TinyLUT: Tiny Look-Up Table for Efficient Image Restoration at the Edge

Huanan Li[1,2], **Juntao Guan**[1,2,3], **Rui Lai**[1,2*], **Sijun Ma**[1,2], **Lin Gu**[4,5*], **Zhangming Zhu**[1,2]
[1]Key Laboratory of Analog Integrated Circuits and Systems (Ministry of Education)
[2]School of Integrated Circuits, Xidian University, Xi'an, China
[3]Hangzhou Institute of Technology, Xidian University, Hangzhou, China
[4]RIKEN AIP, Tokyo103-0027, Japan
[5]The University of Tokyo, Japan
{huananli,sijunma}@stu.xidian.edu.cn
{guanjuntao,zhangmingzhu}@xidian.edu.cn
rlai@mail.xidian.edu.cn, lin.gu@riken.jp

Abstract

Look-up tables(LUTs)-based methods have recently shown enormous potential in image restoration tasks, which are capable of significantly accelerating the inference. However, the size of LUT exhibits exponential growth with the convolution kernel size, creating a storage bottleneck for its broader application on edge devices. Here, we address the storage explosion challenge to promote the capacity of mapping the complex CNN models by LUT. We introduce an innovative separable mapping strategy to achieve over $7\times$ storage reduction, transforming the storage from exponential dependence on kernel size to a linear relationship. Moreover, we design a dynamic discretization mechanism to decompose the activation and compress the quantization scale that further shrinks the LUT storage by $4.48\times$. As a result, the storage requirement of our proposed TinyLUT is around 4.1% of MuLUT-SDY-X2 and amenable to on-chip cache, yielding competitive accuracy with over $5\times$ lower inference latency on Raspberry 4B than FSRCNN. Our proposed TinyLUT enables superior inference speed on edge devices with new state-of-the-art accuracy on both of image super-resolution and denoising, showcasing the potential of applying this method to various image restoration tasks at the edge. The codes are available at: https://github.com/Jonas-KD/TinyLUT.

1 Introduction

With image sensors widely used in IoT applications, the demand for image restoration on edge devices has been stimulated. For decades, classic methods including interpolation [1, 2], sparse coding [3, 4, 5] and domain transformation [6] have been proposed. In recent years, deep learning based algorithms [7, 8, 9, 10, 11] have made encouraging progress in restoration quality, while suffering from high computational load and long inference latency. These issues limit their extensive applications in resource-constrained IoT terminals. Thus it can be seen, the growing demand for edge devices (*e.g.* smartphones and Raspberry Pi) calls for an extremely lightweight solution for image restoration with high quality. Given this, researchers develop LUT-based methods [12, 13, 14] to supplant the intensive convolutional computations with direct memory access operation for accelerating the inference, and exhibiting enormous potential.

In recent research, SRLUT [15] initially transfers the output values of a small trained deep network with receptive field (RF) size 4 to uniformly sampled LUT. To further improve the super resolution

*Corresponding author

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

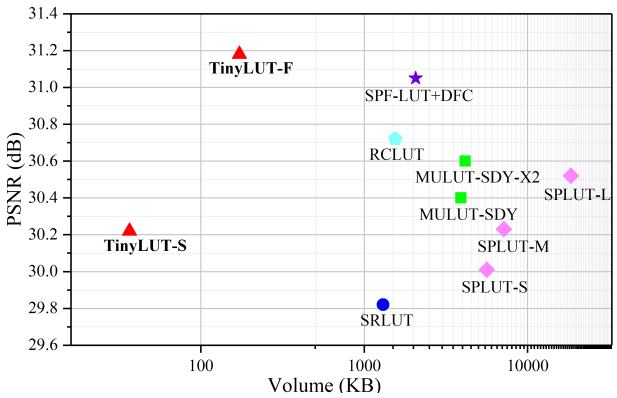


Figure 1: Performance comparison of our TinyLUT and other state-of-the art LUT-based algorithms. PSNR vs. Volume on Set5 benchmark dataset for $\times 4$ super resolution task. TinyLUT outperforms the competitors in accuracy with over $10\times$ lower memory consumption.

Table 1: LUT size estimation when storing 8bit data with output entities $r = 4$. Compared with the full size, the sampled operation effectively reduces the storage consumption of LUT. Furthermore, separable mapping strategy(SMS) and trained dynamic discretization mechanism (DDM) achieve an exaggerated reduction result.

RF	LUT	Full size	SRLUT	SMS	SMS + DDM
1×1 pixels	1D	4KB	272B	4KB	0.89KB
2×2 pixels	4D	64GB	1.27MB	16KB	3.57KB
3×3 pixels	9D	64ZB	1767GB	36KB	8.04KB

quality, Li et al. [14] increases the RF size through the different kernel shape combination with multiple LUTs. Meanwhile, Ma et al. [13] constructs a series-parallel LUT framework based on multiple LUTs cascade to improve the prediction accuracy. However, the existing LUT-based schemes face the challenge of storage explosion when mapping 8bit image with larger than 2×2 kernel size. The storage explosion invalidates previous methods where the required storage of LUT grows exponentially as the number of indexing entries (*i.e.* input entities) increase. As reported in Table 1, 3×3 kernel, commonly used in CNNs, will need a tremendous storage up to $(2^8)^9 \times 8\text{bit} \times 16 = 64\text{ZB}$ and $(2^4 + 1)^9 \times 8\text{bit} \times 16 = 1767\text{GB}$ in full LUT and uniformly sampled LUT [15] with 8bit data respectively. Therefore, the issue of storage explosion presents severe challenge when mapping the complex models by LUT on edge devices.

To overcome the storage explosion challenge, we propose the separable mapping strategy(SMS) and dynamic discretization mechanism (DDM) to decouple the kernel and activation, respectively. As reported in Table 1, the reduction of LUT storage reaches exponential levels using SMS and further achieves a $4.48\times$ times reduction by DDM when the RF size is 3×3 and storing 8bit data.

