Hundred-Kilobyte Lookup Tables for Efficient Single-Image Super-Resolution

Binxiao Huang[1] , **Jason Chun Lok Li**[1] , **Jie Ran**[1] ,
Boyu Li[1] , **Jiajun Zhou**[1] , **Dahai Yu**[2] , **Ngai Wong**[1]
[1] The University of Hong Kong
[2] TCL Corporate Research
{huangbx7, jasonlcl}@connect.hku.hk, {jieran, liboyu, jjzhou, nwong}@eee.hku.hk, dahai.yu@tcl.com

Abstract

Conventional super-resolution (SR) schemes make heavy use of convolutional neural networks (CNNs), which involve intensive multiply-accumulate (MAC) operations, and require specialized hardware such as graphics processing units. This contradicts the regime of edge AI that often runs on devices strained by power, computing, and storage resources. Such a challenge has motivated a series of lookup table (LUT)-based SR schemes that employ simple LUT readout and largely elude CNN computation. Nonetheless, the multi-megabyte LUTs in existing methods still prohibit on-chip storage and necessitate off-chip memory transport. This work tackles this storage hurdle and innovates hundred-kilobyte LUT (HKLUT) models amenable to on-chip cache. Utilizing an asymmetric two-branch multistage network coupled with a suite of specialized kernel patterns, HKLUT demonstrates an uncompromising performance and superior hardware efficiency over existing LUT schemes. Our implementation is publicly available at: https://github.com/jasonli0707/hklut.

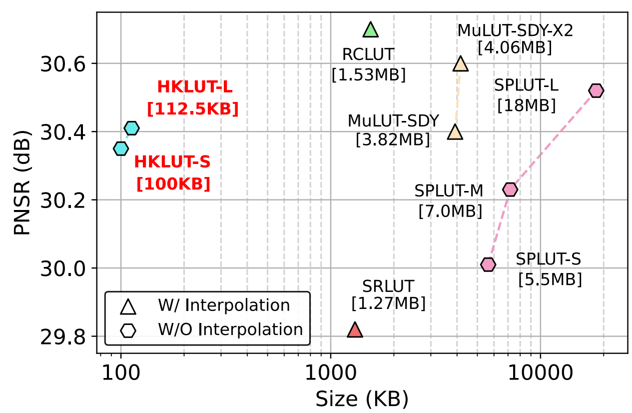

Figure 1: PSNR (dataset: Set5) vs. model size: HKLUTs are the *first* hundred-KB series of LUT-based SISR models targeting the sub-1MB regime (NB. x-axis is in log scale).

1 Introduction

Single image super-resolution (SISR) is a long-standing problem in computer vision that involves generating a high-resolution (HR) image from a low-resolution (LR) image. The goal is to recover high-frequency details that are lost in a downsampled image. Classical non-deep-learning methods can be divided into two main categories: interpolation and sparse coding. Interpolation approaches [Keys, 1981; Kirkland and Kirkland, 2010], such as bicubic interpolation, are simple and computationally efficient. They involve upsampling the LR image using a fixed kernel, e.g., a bicubic filter. However, they often suffer from limited generalizability and produce blurry images. Sparse-coding approaches [Timofte *et al.*, 2013; Timofte *et al.*, 2015], on the other hand, can achieve better results than the former but are computationally expensive and require hand-crafted features.

Like many other problems in computer vision, deep learning has revolutionized SISR. The mapping between LR and HR images can be learned directly from data, often resulting in super-resolved images of much better quality than classical methods. However, deep neural networks (DNNs) require a large number of floating-point operations, resulting in a high power budget. Recently, researchers have explored new schemes that combine lookup tables (LUT) with deep-learning SISR [Jo and Kim, 2021; Li *et al.*, 2022; Ma *et al.*, 2022]. Specifically, this approach pre-computes all possible input-output pairs of a deep SR network and stores them into an LUT for retrieval, thus skipping expensive computation during inference. As a result, LUT-based methods achieve inference speeds comparable to interpolation methods [Jo and Kim, 2021], but produce results with a learning-based quality. Moreover, LUT-based approaches are agnostic to software and hardware platforms, making them much more versatile than DNNs which require dedicated software and hardware support. Nonetheless, recent LUT-based SR schemes [Li *et al.*, 2022; Ma *et al.*, 2022] mainly focus on improving performance by enlarging the receptive field (RF), while overlooking the memory constraints. This contradicts the trending edge AI that often runs on edge devices strained by limited resources, e.g., the popular Ultra96 FPGA board has $<$ 1MB on-chip memory. In fact, existing LUT-based schemes generally require multi-megabyte LUTs where off-chip storage is inevitable. In addition, existing LUT-based approaches may still require expensive interpolation and/or floating-point operations, resulting in higher peak memory and energy costs. This highlights the challenge as well as the need for significant LUT size reduction and better architecture design to address the storage and power issues, while upkeeping performance.

To this end, this work systematically explores the relationship between the number of input pixels, RF, and SR performance. Our study leads to the design of a family of models called Hundred-Kilobyte Look-up Tables (HKLUTs) which offer a compelling balance between memory and performance. To eliminate the time-consuming and energy-intensive interpolation, a two-branch architecture as in SPLUT [Ma *et al.*, 2022] is employed. Yet instead of its symmetric branches [Ma *et al.*, 2022], we exploit the idea of effective receptive field (ERF) [Luo *et al.*, 2016] in different branches and reveal a larger RF is only required for the most-significant-bit (MSB) branch representing contextual semantic information, whereas a much smaller RF is sufficient for processing the least-significant-bit (LSB) branch.

