distance between them across layers. The pre trained V 's weights are frozen during the training, except that the coefficients of the weighted layer-wise sum w_ℓ for layer ℓ are learnable. This distance matches well the similarity of two sets of images according to human perception. Let ℓ be the first few layers, the perceptual loss is written as

$$\frac{1}{2}\sum_{\ell} w_\ell \parallel V_\ell(x) - V_\ell(y) \parallel^2 \tag{7}$$

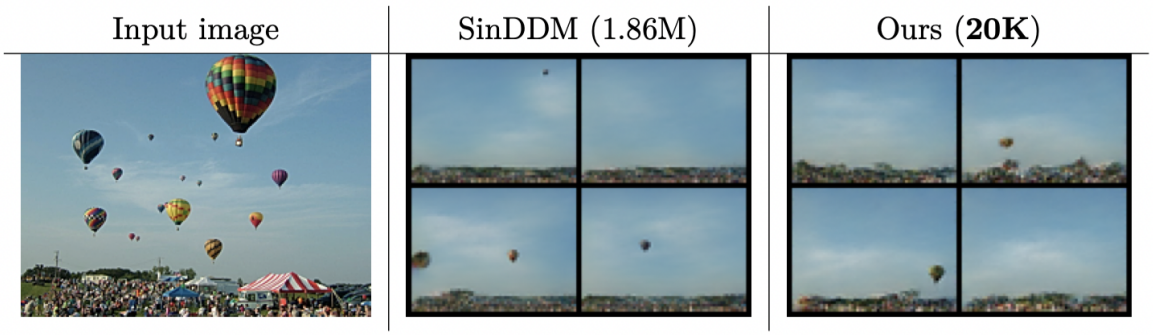

Figure 5: Results on SinDDM

- The entropy term [22] is to encourage the code z to be close to a white noise by minimizing the relative entropy (Kullback-Leibler divergence) of z's distribution from a standard Gaussian distribution. Given two distributions $\mu 1$, $\mu 2$ and $\sigma 1$, $\sigma 2$ which takes shape of [C, H, W], the KL divergence is given by

$$H\left(\mathcal{N}\left(\mu_1, \sigma_1^2 \boldsymbol{I}\right) | \mathcal{N}\left(\mu_2, \sigma_2^2 \boldsymbol{I}\right)\right) = \frac{1}{2} \sum_{h,i,j=1}^{C,H,W} \left(\frac{(\mu_1-\mu_2)^2}{\sigma_2} + \frac{\sigma_1}{\sigma_2} - log\sigma_1 + log\sigma_2 - 1\right) \tag{8}$$

- The GAN term [26] maximizes the distance from an optimal SVM separating hyperplane estimated by the discriminator D. Since the GAN convergence is not stable, we follow [33] to postpone the introduction of this regularization term after the training stabilizes.

The overall optimization goal is therefore

$$L\left(E, G, D, w\right) = \min_{E,G,Z} \max_{D} \mathbb{E}_u \begin{bmatrix} \frac{1}{2}\|u - G\left(E\left(u\right)\right)\|^2 + \\ \frac{\lambda_1}{2} \sum_{\ell} w_\ell \|V_\ell\left(u\right) - V_\ell\left(G\left(E\left(u\right)\right)\right)\|^2 + \\ \lambda_2 H(\mathcal{N}(E\left(u\right), E'\left(u\right)^2 \boldsymbol{I}) | \mathcal{N}\left(0, \boldsymbol{I}\right)) + \\ \lambda_3 (1 + D\left(u\right))_+ + (1 - D\left(G\left(E\left(u\right)\right)\right))_+ \end{bmatrix} \tag{9}$$

Network Structure. For simplicity and to avoid channel space re-sampling for our DeepPrism function, we fix the channel size C = 256. Note that channel-resampling for DeepPrism is feasible, but we hold the channel size as constant for the sake of simplicity and to control variable for the ablation. The architecture of the Encoder and Decoder are as follows:

E : Conv-3*(2*Res-Down)-Res-Attn-Res-Conv

G : Conv-Res-Attn-Res-3*(2*Res-Up)-Conv

In the above network specification, $*$ denotes repetitive layers of the same form (with different parameters), Conv is the 2D convolution with C-channel input and output, 3×3 convolution kernel and zero padding, Down and Up are 2x spatial downscaling and upscaling, Attn denotes the attention block, Res denotes a ResNet block

Res(x) = x +2*(Norm-Act-Conv) (x)

For normalization Norm, we use adaptive group normalization [42] with 32 groups, for the activation function, we adapt to the baseline method with Sigmoid-weighted Linear Unit (SiLU) [10].