From the perspective of kernel decomposition, considering the exponential relationship between the dimension of input entries and LUT size, we propose the innovative SMS to decouple the convolution kernel by spatial location relationship. SMS decomposes the convolution weight of n input entities into the parallelization of n component independent sub-operations, which are instantiated as 1×1 kernel size respectively to reduce the number of input entries in LUT-based inference, consequently reducing the dimension of nD LUT. Then according to the additivity of convolutions with compatible kernel sizes[16], we reconstruct the feature space by averaging each output of n parallel layers. Finally, each branch with 1×1 kernel are individually mapped by 1-dimension LUT (1D LUT) with 1 input entity for s possible values. This approach reduces the size of nD LUT from s^n to $s \times n$ and allows vanilla convolution to be LUT-mapped on resource-constrained devices.

As for the activation compression, previous methods [15, 14, 17] uniformly sample the input values to reduce the possible values of s and apply interpolation to recover the inference accuracy. We analyze the advantages of SPLUT [13] and propose dynamic discretization mechanism (DDM) to decompose the activation into Most Significant Bits (MSBs) and Least Significant Bits (LSBs). Moreover, inspired by the activation quantization methods[18, 19], DDM introduces the learnable clipping parameters to explicitly compress the quantization scale of MSBs and LSBs activation, thereby reducing the possible pixels values and decreasing LUT scales.

As illustrated in Fig 1, we take the classic image restoration task single image super resolution (SISR) as an example. Our proposed TinyLUT-F consumes only 4.1% storage of MuLUT-SDY-X2[14] to achieve over 0.58dB PSNR increase in Set5 testset. Compared to other LUT-based methods, TinyLUT significantly reduces storage overhead while improving accuracy, expanding the application of LUT-based approaches on edge devices.

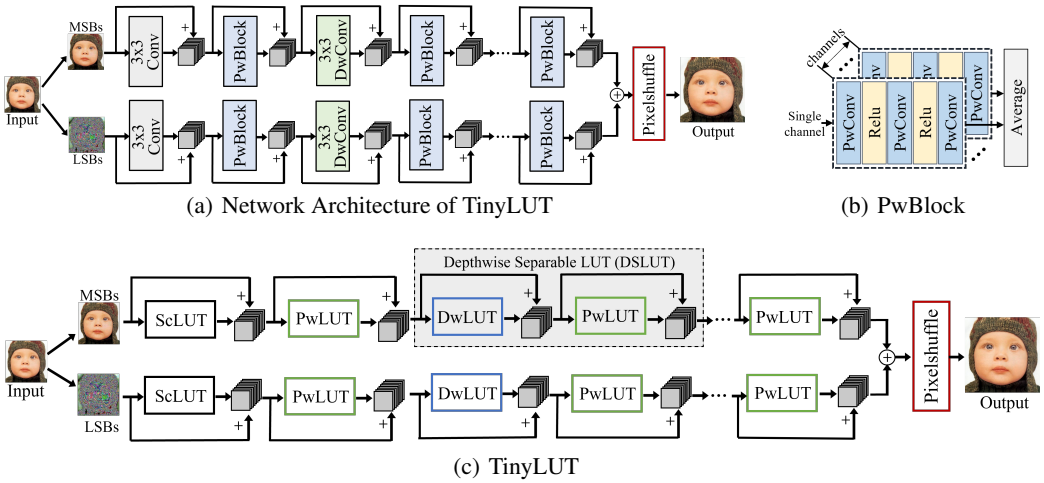
The main contributions can be summarized as follows:

- We reveal the scheme of storage explosion, being the key problem that limits the application of LUT in edge devices. Focus on this, we propose an innovative separable mapping strategy(SMS) to realize dimensional reduction of LUT and solve the storage explosion problem.
- We design a dynamic discretization mechanism(DDM) to decompose the activation and compress the corresponding quantization scale, further shrinking the storage requirement without considerable accuracy loss.
- Our comprehensive experiments demonstrate that TinyLUT, constructed by SMS and DDM, achieves significant restoration accuracy over previous LUT methods with minimal storage consumption, showing its potential for application on edge devices.

2 Related Works

Deep Learning for Image Restoration: In recent years, image restoration performance has been rapidly developed through deep learning methods [20, 8, 21, 22, 10, 7, 11, 23, 24, 12, 25, 26] and achieved outstanding accuracy. The authors of [24] presented a super resolution method using very deep networks and residual learning. Zhang et al. [7] proposed a deep convolutional neural network with residual learning to handle image restoration tasks such as denoising and SISR. FFDNet [10] achieves more efficient and flexible than DnCNN [7] by down-sampled method. Also it have the ability to robustly control the trade-off between noise reduction and detail preservation. As a fast and memory-efficient network, FMEN [23] is constructed by enhanced residual block and high-frequency attention block. Dong et al. [8] explored a more efficient network by re-designing the SRCNN structure. In [27] author proposed a deep convolutional network within a Laplacian pyramid framework for fast and accurate image super-resolution. Denoising results of FastDVDnet [28] feature remarkable temporal coherence, very low flickering, and excellent detail preservation. However, these CNN methods suffer form intensive computations of convolutional operator, which limits their deployment on edge devices.

Lookup Table for Image Restoration: LUT-based methods are commonly used in hardware systems[29] and some embedded devices[30] to accelerate CNNs by pre-storing the results of complex functions with high efficient retrieval operations. In image restoration tasks, LUTs are commonly used in video coding [31, 32], tone-mapping [33] and image enhancement [17]. Furthermore, deep learning methods based on LUTs have also emerged in SISR[15, 14, 13] and gradually garnered attention. SRLUT[15] establishes a spatial mapping from a local RF block to the corresponding high-resolution pixel. In MuLUT[14] and SPLUT[13], the authors proposed the collaborating structure by multiple LUTs to increase the model's RF and improve the inference accuracy. Furthermore, Liu et al.[34] found that the interaction of space and channel features in vanilla convolution allows previous methods to increase the RF at the cost of linearly increasing the LUT size. They proposed RCLUT based on the Reconstructed Convolution(RC) module to enlarge the RF, achieving significant performance with less storage. Realizing the dilemma between the performance improvement and storage of LUT-based models, Li et al.[35] introduce the Diagonal-First Compression(DFC) framework to achieve a better trade-off between storage size and performance for image restoration. Additionally, they also designed a new structure, SPF-LUT, to further improve the performance of LUT-based models. However, the RF and channels of LUT-based methods are limited to adapt the trade-off between storage and accuracy. In comparison, our work is focused on developing a lightweight LUT-based framework to improve the image restoration accuracy.