This key insight leads to an innovative asymmetric parallel structure, which largely reduces the storage space by nearly half without sacrificing performance. Next, we extend our model to multistage by cascading LUTs to enlarge the RF. Our proposed architecture, in contrast to prior works, allows information exchange between two branches during stage transition, resulting in an uncompromising performance. Moreover, we are the first to utilize progressive upsampling on LUT-based approaches, which further reduces the LUT size by half while improving the final performance. Compared to other state-of-the-art (SOTA) LUT-based schemes, HKLUTs deliver competitive performance while being $> 10\times$ smaller in size (cf. Fig. 1). The main contributions of this paper are threefold:

- We systematically study the relationship between RF, the number of input pixels, and performance in the SR task, leading to our design of extra lightweight LUTs.
- We investigate the ERFs of two branches and propose an asymmetric parallel structure to reduce storage by $\approx 2\times$ without sacrificing performance.
- A novel multistage architecture is proposed which adopts a progressive upsampling strategy and enables the communication between the two branches during stage transition. This leads to an improved performance together with another $2\times$ reduction in size.

By seamlessly integrating the above techniques, HKLUTs constitute the smallest LUT-based SISR schemes in the literature offering a highly competitive performance.

2 Related Work

The quest for high-quality SR on edge devices has fueled the interest and research of effective and efficient SISR schemes in recent years, broadly categorized into classical, deep learning and LUT-based Approaches.

Classical Approaches. Basic linear interpolation methods [Keys, 1981; Kirkland and Kirkland, 2010], though simple and efficient, are limited in their abilities to adapt to the image content and often result in blurry images. Early example-based work [Timofte *et al.*, 2013; Timofte *et al.*, 2015] learned sparse dictionaries over pixels or patches to build a compact representation between LR and HR pairs. During inference, the HR output is obtained using the pre-computed projection matrix with the input features. However, it remains time-consuming for sparse coding models to compute the sparse representations.

Deep Learning Approaches. Deep learning methods have shown promising results in reconstructing image details for accurate super-resolution. Many deep learning SR works have been developed to reduce computational burdens by carefully adjusting the network structures, including, ESPCN [Shi *et al.*, 2016a], CARN [Ahn *et al.*, 2018], RRDB [Wang *et al.*, 2018] and IMDN [Hui *et al.*, 2019]. LapSRN [Lai *et al.*, 2017] and its extension MS-LapSRN [Lai *et al.*, 2018] employ Laplacian Pyramid to generate multi-scale predictions and progressively reconstruct HR images in multiple steps. FMEN [Du *et al.*, 2022] shined in the NTIRE 2022 challenge with its low memory cost and short runtime through the enhanced residual block and high-frequency attention module. Although some models can be executed in real-time on GPUs, it is impractical to deploy them on edge devices with constrained hardware resources.

LUT-based Approaches. While deep learning-based methods can produce impressive results, they often require heavy computations and large storage. To accelerate the inference of DNNs, dedicated software and hardware platforms such as CUDA and GPUs are required. This poses challenges for cost-sensitive and resource-restrictive edge devices such as smartphones and TVs. To address this, SRLUT [Jo and Kim, 2021] fuses DNN and LUT for SR applications by pre-computing and caching input-output pairs of a pre-trained SR network into a LUT for retrieval at test time. This approach eliminates expensive computation during DNN inference and achieves similar speed but better SR results than classical approaches. Two independent works around the same time, namely, MuLUT and SPLUT [Li *et al.*, 2022; Ma *et al.*, 2022], propose to expand the RF covered by cascading LUTs. MuLUT proposes the use of multiple complementary LUTs together with the rotation ensemble to solve the intrinsic limitation of a small RF in SRLUT. However, MuLUT preserves the costly interpolation step of SRLUT. On the other hand, SPLUT [Ma *et al.*, 2022] exploits a novel parallel framework where the quantization loss is compensated by using two parallel branches that separately handle the most significant 4 bits and least significant 4 bits. This eliminates the time-consuming interpolation step for a faster inference speed. SPLUT also suggests enlarging the RF through feature aggregation along the horizontal and vertical directions. In addition, the Reconstructed Convolution (RC) module proposed in [Liu *et al.*, 2023] can be transformed into 1D LUTs to efficiently expand the RF, thus improving performance. However, while these methods enhance performance compared to SRLUT, they require more storage space (> 1MB), making them less suitable for edge devices.

3 Development of HKLUT

3.1 Reduced Number of Input Pixels

The storage size of a LUT can be calculated using the formula $v^n \times r^2$, where v, n, and r are the number of quantization levels, number of input pixels, and upscaling factor, respectively. The first term v^n determines the number of entries, whereas the second term r^2 refers to the length of the output vector of each LUT. It is obvious that reducing the number of input pixels n can result in an exponential reduction in storage space. Previous studies have primarily focused on expanding the RF by utilizing a fixed number of pixels (say, 4) to improve the final performance. SPLUT [Ma *et al.*, 2022] uses feature aggregation to expand the RF, namely, the output feature maps generated from the previous LUT are divided into groups along the channel dimension and then padded horizontally or vertically. The padded feature maps are then added to generate a single-channel feature map, which serves as the input for subsequent LUTs. However, in order to generate multi-channel feature maps, the size of intermediate LUTs grows linearly with the number of output channels. This goes against our primary goal, which is to reduce storage space. Subsequently, we deliberately omit feature aggregation. On the other hand, SRLUT [Jo and Kim, 2021] and MuLUT [Li *et al.*, 2022] employ rotation ensemble to increase the area being covered. The final prediction $\hat{y}_i$ is computed as follows:

$$\hat{y}_i = \frac{1}{N}\frac{1}{M_k}\sum_{k=0}^{N}\sum_{j=0}^{M_k} R_j^{-1}(LUT_k(R_j(x_i))) \tag{1}$$

where x_i is the LR input, LUT_k is the kth LUT, R_j is the jth rotation operation, N is the number of kernels (or LUTs), and M_k is the number of rotation operations for the corresponding kernel. We remark that rotation ensemble, unlike feature aggregation, does not require extra storage.

