SOLA-SR: LEARNING SCALE-ORDERED LATENT SUBSPACES FOR ARBITRARY-SCALE SUPER-RESOLUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Latent representations have driven significant recent advances in large language models, and advance efforts to adapt such representations to computer vision have inspired new paradigms for image generation. However, discrete latent representations constrain the modeling of naturally continuous image features, especially in super-resolution (SR), where reconstruction requires richer, finer-grained image features. Inspired by COLA-DLM's separation of continuous latent representations from conditional decoding, we propose Scale-Ordered Latent Subspaces for Arbitrary-Scale Super-Resolution (SOLA-SR). SOLA-SR separates shared image encoding from scale-dependent selective decoding, extracting image features in a single encoding pass and progressively exposing them during decoding as the target scale increases. Specifically, a learnable orthogonal transformation reorganizes feature channels while preserving their spatial layout. Continuous scale gating selects a compact subset of feature dimensions at lower scales and progressively activates additional channel groups as the scale increases. We algebraically fold the selected subspace's linear mapping into the decoder's first query layer, thereby skipping computations associated with inactive input dimensions. A single encoding thus supports deterministic reconstruction at multiple scales without iterative latent-variable sampling. By coupling scale-dependent representation organization with selective decoding, SOLA-SR offers a path toward combining fine-grained scale control, reconstruction quality, and inference efficiency in arbitrary-scale super-resolution. Experiments indicate an overall trend toward faster end-to-end inference and distinguish the roles of representation organization and iterative latent-variable generation.

1 INTRODUCTION

Super-resolution (SR) is an important research direction in computer vision (Dong et al., 2016; Kim et al., 2016; Shi et al., 2016; Lim et al., 2017; Zhang et al., 2018b), with applications in remote-sensing imagery (Panangian & Bittner, 2025), object detection (Kim et al., 2024a; Shermeyer & Van Etten, 2019; Cui et al., 2026), and game graphics enhancement (Mercier et al., 2023); consumer mobile zoom provides another deployment example (Wronski et al., 2019). However, a limitation of fixed-scale SR is that a model trained for selected output factors may not extend to other scales. Researchers have therefore proposed arbitrary-scale super-resolution (ASSR) (Hu et al., 2019; Chen et al., 2021), which aims to reconstruct across scale factors while maintaining visual quality.

As shown in Figure 1(a)–(c), current ASSR research can be broadly categorized into three groups. The first uses scale-aware dynamic upsampling: scale-conditioned filter generation or feature adaptation enables a single network to accommodate different magnification factors (Hu et al., 2019; Wang et al., 2021; Wu et al., 2021). The second models continuous neural functions, using coordinate-conditioned implicit image functions (Chen et al., 2021; Lee & Jin, 2022; Cao et al., 2023; Jiang et al., 2025) or neural operators across resolutions (Wei & Zhang, 2023) to reconstruct images on different target sampling grids. The third uses Gaussian-primitive-driven continuous rendering, representing continuous images or features with two-dimensional Gaussian fields and generating the target resolution through scale-dependent rendering and sampling (Hu et al., 2025; Chen et al.,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

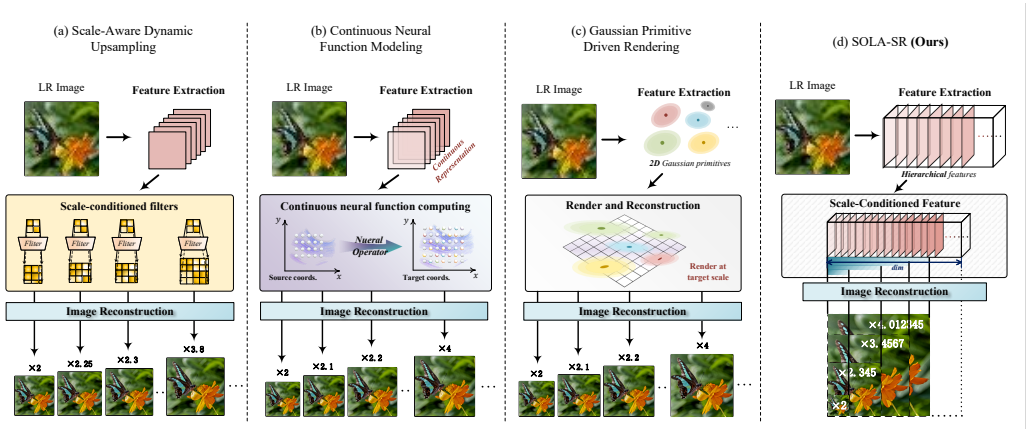

Figure 1: Comparison of representative ASSR paradigms. (a) Scale-Aware Dynamic Upsampling: Scale-conditioned filters or feature adaptation support reconstruction at different magnification factors. (b) Continuous Neural Function Modeling: Coordinate-conditioned implicit functions reconstruct images on arbitrary target grids. (c) Gaussian-Primitive-Driven Rendering: Continuous Gaussian primitives are rendered at the requested scale. (d) SOLA-SR (Ours): We reorganize a shared image encoding into scale-ordered latent subspaces, enabling compact low-scale decoding and progressively richer reconstruction at higher scales.

2025). However, a key challenge remains: learning image representations that support accurate reconstruction of fine details across scales, particularly at large scales or those unseen during training.