(a) Network Architecture of TinyLUT (b) PwBlock

(c) TinyLUT

Figure 2: (a)The network overview of TinyLUT. (b)The structure of PwBlock. (c)The LUT framework overview of TinyLUT.

3 Method

3.1 Overview of TinyLUT

The structure of TinyLUT and foundational components are depicted in Fig 2. We design two parallel branches with cascaded LUTs to process Most Significant Bits (MSBs) and Least Significant Bits (LSBs) data respectively, and introduce separable mapping strategy and dynamic discretization mechanism.

According to the previous LUT-based methods we follow two simple design rules:(i) to improve model accuracy, the RF size needs to be increased; and (ii) the number and storage requirement of LUTs need to be reduced to adapt edge devices. Therefore, our CNN network (Fig 2(a)) is mainly inspired by the architecture of SRLUT [15] and DnCNN [7]. Specifically, the CNN model is built on standard convolution, depthwise convolution (DwConv) and PwBlock. The PwBlock 2(b) is a channel combining module based on SMS and described in Section 3.2. Based on SMS and DDM, the standard convolution, DwConv and PwBlock are mapped by LUT respectively to build standard convolution mapped LUT (ScLUT) , depthwise LUT (DwLUT) and pointwise LUT (PwLUT). In particular, we refer the philosophy of depthwise separable convolutions[36] to combine DwLUT and PwLUT to depthwise separable LUT(DSLUT). The mapping details for DwLUT and PwLUT are described in Section 3.2 and Section 3.3.

As in depthwise separable convolutions[36], the DwLUT optimizes the LUT size while maintaining a large effective RF. The PwLUT is used for cross-channel feature information integration and model channel number transformation. We introduce skip connections to fuse the real-value inputs and the retrieval outputs between DwLUT and PwLUT to avoid the degradation. The RF and output channels are set to 3×3 and 16. Due to each color channel having to be processed independently as in SRLUT [15], the input channel of ScLUT is set to 1.

According to the different number of DSLUT serial stacks, we build TinyLUT-F with 7 stacked. TinyLUT-S is built by ScLUT and double PwLUTs. As shown in Fig 2(c), TinyLUT is constructed by DwLUT and PwLUT in each branch. An additional PwLUT is added at the end of each branch to adjust the number of output channels of different image restoration task. In super resolution, the output channel number of the last PwLUT is r^2 to adapt to the pixelshuffle [21], where r is the upscaling factor. In order to further expand the RF size, we introduce rotational ensemble trick [15] in the inference.

3.2 Separable Mapping Strategy

In Table 1, we provide an estimation of the LUT size of four output channels and single input. It is evident that the SRLUT continues to face the issue of storage explosion, resulting in approximately

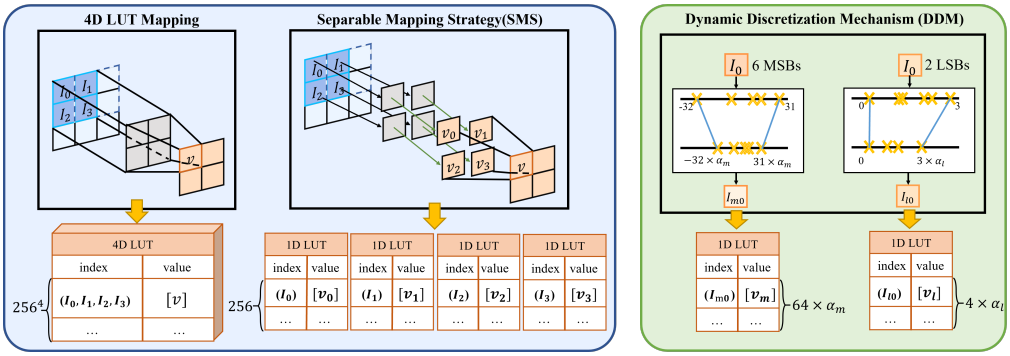


Figure 3: **Left**: Existing LUT-based method maps the 2×2 input convolution to a 4-dimensional LUT (4D LUT). SMS decomposes the convolution kernel to reduce the storage from $256^4 \times 8$bit to $256 \times 4 \times 8$bit. **Right**: The dynamic discretization mechanism (DDM) decouples the activation and compress the precision to reduce the storage to $(\alpha_m \times 64 + \alpha_l \times 4) \times 8$bit using $\alpha_m, \alpha_l \in [0, 1]$.

1767GB of memory overhead when processing 3×3 pixels of input simultaneously. In contrast, the proposed separable mapping strategy (SMS) alleviates the memory bottleneck associated with a large RF and reduces the required volume to just 36KB. Therefore, we enable the transfer of more deep convolutional network model into efficient LUT framework with fewer hardware cost.

The left image in Fig 3 illustrates 4-dimensional LUT(4D LUT) mapping the convolution 2×2 kernel with single output in existing methods [15, 14, 13]. It requires $256^4 \times 8$bit$= 4$GB and exceeds the storage capacity of edge devices. Therefore, we propose separable mapping strategy(SMS) to decouple 2×2 kernel into four 1×1 kernels which reduces the size of the LUT required to $256 \times 4 \times 8$bit$= 1$KB. Therefore, we propose the DwLUT. Firstly, from the original depthwise convolution operation, it can be obtained by:

$$F_{out} = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} I_{i,j} * w_{i,j} \tag{1}$$

The $x_{i,j}$ represents the input feature data, and F_{out} represents the output feature and $w_{i,j}$ stands for the convolution kernel, while $n \times n$ denotes the size. After transferring by original multiple dimensions LUTs, the retrieval process can be represented as:

$$F_{out} = LUT[x_{(0,0)}, \ldots, x_{(n-1,n-1)}] \tag{2}$$

In DwLUT, we utilize multiple 1×1 convolution kernels to build depthwise convolutions. As shown in Fig 3, we decompose the convolution kernel into n^2 of size $C_{out} \times 1 \times 1 \times 1$. Mapping by 1D LUTs, the original LUT retrieval process can be reconstructed as:

$$\hat{F}_{out} = \frac{1}{n^2} \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} LUT_{(i,j)}[x_{(i,j)}] \tag{3}$$

After decomposing and mapping, the storage consumption reduces from $256^{n^2} \times C_{out} \times 8$bit to $256 \times n \times n \times C_{out} \times 8$bit. Meanwhile, the output of the multiple 1D LUTs is averaged to achieve an approximation of the standard LUT inference.