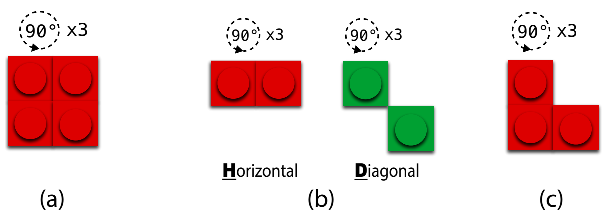

Figure 2: Three types of kernels covering a 3×3 RF with rotation ensemble via different numbers of input pixels. (a) SRLUT (b) HDLUT (c) LLUT.

Method	RF	# Pixels	LUT size	PSNR(dB)
SRNet	3×3	4	1.27 MB	**29.87**
LNet	3×3	3	76.77 KB	29.67
HDNet	3×3	2	**9.03 KB**	29.45

Table 1: Effect of the number of input pixels on PSNR, evaluated on Set5 dataset. "Net" is used to distinguish full-precision CNNs from quantized LUTs.

An important aspect that has been overlooked by previous works is that achieving the same RF is indeed feasible *with fewer pixels*. To fill in this gap, we design LLUT and HDLUT (named after their input-pixel shapes, cf. Fig. 2) to explore the relationship between the number of input pixels and the final PSNR subject to a prescribed RF size. Specifically, LLUT and HDLUT are designed to completely cover a 3×3 area without any overlapping pixels, except for the pivot pixel, using a rotation ensemble. Table 1 shows the PSNR improves with an increasing number of input pixels, which is consistent with the intuition that having more pixels gathers more information, at the cost of a higher storage, viz. a 4-pixel LUT requires megabytes of storage, whereas the performance of the 2-pixel LUT degrades obviously. To balance storage and performance, we devise HDBLUT, which consists of three 3-pixel kernels. Using rotation ensemble, it covers an entire 5×5 RF without any overlap, except for the pivot pixel (Fig. 3). In contrast to SRLUT and MuLUT, which have a 3×3 RF with 4 overlapping pixels and a 5×5 RF with 11 overlapping pixels, respectively, our designs, HDLUT and HDBLUT, aim to match the same RF with minimal input pixels. This achieves a near-optimal tradeoff between memory footprint and performance (a comparison of the RF of our designs vs others is in the Appendix). Using HDBLUT and HDLUT, along with the asymmetric parallel structure discussed in the sequel, we are able to outperform SRLUT with $> 6\times$ less storage.

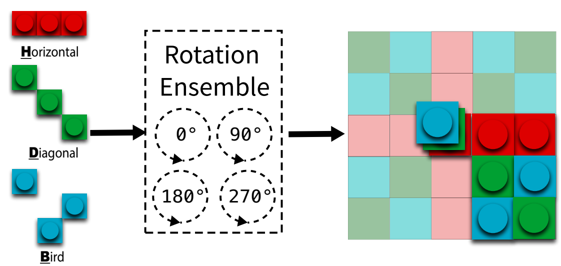

Figure 3: With rotation ensemble, HDBLUT can cover a 5×5 area with 3 three-pixel kernels. On the right, each square represents a single pixel, with a color indicating which kernel covers it.

3.2 Asymmetric Parallel Structure

Previous studies have attempted to reduce the size of LUTs by limiting the number of quantization levels to 17 and using interpolation to recover the quantization loss [Jo and Kim, 2021; Li *et al.*, 2022]. However, the interpolation stage is time-consuming and energy-intensive, hindering the practicality on edge devices. To address this, SPLUT [Ma *et al.*, 2022] proposes a parallel branch structure that avoids the need for interpolation. The structure divides the 8-bit input image into its most and least significant bits, viz. MSB and LSB branches each with 4 bits (viz. 16 levels) of information. The final SR image is obtained by fusing the results of the two branches at the last stage. While this approach gives promising results, it overlooks an important fact, namely, the effective receptive fields (ERFs) of MSB and LSB branches are completely different as they represent different kinds of information. Therefore, using the same sets of LUTs for both MSB and LSB branches might not be an optimal choice.

We trained two identical SRNets, using a 5×5 kernel for both MSB and LSB branches. The results from the

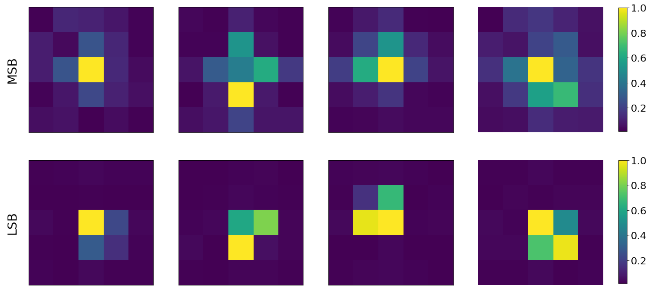

Figure 4: Four randomly selected effective receptive fields (ERFs). **Top:** Most significant 4 bits. **Bottom:** Least significant 4 bits. The brightness denotes model's sensitivity to that pixel, justifying the assymetric 5×5 and 3×3 kernels for respective branches.

two branches are summed and gradients are computed using mean-squared error (MSE) loss. Fig. 4 shows the ERF, in terms of normalized gradients obtained from randomly selected 4×4 regions of the SR result with respect to inputs, using the baby image from the Set5 dataset (cf. Fig. 5) (more demonstrations can be found in the Appendix). Although both MSB and LSB are processed under the same theoretical RF (i.e., 5×5), the results show that the MSB branch, which captures the high-level contextual semantic information, has a larger ERF. On the other hand, a smaller ERF is sufficient for the LSB branch to describe fine details. This observation justifies an asymmetric parallel structure that exploits the differences between MSB and LSB to save storage space. By using a set of 3-pixel LUTs (i.e., HDBLUT) corresponding to CNNs with a 5×5 RF for the MSB branch, and a set of 2-pixel LUTs (i.e., HDLUT) corresponding to CNNs with a 3×3 RF for the LSB branch, we can nearly halve the storage space without compromising performance (cf. Table 2).