Training Details. We first validate DeepPrism on the CIFAR10 dataset. The dataset is composed of 50K training images and 10K validation images of resolution 32×32. We use the Adam optimizer and tune the base learning rate to 10−7, then we scale up the learning rate by the batch size. We take a batch size of B = 2048 (distributed on multiple GPUs, and separated into 2 steps by gradient accumulation), set the perceptual loss weight $\lambda 1 = 1$, entropy loss weight $\lambda 2$ = 1e−6, and adversarial loss weight $\lambda 3$ =1/2. We used neither learning rate scheduler nor dropout since the learning converges without overfitting. We train the autoencoder on 8 NVIDIA A100 GPUs for 150 epochs or until the loss does not drop anymore.

Then we explore the performance of our DeepPrism network on high-resolution Imagenet dataset. The dataset contains 1281K training images and 50K validation images resized to 256×256. We keep the same network structure as above, and tune the learning rate to 4.5e−6 for standard convolution and 2e−5 for DeepPrism.

4.2 Single-Image Denoising Diffusion

Next, we experiment our DeepPrism network on diffusion model trained on a single image. The network architecture is G = 4*(Conv-DeepPrism-GeLU-DeepPrism)

All the network structure and training settings follow [24]. We only replace the 2D convolution with the proposed DeepPrism convolution. The qualitative results of our method in figure 5 is comparable with the baseline but is 93 times lighter in parameter size.

4.3 Diffusion Models

Finally, we explore the capacity of our model in the high-fidelity diverse generation task, using the Latent Diffusion Model (LDM) [33]. The training details are the same in the baseline model, and we trained both models for ∼200 epochs. For the network structure, we keep all the settings in the baseline except that we fix the network width to C = 256. The result is shown in figure 6. Our model achieves high performance, visually comparable to the baseline model, and the parameter size shrinks to ∼500 times smaller compared to the baseline model of 37.9M parameters, without counting the time embedding layers nor replacing the 2D convolution since it's a minor structure.

Figure 6: Results on LDM, where DeepPrism achieves similar results with a ∼500x smaller model.

5 CONCLUSION

In this work, we have proposed a lightweight convolution structure for neural generative models. It's the very first successful generative architecture reaching such a small parameter size. The implications of DeepPrism also include a geometrically explainable channel space, originating from translation equivariance, to enlarge the convolutional weight sharing to the channel dimension.

ACKNOWLEDGMENTS

This work was granted access to the HPC resources of IDRIS under the allocation 2022-AD011013178 made by GENCI, and supported by cloud TPU from Google's TPU Research Cloud (TRC). We thank Jamal Atif, Gabriel Peyré and anonymous reviewers for fruitful discussions and helpful comments.