The pointwise convolution applies a 1×1 convolution to combine the outputs the depthwise convolution [36]. The operation of pointwise convolution with a kernel size of $C_{out} \times C_{in} \times 1 \times 1$ is defined as:

$$F_{out} = \sum_{c=0}^{C_{in}} \sum_{t=0}^{C_{out}} x_c \cdot w_{c,t} \tag{4}$$

In this context, x represents the input data and w represents weights. It still faces the problem of exponential size increase in the LUT mapping process due to multiple channels input simultaneously. Then we decouple the input channels in pointwise convolution. Subsequently, it is decoupled into C_{in} number of $C_{out} \times 1 \times 1 \times 1$ convolution kernel. The storage space required for the pointwise

convolution, is reduced from $(256^{C_{in}}) \times C_{out} \times 8$bit to $256 \times C_{in} \times C_{out} \times 8$bit. The final output channel is set to C_{out}. At this point, the original cross-channel per-pixel LUT retrieval operation is reconstructed as:

$$\hat{F}_{out} = \frac{1}{C_{in}^2} \sum_{c=0}^{C_{in}} LUT_c[x_c] \tag{5}$$

As depicted in Eq 5, the quantity of LUTs is denoted as C_{in}. Additionally, the number of output channels for each LUT retrieval is represented as C_{out}. According to Eq 4 and Eq 5, the PwBlock in Fig 2(b) consists of several parallel branches with the number of channels. Each branch includes 3 pointwise convolutions followed by ReLU except for the last layer. The channels in the pointwise convolution is set to 64 and that of the last layer is set to 16.

3.3 Dynamic Discretization Mechanism

In previous methods, in order to further reduce the volume requirement of LUT, it is common to sample the indexing entries. Related algorithms [15, 14, 37] use interpolation methods to restore accuracy after table retrieval. However interpolating in large RF and multiple channels will inevitably result in a significant amount of computation. To address this issue, we proposes the dynamic discretization mechanism (DDM). It adaptively explores the parameterized clipping level of quantization range for each channel based on the gradients with Straight-Through Estimator (STE)[38, 39], which significantly optimize the equilibrium of accuracy and LUT size.

As shown in Fig 2, an 8bit input data is decomposed into MSBs and LSBs, then obtain output results through their respective branches and add them. It should be noted that the 8bit data stored in the LUT is defined in the domain of -128 to 127, while the data range of MSBs data is $[-m, m-1]$, and the data range of LSBs data is $[0, k]$. Therefore, in Eq 3 mapping depthwise convolution to LUTs for the MSBs branch, the LUTs size can be obtained by:

$$Size(MSB) = (2 \times m) \times n^2 \times r^2 \tag{6}$$

and the size of LUT in LSBs branch:

$$Size(LSB) = (k+1) \times n^2 \times r^2 \tag{7}$$

This operation reduces the index range of the LUT, realizes the discretization of the LUT and reduces the size.

The index range is 2^4 due to the activation data range are equal to LUTs indexes range in the mapping process. Therefore, inspired by activation quantization [18, 19], we use the learnable clipping parameter α to constrain the bit width of features. Specifically, α_{my} and α_{ly} are used to constrain the MSBs and LSBs of the layer y, respectively, to control the size of the LUT. For example, using F to denote feature data, this operation can be calculated as:

$$F_q = round(F * \alpha_y) \tag{8}$$

The LUT size is written as :

$$Size = (max(F_q) - min(F_q)) \times n^2 \times r^2 \tag{9}$$

In Eq 9, n^2 denotes the input entries and r^2 denotes the output entries respectively. In order to reduce the index number of LUTs, we initialize α to 0.8 and then apply L2-regularization for α with the $0.1\times$ regularization parameter λ used to compress the activation precision scale in each layer.

4 Experiments

4.1 Implementation Details

Training Settings To demonstrate the effectiveness of our framework, we evaluate TinyLUT on multiple datasets for SISR. The CNN model of TinyLUT is trained in an end-to-end manner. We use DIV2K [40] dataset which has been widely applied in image processing tasks. TinyLUT model is trained for 200000 iterations with Pytorch [41] on Nvidia 3090 GPU. We employed the Adam optimizer [42], where $\beta_1 = 0.9$, $\beta_2 = 0.999$. The learning rate is set at 5×10^{-3} and was dynamically adjusted using a cosine annealing mechanism [43]. We randomly cropped degraded data into 48×48

Table 2: Quantitative comparisons on 5 standard SISR test sets for an upscaling factor of 4. The size of TinyLUTs are smaller than other LUT methods and achieves better PSNR and SSIM average values with a good margin. $*$: The storage overhead of weight parameters in the DNN. The inference latency evaluation environments are the same as [15, 14]