3.3 Multistage

LUT-based methods [Ma *et al.*, 2022; Li *et al.*, 2022] have demonstrated the efficacy of cascaded stages to enlarge the RF for better performance. SPLUT [Ma *et al.*, 2022] splits the image into the MSB and LSB branches and deals with them separately with uniform quantization until the final addition, which prevents each branch from getting information from the other. MuLUT [Li *et al.*, 2022] utilizes quantization and interpolation to build the connection between stages, which requires later fine-tuning to compensate for the loss of information caused by quantization.

Here, we propose a new multistage framework as illustrated in Fig. 5. In each stage, we split the 8-bit input image into MSB and LSB branches for different LUT blocks to extract features and upscale the resolution. The features from both branches are then combined to generate the complete enlarged image. The input and output of each stage are 8-bit images with different resolutions, unless the upscaling factor is 1. This module can be stacked to create cascaded stages, where the latter stage can access information from the two branches of the former stages. Residual connections can be conveniently implemented. The entire flow does not require interpolation, which significantly speeds up inference.

Furthermore, previous works utilize cascaded stages to increase the RF for a better performance, where upsampling is usually done in the last step. It is intuitively difficult to reproduce a high-resolution patch from a few pixels in one single step, especially when the upscaling factor is large. Besides, the storage space of LUT is proportional to the square of the upscaling factor r, we can save half the storage space by dividing the upscaling factor $4\times$ (viz. $4 \times 4 = 16$) into two factors $2\times$ (viz. $2 \times 2 + 2 \times 2 = 8$). This simple yet previously unexplored progressive upsampling trick gives another booster to our proposed SISR scheme. Table 3 verifies the effectiveness of progressive upsampling in our LUT-based methods, along with further savings in storage.

4 Experiments

4.1 Implementation Details

Datasets and Metrics. We train the proposed HKLUT on the DIV2K dataset [Agustsson and Timofte, 2017], a popular dataset in the SR field. The DIV2K dataset provides 800 training images and 100 validation images with 2K resolution and the corresponding downsampled images. We use the widely adopted Peak Signal-to-Noise Ratio (PSNR) and structural similarity index (SSIM) [Wang *et al.*, 2004] as evaluation metrics. Five well-known datasets: Set5 [Bevilacqua *et al.*, 2012], Set14 [Zeyde *et al.*, 2012], BSDS100 [Martin *et al.*, 2001], Urban100 [Huang *et al.*, 2015] and Manga109 [Matsui *et al.*, 2017] are benchmarked. Following other LUT-based methods, we mainly focus on SR tasks with an upscaling factor of 4. We also report the runtime, theoretical energy cost, and peak memory of generating a 1280×720 HR image from a 640×360 LR image.

Experimental setting. Before converting into LUTs, the network is trained with 800 training images in DIV2K for 200k iterations with a batch size of 16 on Nvidia RTX 3090 GPUs. We utilize the Adam optimizer [Kingma and Ba, 2014] ($\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1e-8$) with the MSE loss to train the HKLUT. The initial learning rate is set to 5×10^{-4}, which decays to one-tenth after 100k and 150k iterations, respectively. We randomly crop LR images into 48×48 patches as input and enhance the dataset by random rotation and flipping. The runtime is measured on an Intel Core i5-10505 CPU with 16GB RAM, averaged over 10 runs.

Baselines. Aligning with prior work, we evaluate HKLUT against several well-recognized SISR methods, such as interpolation-based methods (nearest neighbor [Bevilacqua *et al.*, 2012], bilinear [Kirkland and Kirkland, 2010] and bicubic [Keys, 1981] interpolation), sparse-coding-based methods (ANR [Timofte *et al.*, 2013] and A+ [Timofte *et al.*, 2015]), DNN-based methods (CARN [Du *et al.*, 2022] and RRDB [Wang *et al.*, 2018]), and LUT-based methods (SRLUT [Jo and Kim, 2021], SPLUT [Ma *et al.*, 2022], MuLUT [Li *et al.*, 2022] and RCLUT [Liu *et al.*, 2023]).

4.2 Quantitative Results

Asymmetric Parallel Structure. To demonstrate the effectiveness of our proposed asymmetric parallel structure, we utilize the 2-pixel HDLUT with 3×3 RF and 3-pixel HDBLUT with 5×5 RF to investigate the relationship between RF, LSB, and MSB for the $4\times$ SR. Table 2 shows