REFERENCES

[1] Yonathan Aflalo, Haim Brezis, and Ron Kimmel. "On the optimality of shape and data representation in the spectral domain". In: SIAM Journal on Imaging Sciences 8.2 (2015), pp. 1141–1160.
[2] Davis Blalock *et al.* "What is the state of neural network pruning?" In: Proceedings of machine learning and systems 2 (2020), pp. 129–146.
[3] Joan Bruna and Stéphane Mallat. "Invariant scattering convolution networks". In: IEEE transactions on pattern analysis and machine intelligence 35.8 (2013), pp. 1872–1886.
[4] Benjamin Chamberlain *et al.* "Beltrami flow and neural diffusion on graphs". In: Advances in Neural Information Processing Systems 34 (2021), pp. 1594–1609.
[5] Rewon Child *et al.* "Generating long sequences with sparse transformers". In: arXiv preprint arXiv:1904.10509 (2019).
[6] François Chollet. "Xception: Deep learning with depthwise separable convolutions". In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017, pp. 1251–1258.
[7] Taco S Cohen and Max Welling. "Group equivariant convolutional networks". In: International conference on machine learning. PMLR. 2016, pp. 2990–2999.
[8] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. "Convolutional neural networks on graphs with fast localized spectral filtering". In: Advances in neural information processing systems 29 (2016).
[9] Alexey Dosovitskiy *et al.* "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: International Conference on Learning Represen tations.
[10] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning". In: Neural Networks 107 (2018), pp. 3–11.
[11] Patrick Esser, Robin Rombach, and Bjorn Ommer. "Taming transformers for high-resolution image synthesis". In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021, pp. 12873–12883.
[12] Patrick Esser *et al.* "Structure and Content-Guided Video Synthesis with Diffusion Models". In: arXiv preprint arXiv:2302.03011 (2023).
[13] Ian Goodfellow *et al.* "Generative adversarial networks". In: Communications of the ACM 63.11 (2020), pp. 139–144.
[14] Simon Graham, David Epstein, and Nasir Rajpoot. "Dense steerable filter cnns for exploiting rotational symmetry in histology images". In: IEEE Transactions on Medical Imaging 39.12 (2020), pp. 4124–4136.
[15] Dongyoon Han *et al.* "Rethinking channel dimensions for efficient model design". In: Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition. 2021, pp. 732–741.
[16] Kaiming He *et al.* "Deep residual learning for image recognition". In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016, pp. 770– 778.
[17] Tyler Highlander and Andres Rodriguez. "Very efficient training of convolutional neural networks using fast fourier transform and overlap-and-add". In: arXiv preprint arXiv:1601.06815 (2016).
[18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In: Advances in Neural Information Processing Systems 33 (2020), pp. 6840– 6851.
[19] Jonathan Ho *et al.* "Video diffusion models". In: arXiv preprint arXiv:2204.03458 (2022).
[20] Andrew G Howard *et al.* "Mobilenets: Efficient convolutional neural networks for mobile vision applications". In: arXiv preprint arXiv:1704.04861 (2017).
[21] Tero Karras, Samuli Laine, and Timo Aila. "A style-based generator architecture for generative adversarial networks". In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019, pp. 4401–4410.
[22] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: arXiv preprint arXiv:1312.6114 (2013).
[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: Communications of the ACM 60.6 (2017), pp. 84–90.
[24] Vladimir Kulikov *et al.* "SinDDM: A Single Image Denoising Diffusion Model". In: arXiv preprint arXiv:2211.16582 (2022).
[25] Yann LeCun *et al.* "Handwritten digit recognition with a back-propagation network". In: Advances in neural information processing systems 2 (1989).
[26] Jae Hyun Lim and Jong Chul Ye. "Geometric gan". In: arXiv preprint arXiv: 1705.02894 (2017).
[27] Sachin Mehta and Mohammad Rastegari. "MobileViT: Light-weight, General purpose, and Mobile-friendly Vision Transformer". In: International Conference on Learning Representations.
[28] Chenlin Meng *et al.* "Sdedit: Guided image synthesis and editing with stochastic differential equations". In: International Conference on Learning Representations. 2021.
[29] Chenlin Meng *et al.* "On distillation of guided diffusion models". In: arXiv preprint arXiv:2210.03142 (2022).
[30] Markus Nagel *et al.* "A white paper on neural network quantization". In: arXiv preprint arXiv:2106.08295 (2021).
[31] Alexander Quinn Nichol *et al.* "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models". In: International Conference on Machine Learning. PMLR. 2022, pp. 16784–16804.
[32] Pengzhen Ren *et al.* "A comprehensive survey of neural architecture search: Chal lenges and solutions". In: ACM Computing Surveys (CSUR) 54.4 (2021), pp. 1–34.
[33] Robin Rombach *et al.* "High-resolution image synthesis with latent diffusion mod els". In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pat tern Recognition. 2022, pp. 10684–10695.
[34] Mark Sandler *et al.* "Mobilenetv2: Inverted residuals and linear bottlenecks". In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018, pp. 4510–4520.
[35] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. "Singan: Learning a gen erative model from a single natural image". In: Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019, pp. 4570–4580.
[36] Uriel Singer *et al.* "Make-a-video: Text-to-video generation without text-video data". In: arXiv preprint arXiv:2209.14792 (2022).
[37] Jascha Sohl-Dickstein *et al.* "Deep Unsupervised Learning using Nonequilibrium Thermodynamics". In: Proceedings of the 32nd International Conference on Ma chine Learning. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Ma chine Learning Research. Lille, France: PMLR, 2015, pp. 2256–2265. url: https: //proceedings.mlr.press/v37/sohl-dickstein15.html.
[38] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. "Deep image prior". In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018, pp. 9446–9454.
[39] Ashish Vaswani *et al.* "Attention is all you need". In: Advances in neural information processing systems 30 (2017).
[40] Jiayun Wang *et al.* "Orthogonal convolutional neural networks". In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020, pp. 11505–11515.
[41] Maurice Weiler, Fred A Hamprecht, and Martin Storath. "Learning steerable fil ters for rotation equivariant cnns". In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018, pp. 849–858.
[42] Yuxin Wu and Kaiming He. "Group normalization". In: Proceedings of the Euro pean conference on computer vision (ECCV). 2018, pp. 3–19.
[43] Xiyu Yu *et al.* "On compressing deep models by low rank and sparse decomposition". In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2017, pp. 7370–7379.

[44] Richard Zhang *et al.* "The unreasonable effectiveness of deep features as a per ceptual metric". In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018, pp. 586–595.
[45] Xiangyu Zhang *et al.* "Shufflenet: An extremely efficient convolutional neural network for mobile devices". In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2018, pp. 6848–6856.
[46] Dao, T., Fu, D., Ermon, S., Rudra, A. and Ré, C., 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. Advances in Neural Information Processing Systems, 35, pp.16344-16359.