	Method	Storage	Runtime		Set5	Set14	Urban100	BSD100	Manga109	Average
			Xiaomi 11	Raspberry 4B						
LUTs	SRLUT-S [15]	1304KB	137ms	247ms	29.82/0.8478	27.01/0.7355	24.02/0.6990	26.53/0.6953	26.80/0.8380	26.84/0.7631
	SPLUT-L [13]	18432KB	265ms	456ms	30.52/0.8630	27.54/0.7520	24.46/0.7191	26.87/0.7090	27.70/0.8581	27.42/0.7802
	MuLUT-SDY-X2 [14]	4159KB	242ms	403ms	30.60/0.8653	27.60/0.7541	24.46/0.7194	26.86/0.7110	27.90/0.8633	27.48/0.7808
	RCLUT [34]	1549KB	232ms	-	30.72/0.8677	27.67/0.7577	24.57/0.7253	26.95/0.7154	28.05/0.8655	27.59/0.7863
	SPF-LUT+DFC [35]	2066KB	-	-	31.05/0.8755	27.88/**0.7632**	24.81/0.7357	27.08/**0.7190**	28.58/0.8779	27.88/0.7943
	TinyLUT-S	**37KB**	**29ms**	**88ms**	30.22/0.8535	27.33/0.7450	24.19/0.7066	26.71/0.7042	27.21/0.8458	27.13/0.7710
	TinyLUT-F	171KB	146ms	387ms	**31.18/0.8771**	**28.01**/0.7630	**24.92/0.7397**	**27.13**/0.7184	**28.83/0.8798**	**28.01/0.7956**
DNN	SRCNN [12]	228KB*	-	27448ms	30.48/0.8628	27.50/0.7513	24.52/0.7221	26.90/0.7101	27.10/0.8457	27.30/0.7784
	VDSR [24]	2660KB*	-	106972ms	31.35/0.8830	28.02/0.7680	25.18/0.7540	27.29/0.7260	28.50/0.8812	28.07/0.8024
	FSRCNN [8]	48KB*	350ms	2143ms	30.71/0.8656	27.60/0.7543	24.61/0.7263	26.96/0.7129	27.90/0.8610	27.56/0.7840
	CARN-M [26]	1648KB*	3300ms	17609ms	31.82/0.8898	28.29/0.7747	25.62/0.7694	27.42/0.7350	29.85/0.8993	28.60/0.8136

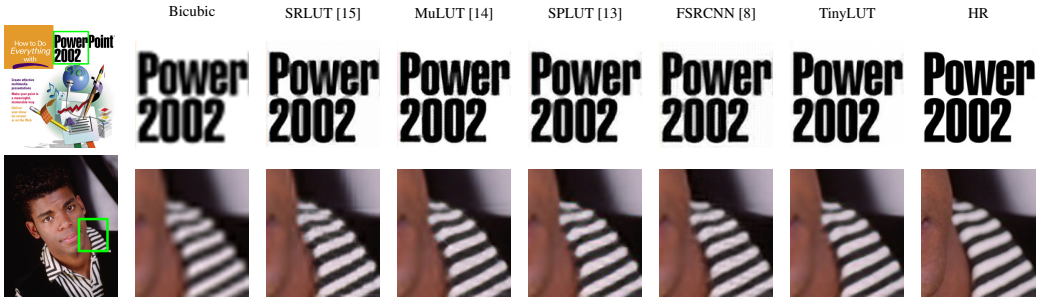


Figure 4: Qualitative comparisons of bicubic interpolation, SRLUT [15], MuLUT [14], SPLUT [13], FSRCNN [8], our TinyLUT and HR images.

patches with a batch size of 32. Data augmentation was performed through random rotations and flips. Throughout the experimental process, we employed PSNR and SSIM [44] as evaluation metrics for restoration accuracy. Besides, we measured and reported the runtime on multiple edge devices including Xiaomi 11 smartphones with a Qualcomm Snapdragon 888 CPU and Raspberry 4B.

Evaluation Settings For single-image super resolution, assessed the effectiveness of our method on five widely used benchmark datasets: Set5, Set14, BSD100 [45], Urban100 [46], and Manga109 [47]. We compare our method with various SISR algorithms based on interpolation which include methods based on deep learning including FSRCNN [8], VDSR [24], RCAN [25] and CARN-M [26], and SR methods based on LUTs, SRLUT [15], MuLUT [14], RCLUT [34], SPF-LUT [35] and SPLUT [13]. In tables, the best result is highlighted in **bold**.

4.2 Evaluation on Image Super Resolution

Accuracy and Storage The quantitative results of image super resolution are shown in Table 2. For single image super resolution task, the SSIM and PSNR values are computed at the Y-channel in the YCbCr color space. As reported in Table 2, we build a series of TinyLUT models with volume from 37KB to 171KB to fit various edge devices. Our TinyLUT-F model achieves 0.58dB PSNR increase with nearly $24\times$ storage reduction of MuLUT-SDY-X2 [14]. Even for the compact TinyLUT-S model, it only consumes 2.83% memory overhead and achieves 0.4dB PSNR and 0.02 SSIM improvement compared to SRLUT-S [15]. The qualitative results in Fig 4 of MuLUT-SDY-X2 [14] and SPLUT-L [13] have severe ringing artifacts in the white area. On the contrary, our method generates more natural textures and fewer artifacts. As shown in Table 2, TinyLUT-F achieves much higher PSNR with $9\times$ and $12\times$ lower storage consumption than RCLUT [34] and SPF-LUT+DFC [35], respectively. While TinyLUT achieves restoring clearer edges and $5\times$ lower latency on Raspberry 4B compared with computation-heavy method(FSRCNN[8]).

Table 3: Quantitative comparisons on Set12 and BSD68. The size of TinyLUTs are smaller than other LUT methods and achieves better PSNR and SSIM values with a good margin. $*$: The storage overhead of weight parameters in DNN. The evaluation environments are the same as [48]

Method	Storage	Runtime		Set12			BSD68			Average
		Xiaomi 11	Raspberry 4B	15	25	50	15	25	50	
SRLUT [15]	82KB	**7ms**	**21ms**	30.42	27.19	22.62	29.78	26.85	22.39	26.54
MuLUT-SDY-X2 [14]	289KB	26ms	44ms	31.50	28.94	25.46	30.63	28.18	24.97	28.28
MuLUT-SDYEHO-X2 [48]	978KB	51ms	89ms	31.77	29.18	25.47	30.89	28.34	24.96	28.44
TinyLUT-S	**22KB**	20ms	27ms	31.10	28.26	24.29	30.24	27.48	23.83	27.53
TinyLUT-F	187KB	146ms	254ms	**32.22**	**29.69**	**26.27**	**31.20**	**28.65**	**25.53**	**28.93**
DnCNN [7]	2220KB*	635ms	6859ms	32.86	30.44	27.18	31.73	29.23	26.23	29.61