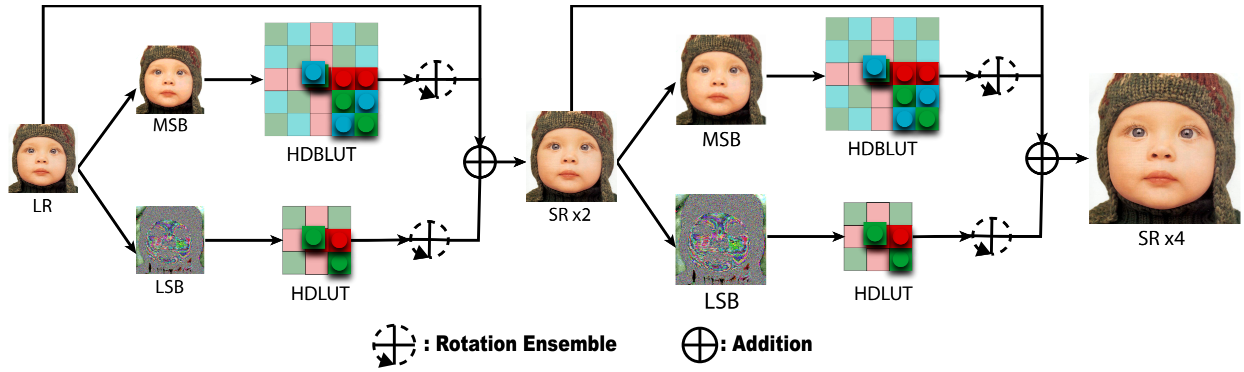

Figure 5: Overall architecture of the proposed method (HKLUT-S).

Method		RF	# Pixels	Storage	Set 5	Set 14	BSDS100	Urban100	Manga109
MSB	LSB	MSB/LSB	MSB/LSB		PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM
HD	HD	$3 \times 3/3 \times 3$	2/2	16KB	29.39/0.835	26.79/0.730	26.38/0.692	23.73/0.689	26.28/0.825
HD	HDB	$3 \times 3/5 \times 5$	2/3	200KB	29.40/0.835	26.79/0.730	26.39/0.692	23.73/0.689	26.29/0.825
HDB	HD	$5 \times 5/3 \times 3$	3/2	200KB	29.98/0.848	27.13/0.741	26.61/0.702	24.01/0.700	26.88/0.837
HDB	HDB	$5 \times 5/5 \times 5$	3/3	384KB	29.99/0.849	27.15/0.742	26.63/0.703	24.02/0.701	26.90/0.839

Table 2: Asymmetric parallel structure: a smaller LSB-branch RF yields negligible performance drops but large storage savings.

# Stages	Upscale	Storage	Runtime	Set5 PSNR/SSIM	Set14 PSNR/SSIM	BSDS100 PSNR/SSIM	Urban100 PSNR/SSIM	Manga109 PSNR/SSIM
1	4	200KB	108.8ms	29.97/0.84	27.13/0.741	26.61/0.702	24.01/0.700	26.86/0.837
2	1×4	213KB	115.5ms	29.99/0.848	27.14/0.742	26.62/0.702	24.02/0.700	26.86/0.837
2	2×2	100KB	201.7ms	30.35/0.859	27.39/0.748	26.73/0.706	24.23/0.711	27.38/0.852
3	$1 \times 2 \times 2$	112.5KB	212.1ms	30.34/0.860	27.40/0.748	26.75/0.707	24.25/0.712	27.46/0.853
3	$2 \times 1 \times 2$	112.5KB	235.3ms	30.41/0.860	27.44/0.749	26.78/0.707	24.27/0.713	27.51/0.854
3	$2 \times 2 \times 1$	112.5KB	475.0ms	30.43/0.861	27.47/0.750	26.79/0.707	24.31/0.713	27.58/0.856

Table 3: Ablating various progressive upsampling strategies. Runtime is measured for outputting a 1280×720 image on a desktop CPU averaged across 10 runs. We refer to models in the lime and blue rows as HKLUT-S and HKLUT-L, respectively.

that using HDLUTs for both branches results in worse performance (29.39dB for Set5). A negligible increase (+0.01dB) in performance is achieved when we replace the LSB branch with the larger-RF HDBLUT, at a cost of over 12 times the growth in model size (200KB vs. 16KB). However, by simply swapping the two branches, i.e., using HDBLUT for MSB and HDLUT for LSB, we obtain a significant boost in performance (+0.58dB) with the same storage and computing. When using HDBLUT for both MSB and LSB branches, we only obtain a marginal increase in PSNR (+0.01dB), at a cost of nearly doubling the storage (384KB vs. 200KB). This validates our conjecture that a 2-pixel LUT is sufficient for the LSB because of the small ERF, while a 3-pixel LUT is necessary for the MSB to capture contextual information such as texture. In short, the proposed asymmetric parallel structure allows for the efficient allocation of resources and saves half of the storage space without sacrificing performance.

Multistage. Table 3 examines various upsampling strategies to demonstrate the effectiveness of progressive upsampling. Following the asymmetric parallel structure, HDBLUT and HDLUT are used for each stage's MSB and LSB branches, respectively. Comparing the 1st and 3rd rows, using an asymmetric parallel structure, 2-stage progressive upsampling (2×2) shows a clear advantage in terms of SR performance compared to 1-stage upsampling ($4\times$) (30.35dB vs. 29.97dB on Set 5), while requiring only half the storage space (100KB vs. 200KB). The advantage of progressive upsampling becomes even clearer when compared to results on the 2nd row, which corresponds to the same 2-stage but with upsampling in a single step (1×4) (-0.36dB). Our results show that $2 \times 1 \times 2$ upsampling achieves the best tradeoff by considering storage, runtime, and performance. We, therefore, refer to models listed on the 3rd and 5th rows as HKLUT-S (highlighted in lime) and HKLUT-L (highlighted in blue), respectively.

Performance and Storage In Table 4, we compared our proposed HKLUT with well-recognized SISR approaches such as interpolation, sparse-coding, and CNNs, as well as the latest LUT-based methods. Our approach has a clear advantage in storage, which aligns with the objective of this paper. HKLUT-S, being the smallest parameterized model, only requires 100KB, which is $10\times$ less than the second smallest SRLUT (1.27MB) with a much better performance. Both HKLUT-S and HKLUT-L outperform all classical baselines, including A+, which is significantly larger than HKLUT-S (15.17MB vs. 100KB). Although CNN-based approaches ex-

Method	Size	Ratio	Set5 PSNR/SSIM	Set14 PSNR/SSIM	BSDS100 PSNR/SSIM	Urban100 PSNR/SSIM	Manga109 PSNR/SSIM
Nearest [Bevilacqua *et al.*, 2012]	-	-	26.25/0.737	24.65/0.653	25.03/0.629	22.17/0.615	23.45/0.741
Bilinear [Kirkland and Kirkland, 2010]	-	-	27.55/0.788	25.42/0.679	25.54/0.646	22.69/0.635	24.21/0.767
Bicubic [Keys, 1981]	-	-	28.42/0.810	26.00/0.702	25.96/0.667	23.14/0.657	24.91/0.787
ANR [Timofte *et al.*, 2013]	1.43MB	14.64	29.70/0.842	26.86/0.737	26.52/0.699	23.89/0.696	26.18/0.821
A+ [Timofte *et al.*, 2015]	15.17MB	155.34	30.27/0.860	27.30/0.750	26.73/0.709	24.33/0.719	26.91/0.848
CARN-M [Ahn *et al.*, 2018]	1.59MB	16.28	31.82/0.890	28.29/0.775	27.42/0.731	25.62/0.769	29.85/0.899
RRDB [Wang *et al.*, 2018]	63.83MB	653.62	32.60/0.900	28.88/0.790	27.76/0.743	26.73/0.807	31.16/0.916
SRLUT [Jo and Kim, 2021]	1.27MB	13.05	29.82/0.847	27.01/0.736	26.53/0.695	24.02/0.699	26.80/0.838
SPLUT-S [Ma *et al.*, 2022]	5.5MB	56.32	30.01/0.852	27.20/0.743	26.68/0.702	24.13/0.706	27.00 0.843
SPLUT-M [Ma *et al.*, 2022]	7MB	71.68	30.23/0.857	27.32/0.746	26.74/0.704	24.21/0.709	27.20 0.848
SPLUT-L [Ma *et al.*, 2022]	18MB	184.32	30.52/0.863	27.54/0.752	26.87/0.709	24.46/0.719	27.70 0.858
MuLUT-SDY [Li *et al.*, 2022]	3.82MB	39.15	30.40/0.860	27.48/0.751	26.79/0.709	24.31/0.714	27.52/0.855
MuLUT-SDY-X2 [Li *et al.*, 2022]	4.06MB	41.60	30.60/0.865	27.60/0.754	26.86/0.711	24.46/0.719	27.90/0.863
RCLUT [Liu *et al.*, 2023]	1.51MB	15.46	30.72/0.868	27.67/0.758	26.95/0.715	24.57/0.725	28.05/0.8655
HKLUT-S (**ours**)	**100 KB**	1	30.35/0.859	27.39/0.748	26.73/0.706	24.23/0.711	27.38/0.852
HKLUT-L (**ours**)	**112.5 KB**	1.13	30.41/0.860	27.44/0.749	26.78/0.707	24.27/0.713	27.51/0.854

Table 4: Different SR methods on 5 benchmark datasets. The size ratio is w.r.t. HKLUT-S. The 4 groups from top to bottom are interpolation, sparse-coding, CNN & LUT approaches, respectively.

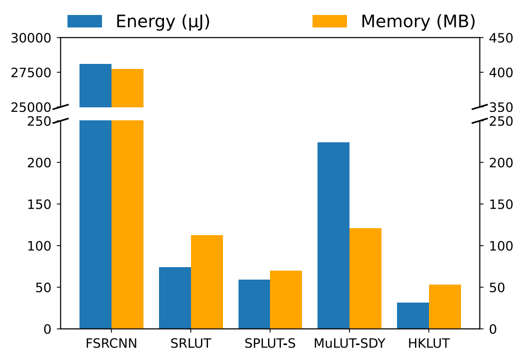

Figure 6: Energy cost and peak memory to output a 1280×720 image through $2\times$ SR.

hibit good performance in terms of PSNR and SSIM, they often require larger storage. For instance, RRDB is $650\times$ larger than our HKLUT-S, making it challenging to deploy on edge devices. Compared to LUT-based approaches, our methods not only require less storage but also achieve comparable performance. HKLUT-L outperforms MuLUT-SDY on the Set5 dataset (30.41dB vs. 30.40dB) with $35\times$ less storage. Despite being $70\times$ smaller, HKLUT-S significantly outperforms SPLUT-M (30.35dB vs. 30.23dB).

Energy & Peak Memory. Following the setting in MuLUT [Li *et al.*, 2022], Fig. 6 shows the theoretical energy costs and peak memory usages of FSRCNN (32.70dB) and several LUT-based methods (31.73dB$\sim$32.35dB) for generating a 1280×720 image through $2\times$ SR. We estimated the theoretical energy cost using the same protocol as AdderSR [Song *et al.*, 2021], and computed the peak memory usage using *memray*[1]. For LUT-based approaches, we used their smallest variants. One of the major advantages of LUT-based methods is their ability to drastically reduce energy and memory consumption. Even FSRCNN, the simplest CNN for SISR, consumes $900\times$ and $7.5\times$ more energy and memory compared to HKLUT. The feature map aggregation module of SPLUT not only instantiates larger LUTs but also

Method	FSRCNN	SRLUT	SPLUT-S	MuLUT-SDY	HKLUT
CPU (ms)	332.64	1412.24	197.49	5128.17	**148.67**
Raspberry (ms)	3589.86	8021.56	1096.73	35637.12	**908.52**

Table 5: Comparing runtimes for generating a 1280×720 image through $2\times$ SR. Results are obtained by averaging across 10 runs.

requires floating-point operations, resulting in increased energy consumption and peak memory usage. Meanwhile, both SRLUT and MuLUT consume a significant amount of energy for the interpolation operation. In contrast, HKLUT gets rid of energy-intensive interpolation and solely relies on integer operations. Fig. 6 shows that HKLUT exhibits the lowest energy cost and peak memory usage amongst all competitors.