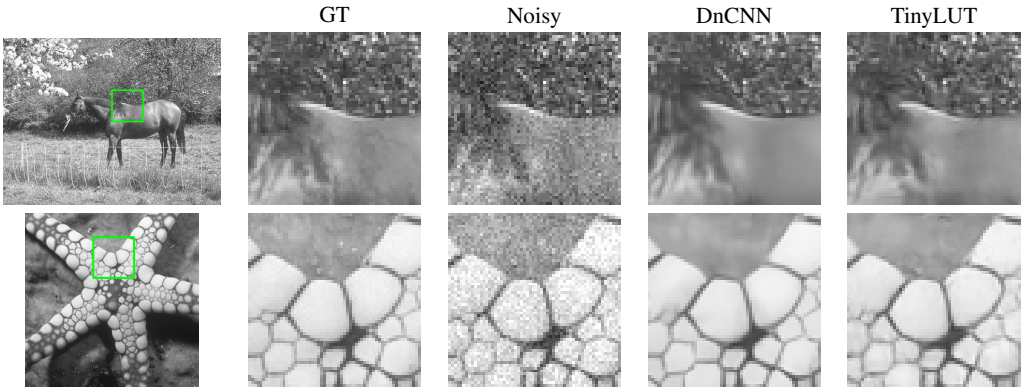


Figure 5: Qualitative comparisons of ground truth, noise image, DnCNN [7] and our TinyLUT.

Running Time Inference speed is also an important impact for edge devices deployment. In the view of Table 2, we report the runtime of LUT-base methods and other schemes. Results are obtained by averaging across 20 runs. Our approach achieves $4\times$ and $10\times$ acceleration on SISR compared to the fastest LUT-based and CNN method respectively. Meanwhile TinyLUT obtains absolute accuracy advantage compared to LUT-based methods. Overall the experiments indicate that the SMS is achieving significant storage reduction while organizing the model in a more flexible manner, thus capable of improving accuracy. TinyLUT achieves accuracy close to CNN and exhibits the enormous application potential of LUT-based method.

4.3 Evaluation on Image Denoising

In order to further show the capacity of the proposed TinyLUT framework, we remove the pixelshuffle and set the out channels of the last PwLUT to 1. Meanwhile, we adopt the residual learning formulation such as DnCNN [7] to accelerate model training. For additive white Gaussian noise (AWGN), we use Set12 and BSD68 benchmark datasets with noise level 15, 25 and 50. We compare our method with classical DNN method [7] and LUT methods [15, 48].

The quantitative results of image denoising is shown in Table 3. Compare with LUT-based methods, our method significantly improves PSNR with the minimal storage in image denoising. Meanwhile, TinyLUT-S achieves the fastest inference speed with only 22KB storage overhead compare with DNN based denoising schemes. The denoising results in Fig 5 illustrate that our TinyLUT is able to obtain similar visual quality to DnCNN [7]. The quantitative and qualitative experiments prove the effectiveness of storage reduction and inference acceleration of our approach.

4.4 Evaluation on Image Deblocking

In this subsection, image deblocking is used to further assess the generality of TinyLUT in image restoration tasks. Table 4 reports the comparison results of PSNR-B for LUT-based methods on Classic5[49] and LIVE1[50]. We also refer to the classical methods and DNN algorithms, including SA-DCT[51] and ARCNN[52]. TinyLUT-F achieves comparable PSNR-B results to the DNN method

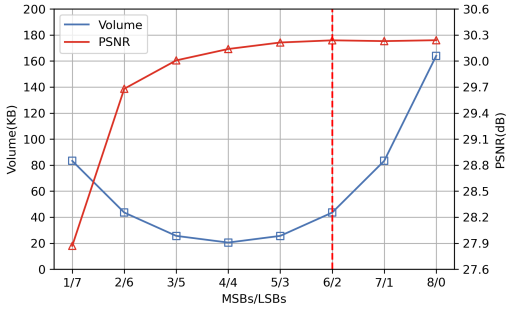


Figure 6: Illustration of the storage overhead and PSNR result for different combinations of MSBs and LSBs in TinyLUT-S for single image super resolution task.

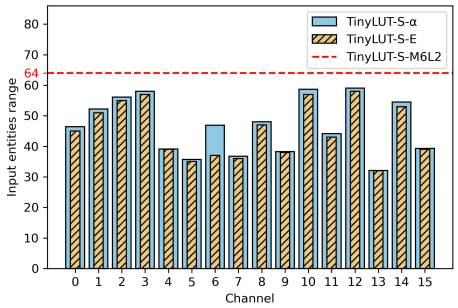


Figure 7: Illustration of the input entities precision range clipping by parameter α_{MSB} when mapping process in TinyLUT-S-α in SISR. TinyLUT-S-E indicates the input entities range of PwLUT as a statistical result.

and better than other LUT-based methods. The results indicate the generality of our TinyLUT to deblocking.

Table 4: The quantitative comparisons of PSNR-B on standard benchmark datasets for image deblocking under a quality factor of 10.

	Classical		DNN	LUT-based				
	JPEG	SA-DCT[51]	ARCNN[52]	SRLUT[15]	MuLUT[14]	SPF-LUT[35]	SPF-LUT+DFC[35]	TinyLUT-F
Storage	-	-	415KB	**81KB**	489KB	3017KB	595KB	187KB
Classic5	25.21	28.15	28.76	27.58	28.29	28.63	28.62	**28.74**
LIVE 1	25.33	28.01	28.77	27.69	28.39	28.62	28.61	**28.67**

4.5 Ablation Studies

We perform several ablation experiments for single image super resolution to demonstrate that our contributions in TinyLUT provides quality improvements.

The effectiveness of SMS We conduct an experiment with combinations of SMS and TinyLUT-S. In Table 5, SMS denotes the TinyLUT-S includes SMS without DDM when constructed the model with 8bit data values. Original denotes the TinyLUT-S without SMS and evaluates using the quantized CNN model due to the excessive storage overhead of its LUT model. The accuracy of SMS is competitive with corresponding mapped neural network with 8bit data in SISR. In particular, SMS scheme achieves significant accuracy improvement compared to uniformly sampled method [15] with sampling interval size of 2^4, while yielding over **7**$\times$ storage reduction. This demonstrates the effectiveness of storage reduction using SMS with minimal loss of accuracy.

Table 5: Impact of SMS. Ablation studies on TinyLUT-S for $4\times$ SISR.