Runtime. Using the same settings as for calculating energy and peak memory, we compared the runtimes of FSRCNN and LUT-based approaches on both a desktop CPU and the Raspberry Pi 4 Model B. To ensure a fair comparison, we used the official Python code released by the authors, without relying on native mobile implementation or any speedup tricks. We recorded the average runtime of generating a 1280×720 image from a 640×360 image in Table 5. It is clear that the 4-simplex interpolation bottlenecks the inference speed of SRLUT and MuLUT at the final stage. SPLUT and our approaches, featuring parallel branches, exhibit shorter inference times on Raspberry Pi and CPU. Unsurprisingly, HKLUT achieves the fastest inference speed.

4.3 Qualitative Results

Fig. 7 presents a visual comparison between the bicubic interpolation baseline and different LUT-based approaches on images selected from five benchmark datasets. The results clearly demonstrate that LUT-based approaches improve sharpness compared to bicubic interpolation which generates blurry images. The 3rd row (Manga109) shows that the SR-LUT approach exhibits severe rippling artifacts around the

[1]Measured with *memray* (https://github.com/bloomberg/memray) on a desktop CPU.

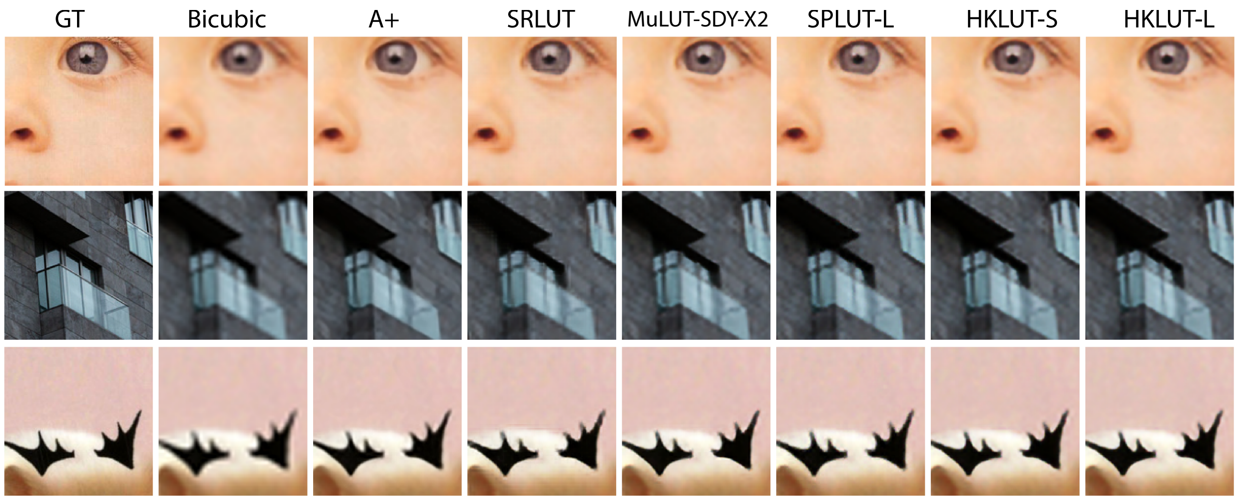

Figure 7: Qualitative results on selected images from Set5, Urban100 and Manga100 datasets.

edges, which may be caused by its limited RF. HKLUT-S produces similar sharpness to MuLUT-SDY-X2 and SPLUT-L, despite being more than $40\times$ and $180\times$ smaller, respectively.

5 Ablation Studies

5.1 Ablation on Kernel Shapes

To obtain the HDB kernel for our MSB branch, we conducted an ablation study on kernel shape. Instead of exhausting all possible combinations, we employed softmax during training with an annealing temperature to select the most important pixel per column. After training, we evaluated the binarized kernels. We performed 20 runs with different random seeds using a one-stage model, with the LSB branch fixed as the HD kernel. Each run consisted of 20k (10%) training iterations. Next, we selected the top 3 best-performing candidates with non-overlapping kernels and retrained them from scratch for 200k iterations. Fig. 8 shows the 3 best-performing for MSB branch, where HDB (i.e. (a)) ranks as the best.

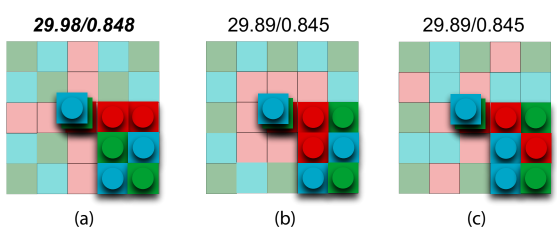

Figure 8: The 3 best-performing kernel designs for MSB are reported, along with the PSNR/SSIM for $4\times$ SR on the Set5 dataset.

Cascading Approach	Set5 PSNR/SSIM	Set14 PSNR/SSIM	BSDS100 PSNR/SSIM
HKLUT	**30.35/0.859**	**27.39/0.748**	**26.73/0.706**
SPLUT	29.32/0.836	26.55/0.724	26.15/0.686

Table 6: Ablation on different ways of cascading LUTs. HDBLUT and HDLUT are used for MSB and LSB branch, respectively.

5.2 Ablation on LUT Cascading

In previous work [Ma *et al.*, 2022], a series-parallel structure was proposed to remove the need for interpolation. This structure divides an 8-bit image into MSB and LSB parts and processes them separately. The series structure involves cascading LUTs within each branch. Communication between branches only occurs at the end of the model to generate the final SR image by combining information. The output activations of each LUT in intermediate stages are clipped between [-8, 7] and quantized into 16 levels. However, we believe that this approach is counter-intuitive and suboptimal. On the other hand, our method allows communication between branches during stage transition. The output of each stage is clipped between [0, 255] represented in INT8, enabling meaningful results even in intermediate stages. Table 6 demonstrates the superior performance of our approach compared to the multistage structure in SPLUT [Ma *et al.*, 2022] (30.35dB vs. 29.32dB for Set5), both using the 2×2 progressive upsampling technique.

6 Conclusion

This paper aims to develop minimalist LUTs for efficient LUT-based SISR. We propose a new class of models called HKLUTs employing kernels with fewer input pixels while retaining the RF, resulting in an exponential reduction in storage without compromising performance. In particular, HKLUT consists of an asymmetric parallel structure that employs big-little kernel patterns to adapt to the underlying characteristics of MSB and LSB branches. A progressive multi-stage architecture, without expensive interpolation between stages, is devised to reduce storage by half. By seamlessly fusing these innovations, HKLUT models set the first record to attain $>$ 30dB on the Set 5 dataset at merely one-tenth of 1MB storage. Extensive experiments verify that HKLUT is a versatile alternative to previous large LUT-based approaches on resource-constrained devices.

Acknowledgments

This work was supported in part by the Theme-based Research Scheme (TRS) project T45-701/22-R of the Research Grants Council (RGC), Hong Kong Special Administrative Region (HKSAR), and in part by the HKU-TCL Joint Research Centre for Artificial Intelligence.

Contribution Statement

Binxiao Huang and Jason Li contributed equally to this paper.

References

[Agustsson and Timofte, 2017] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 126–135, 2017.

[Ahn *et al.*, 2018] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network. In *Proceedings of the European conference on computer vision (ECCV)*, pages 252–268, 2018.

[Bevilacqua *et al.*, 2012] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie Line Alberi-Morel. Low-complexity single-image super-resolution based on non-negative neighbor embedding. 2012.

[Du *et al.*, 2022] Zongcai Du, Ding Liu, Jie Liu, Jie Tang, Gangshan Wu, and Lean Fu. Fast and memory-efficient network towards efficient image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 853–862, 2022.

[Glorot *et al.*, 2011] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings, 2011.

[Huang *et al.*, 2015] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5197–5206, 2015.

[Hui *et al.*, 2019] Zheng Hui, Xinbo Gao, Yunchu Yang, and Xiumei Wang. Lightweight image super-resolution with information multi-distillation network. In *Proceedings of the 27th acm international conference on multimedia*, pages 2024–2032, 2019.

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

[Keys, 1981] Robert Keys. Cubic convolution interpolation for digital image processing. *IEEE transactions on acoustics, speech, and signal processing*, 29(6):1153–1160, 1981.

[Kingma and Ba, 2014] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Kirkland and Kirkland, 2010] Earl J Kirkland and Earl J Kirkland. Bilinear interpolation. *Advanced Computing in Electron Microscopy*, pages 261–263, 2010.

[Lai *et al.*, 2017] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 624–632, 2017.

[Lai *et al.*, 2018] 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, 2018.

[Li *et al.*, 2022] Jiacheng Li, Chang Chen, Zhen Cheng, and Zhiwei Xiong. Mulut: Cooperating multiple look-up tables for efficient image super-resolution. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XVIII*, pages 238–256. Springer, 2022.

[Liu *et al.*, 2023] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12217–12226, 2023.

[Luo *et al.*, 2016] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the effective receptive field in deep convolutional neural networks. *Advances in neural information processing systems*, 29, 2016.

[Ma *et al.*, 2022] Cheng Ma, Jingyi Zhang, Jie Zhou, and Jiwen Lu. Learning series-parallel lookup tables for efficient image super-resolution. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XVII*, pages 305–321. Springer, 2022.

[Martin *et al.*, 2001] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 416–423. IEEE, 2001.

[Matsui *et al.*, 2017] Yusuke Matsui, Kota Ito, Yuji Aramaki, Azuma Fujimoto, Toru Ogawa, Toshihiko Yamasaki, and Kiyoharu Aizawa. Sketch-based manga retrieval using manga109 dataset. *Multimedia Tools and Applications*, 76:21811–21838, 2017.

[Shi *et al.*, 2016a] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE* *conference on computer vision and pattern recognition*, pages 1874–1883, 2016.

[Shi *et al.*, 2016b] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016.

[Song *et al.*, 2021] Dehua Song, Yunhe Wang, Hanting Chen, Chang Xu, Chunjing Xu, and DaCheng Tao. Addersr: Towards energy efficient image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15648–15657, 2021.

[Timofte *et al.*, 2013] Radu Timofte, Vincent De Smet, and Luc Van Gool. Anchored neighborhood regression for fast example-based super-resolution. In *Proceedings of the IEEE international conference on computer vision*, pages 1920–1927, 2013.

[Timofte *et al.*, 2015] Radu Timofte, Vincent De Smet, and Luc Van Gool. A+: Adjusted anchored neighborhood regression for fast super-resolution. In *Computer Vision–ACCV 2014: 12th Asian Conference on Computer Vision, Singapore, Singapore, November 1-5, 2014, Revised Selected Papers, Part IV 12*, pages 111–126. Springer, 2015.

[Wang *et al.*, 2004] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[Wang *et al.*, 2018] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European conference on computer vision (ECCV) workshops*, pages 0–0, 2018.

[Zeyde *et al.*, 2012] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *Curves and Surfaces: 7th International Conference, Avignon, France, June 24-30, 2010, Revised Selected Papers 7*, pages 711–730. Springer, 2012.