Model	Method	Set5	Set14	Urban100	BSD100	Manga109	LUT Size
TinyLUT-S	Original	30.35	27.42	26.77	24.31	27.41	9.7×10^{24}PB
	Uniformly Sampled [15]	29.82	27.01	26.53	24.02	26.80	1.274MB
	SMS	30.24	27.33	26.72	24.19	27.23	**164KB**

The effectiveness of DDM As shown in Fig 6, the PSNR results of super resolution and image denoising are approximately equal to the global maximum at 6 MSBs and 2 LSBs (6M2L). Meanwhile the storage overhead is much smaller than the combination of 5 MSBs and 3 LSBs. Therefore we set the activation precision for the two branches of TinyLUT to 6 MSBs and 2 LSBs respectively.

As shown in Table 6, compared with the full LUT with 2^8 entries for each input pixel, DDM reduces TinyLUT storage overhead while ensuring accuracy in super resolution. The results prove that the

Table 6: Effects of DDM. Ablation studies for DDM with $4\times$ SISR.

Model	Method	Set5	Set14	Urban100	BSD100	Manga109	LUT Size
TinyLUT-S	Full LUT	30.24	27.33	26.72	24.19	27.23	164KB
	DDM	30.22	27.33	26.71	24.19	27.21	**37**KB

LUT sampling method based on DDM has higher compression ratio of storage than the 8bit full sampling LUT, and achieves a better trade-off between storage and accuracy.

As shown in Fig 7, the precision range of the MSBs activation data is adjusted more adaptively by clipping parameter α in SISR, which reduces the possible value range of the input entities of PwLUT in TinyLUT-S, and then reduces the size in MSBs branch. As a comparison, the marked red line in the figure is the s value when the input entities range is fixed to 6bit during mapping in the MSBs of PwLUT. It can be observed from the figure that compressed by the DDM, not only the LUT size is reduced, but also all possible values of input entities are covered in the mapping process. Compared with the 43.6KB of TinyLUT-S-6M2L with a fixed combination of 6 MSBs and 2 LSBs, the size of TinyLUT-S-E is reduced to 36.6KB via the adjustment of DDM, achieving a reduction of 15%. Therefore, DDM yields additional storage reduction compared to decomposing data into fixed MSBs and LSBs [13]. More intuitively, the interpretability of DDM is also illustrated in Fig 7.

5 Limitations and Future Direction

The method in this paper achieves significant reduction in storage with high compatibility when mapping convolutional neural network. However, there are difficulties in mapping other models such as Mamba [53] and Transformer [54, 55] using TinyLUT. This can be further explored the unified mapping approach for other model. Exploring additional image restoration tasks on edge devices using LUTs would also contribute to the community.

6 Conclusion

In this paper, we analyze previous successful LUT-based deep learning approaches and summarize the key problem of storage explosion, which limits the further popularization of LUT in image restoration on edge devices. To address the storage explosion, we propose the separable mapping strategy (SMS) and dynamic discretization mechanism(DDM) to decompose the kernel and activation, respectively. In particular, we design the TinyLUT framework based on SMS and DDM. By seamlessly integrating these innovations, TinyLUT-F sets a new record for SISR by achieving over 31dB PSNR on the Set5 dataset at just 171KB LUT storage. Overall, extensive experiments across seven benchmark datasets and two classic image restoration tasks demonstrate the effectiveness and efficiency of TinyLUT on resource-constrained devices.

7 Acknowledgement

This work was supported in part by the National Science and Technology Innovation 2030-Major Projects under Grant 2021ZD0114400, in part by Young Scientists Fund of the National Natural Science Foundation of China 62304162, and in part by China Postdoctoral Science Foundation under Grant 2024M762532, and in part by Postdoctoral Fellowship Program of CPSF under Grant GZC20241313, and in part by Shaanxi Provincial Natural Science Foundation for Basic Research Program 2024JC-YBMS-794, and in part by Fundamental Research Funds for the Central Universities under Grant XJSJ24090. Dr. Lin Gu was supported by JST Moonshot R&D Grant Number JPMJMS2011 Japan.

References

[1] R. Keys. Cubic convolution interpolation for digital image processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29(6):1153–1160, 1981.

[2] R. G. Keys. Cubic convolution interpolation for digital image processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29, 2003.

[3] Radu Timofte, Vincent Desmet, and Luc Vangool. A+: Adjusted anchored neighborhood regression for fast super-resolution. In *Springer International Publishing*, 2014.

[4] Radu Timofte, Vincent De, and Luc Van Gool. Anchored neighborhood regression for fast example-based super-resolution. In *2013 IEEE International Conference on Computer Vision*, pages 1920–1927, 2013.

[5] Hong Chang, Dit Yan Yeung, and Yimin Xiong. Super-resolution through neighbor embedding. In *IEEE Computer Society Conference on Computer Vision Pattern Recognition*, 2004.

[6] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8):2080 – 2095, 2007.

[7] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017.

[8] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. *Springer, Cham*, 2016.

[9] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Chen Change Loy, Yu Qiao, and Xiaoou Tang. ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks. *arXiv e-prints*, page arXiv:1809.00219, September 2018.

[10] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, September 2018.

[11] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 1833–1844, October 2021.

[12] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image Super-Resolution Using Deep Convolutional Networks. *arXiv e-prints*, page arXiv:1501.00092, December 2014.

[13] Cheng Ma, Jingyi Zhang, Jie Zhou, and Jiwen Lu. Learning Series-Parallel Lookup Tables for Efficient Image Super-Resolution. *arXiv e-prints*, page arXiv:2207.12987, July 2022.

[14] Jiacheng Li, Chang Chen, Zhen Cheng, and Zhiwei Xiong. Mulut: Cooperating multiple look-up tables for efficient image super-resolution. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, pages 238–256, Cham, 2022. Springer Nature Switzerland.

[15] Younghyun Jo and Seon Joo Kim. Practical single-image super-resolution using look-up table. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 691–700, 2021.

[16] Xiaohan Ding, Yuchen Guo, Guiguang Ding, and Jungong Han. Acnet: Strengthening the kernel skeletons for powerful cnn via asymmetric convolution blocks, 2019.

[17] Hui Zeng, Jianrui Cai, Lida Li, Zisheng Cao, and Lei Zhang. Learning Image-adaptive 3D Lookup Tables for High Performance Photo Enhancement in Real-time. *arXiv e-prints*, page arXiv:2009.14468, September 2020.

[18] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, I Jen Chuang, and Kailash Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks. 2018.

[19] Huixia Li, Chenqian Yan, Shaohui Lin, Xiawu Zheng, Yuchao Li, Baochang Zhang, Fan Yang, and Rongrong Ji. Pams: Quantized super-resolution via parameterized max scale. 2020.

[20] Jiayi Qin, Lihui Chen, Seunggil Jeon, and Xiaomin Yang. Progressive interaction-learning network for lightweight single-image super-resolution in industrial applications. *IEEE Transactions on Industrial Informatics*, 19(2):2183–2191, 2023.

[21] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[22] Zheng Hui, Xiumei Wang, and Xinbo Gao. Fast and accurate single image super-resolution via information distillation network. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 723–731, 2018.

[23] Zongcai Du, Ding Liu, Jie Liu, Jie Tang, Gangshan Wu, and Lean Fu. Fast and memory-efficient network towards efficient image super-resolution. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 852–861, 2022.

[24] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate Image Super-Resolution Using Very Deep Convolutional Networks. *arXiv e-prints*, page arXiv:1511.04587, November 2015.

[25] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *ECCV*, 2018.

[26] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, Accurate, and Lightweight Super-Resolution with Cascading Residual Network. *arXiv e-prints*, page arXiv:1803.08664, March 2018.

[27] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Fast and accurate image super-resolution with deep laplacian pyramid networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(11):2599–2613, 2019.

[28] Matias Tassano, Julie Delon, and Thomas Veit. Fastdvdnet: Towards real-time deep video denoising without flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[29] SHARIF, MD., and HAIDAR. High-performance mathematical functions for single-core architectures. *Journal of Circuits, Systems Computers*, 2014.

[30] Hakki Can Karaimer and Michael S. Brown. A software platform for manipulating the camera imaging pipeline. In *European Conference on Computer Vision*, 2016.

[31] Jun Young Lee, Jae Jin Lee, and Seong Mo Park. New lookup tables and searching algorithms for fast h.264/avc cavlc decoding. *IEEE Transactions on Circuits Systems for Video Technology*, 20(7):1007–1017, 2010.

[32] Sik Ho Tsang, Yui Lam Chan, and Wan Chi Siu. Region-based weighted prediction for coding video with local brightness variations. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(3):549–561, 2013.

[33] Ishtiaq Rasool Khan, Susanto Rahardja, Muhammad Murtaza Khan, Muhammad Mobeen Movania, and Fidaa Abed. A tone-mapping technique based on histogram using a sensitivity model of the human visual system. *IEEE Transactions on Industrial Electronics*, 65(4):3469–3479, 2018.

[34] Guandu Liu, Yukang Ding, Mading Li, Ming Sun, Xing Wen, and Bin Wang. Reconstructed convolution module based look-up tables for efficient image super-resolution. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12183–12192, 2023.

[35] Yinglong Li, Jiacheng Li, and Zhiwei Xiong. Look-up table compression for efficient image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 26016–26025, June 2024.

[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 e-prints*, page arXiv:1704.04861, April 2017.

[37] Canqian Yang, Meiguang Jin, Yi Xu, Rui Zhang, Ying Chen, and Huaida Liu. SepLUT: Separable Image-adaptive Lookup Tables for Real-time Image Enhancement. *arXiv e-prints*, page arXiv:2207.08351, July 2022.

[38] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. 2013.

[39] Nitsh Srivastava Geoffrey Hinton and Kevin Swersky. Neural networks for machine learning. *Coursera, video lectures*, 2012.

[40] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017.

[41] Anmol Chaudhary, Kuldeep Singh Chouhan, Jyoti Gajrani, and Bhavna Sharma. *Deep Learning With PyTorch*. Machine Learning and Deep Learning in Real-Time Applications, 2020.

[42] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Computer Science*, 2014.

[43] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. 2016.

[44] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, April 2004.

[45] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *IEEE International Conference on Computer Vision*, 2002.

[46] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5197–5206, 2015.

[47] Yusuke Matsui, Kota Ito, Yuji Aramaki, Toshihiko Yamasaki, and Kiyoharu Aizawa. Sketch-based Manga Retrieval using Manga109 Dataset. *arXiv e-prints*, page arXiv:1510.04389, October 2015.

[48] Jiacheng Li, Chang Wen Chen, Zhen Cheng, and Zhiwei Xiong. Toward dnn of luts: Learning efficient image restoration with multiple look-up tables. *IEEE transactions on pattern analysis and machine intelligence*, PP, 2023.

[49] Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Pointwise shape-adaptive dct for high-quality denoising and deblocking of grayscale and color images. *IEEE Transactions on Image Processing*, 16(5):1395–1411, 2007.

[50] H Sheikh. Live image quality assessment database release 2. *http://live.ece.utexas.edu/research/quality,2005.7*.

[51] Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Pointwise shape-adaptive dct for high-quality denoising and deblocking of grayscale and color images. *IEEE Transactions on Image Processing*, 16(5):1395–1411, 2007.

[52] Chao Dong, Yubin Deng, Chen Change Loy, and Xiaoou Tang. Compression artifacts reduction by a deep convolutional network. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 576–584, 2015.

[53] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2023.

[54] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[55] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

8 NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main contributions have been clearly summarized in the abstract and Section 1.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations of the work are discussed in Section 5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: Deduction and experiments in Section **??** and Section 4 demonstrate the validity of theoretical assumptions.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We clarify the main setups of our method in Section **??** and settings of experiments in Section 4 for the reproducibility in the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Upon publication, we intend to release the code for broader community access.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We specify the main training and test details in Section 4 to understand the results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [NA]

Justification: The benchmark result in Section 4 is already obtained by multiple testsets and noise level, and other previous works didn't reported the full experiment with the error bars.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The main hardware devices used in this paper are listed in Section 4

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conduct in this paper conform, in every respect, with the NeurlPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Upon publication, we intend to supplement the discussion of the broader implications.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The data and methods used in the paper are properly credited.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Upon publication, we intend to release the well documented code and data for broader community access.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.