Recently, COLA-DLM (Guo et al., 2026) introduced a language generation framework based on diffusion modeling in a continuous latent space. It constructs continuous latent representations through a text variational autoencoder, separating global semantic modeling in the latent space from token generation in a conditional decoder. Inspired by this approach, can we separate a shared encoding of image content from scale-dependent selective decoding in ASSR? Specifically, we organize the features extracted in a single image-encoding pass into scale-ordered nested subspaces, allowing lower-scale reconstruction to use only a subset of dimensions while progressively exposing additional dimensions as the target scale increases.

To this end, we propose Scale-Ordered Latent Subspaces for Arbitrary-Scale Super-Resolution (SOLA-SR), which organizes a shared image representation according to the target scale and selectively exposes a scale-dependent subset of latent dimensions for reconstruction. We introduce a learnable orthogonal transformation between the encoder and the implicit decoder to reorganize feature channels while preserving their two-dimensional spatial layout. Continuous scale-dependent group weights allow lower-scale reconstruction to use only a subset of channels and progressively incorporate additional groups as the scale increases. In our primary instantiation, we use an RDN encoder (Zhang et al., 2018c) together with a LIIF coordinate-based decoder (Chen et al., 2021). To translate this representation selection into computational savings during decoding, we algebraically fold the selected subspace-to-feature-space mapping into the first query layer of the LIIF decoder. This layer thus directly processes the selected dimensions and omits inactive feature inputs. SOLA-SR can therefore reuse a single image encoding for deterministic reconstruction across scales without iterative sampling, while using more compact query inputs at lower scales and reducing decoding overhead.

Our contributions are as follows:

1. We transfer COLA-DLM's idea of hierarchical continuous latent representations to arbitrary-scale super-resolution and propose SOLA-SR to support finer-grained super-resolution.

2. We combine orthogonal feature reorganization with continuous scale gating to progressively activate features according to scale. By algebraically folding the linear transformations, we reduce computations associated with inactive feature inputs in the decoder's first layer, thereby lowering computational overhead.

3. We establish a high-resolution-assisted representation-capacity experiment and a low-resolution-conditioned detail-generation experiment to evaluate the roles of representation organization and iterative latent-variable generation, respectively.

2 Related Work

2.1 Arbitrary-Scale Super-Resolution

Existing arbitrary-scale super-resolution methods aim to enable a single model to reconstruct images at different continuous or non-integer scales. Continuous optical zoom benchmarks document the additional challenges of real imagery (Fu et al., 2024), while arbitrary-scale blind SR considers unknown degradation (Weng et al., 2024; Neshatavar et al., 2024).

Scale-aware dynamic upsampling. Meta-SR introduces scale-conditioned filter prediction, where the target scale and coordinate information are used to dynamically generate upsampling filters (Hu et al., 2019). ArbSR further incorporates scale information into feature adaptation and scale-aware upsampling through dynamic filters (Wang et al., 2021). These methods mainly modify the reconstruction process while maintaining a shared feature representation. They condition convolution kernels or feature transforms on the scale factor while reusing one shared encoder across all magnifications, so scale adaptation is concentrated at the upsampling stage rather than in the encoded representation.

Continuous neural function modeling. LIIF models images as continuous implicit functions by querying RGB values from latent features and spatial coordinates (Chen et al., 2021). Subsequent methods such as LTE, CiaoSR, and HIIF improve continuous reconstruction by introducing frequency modeling, attention mechanisms, or hierarchical representations (Lee & Jin, 2022; Cao et al., 2023; Jiang et al., 2025). SRNO formulates super-resolution as a neural operator mapping between resolution spaces (Wei & Zhang, 2023); IM-LUT instead learns scale-dependent interpolation mixtures (Park et al., 2025). Related implicit representations have also been used for face SR and image stitching (Tsai et al., 2024; Kim et al., 2024b). These approaches primarily improve how a representation is decoded or adapted to the output grid.

Gaussian-primitive-driven rendering. Gaussian-based methods represent continuous images or features using learnable Gaussian primitives and reconstruct target resolutions through scale-dependent rendering (Hu et al., 2025; Chen et al., 2025). Each primitive carries position, color, and opacity attributes and is differentiably splatted onto an arbitrary target grid, bypassing a fixed discrete pixel lattice; scale changes are therefore handled mainly through the projection and rendering step rather than by restructuring the encoded representation.

Different from previous ASSR methods, SOLA-SR investigates how the latent representation itself can be organized according to scale.

3 Method

3.1 Arbitrary-Scale Super-Resolution

For LR image x and target scale s, an encoder gives $z = E(x) \in \mathbb{R}^{H\times W\times C}$ and a coordinate-query decoder predicts $\hat{y}_s(q) = D(z, q, c(s))$ at query q with cell size $c(s)$. The output grid fixes image dimensions, while our scale-dependent subspace selects latent dimensions for each query. The RDN–LIIF implementation retains spatial layout, relative coordinates, cell decoding, feature unfolding, and local ensemble (Zhang et al., 2018c; Chen et al., 2021); notation is expanded in Appendix D.

3.2 Overview of SOLA-SR

Figure 2 summarizes SOLA-SR: an orthogonal basis reorganizes one image encoding, a continuous scale gate selects nested channel groups, and exact selective decoding feeds only selected dimensions to the query network. Encoding can be reused across scales, although this reuse is also possible in other coordinate-query models; independent execution repeats it. Reconstruction is deterministic and requires no iterative latent-variable sampler.

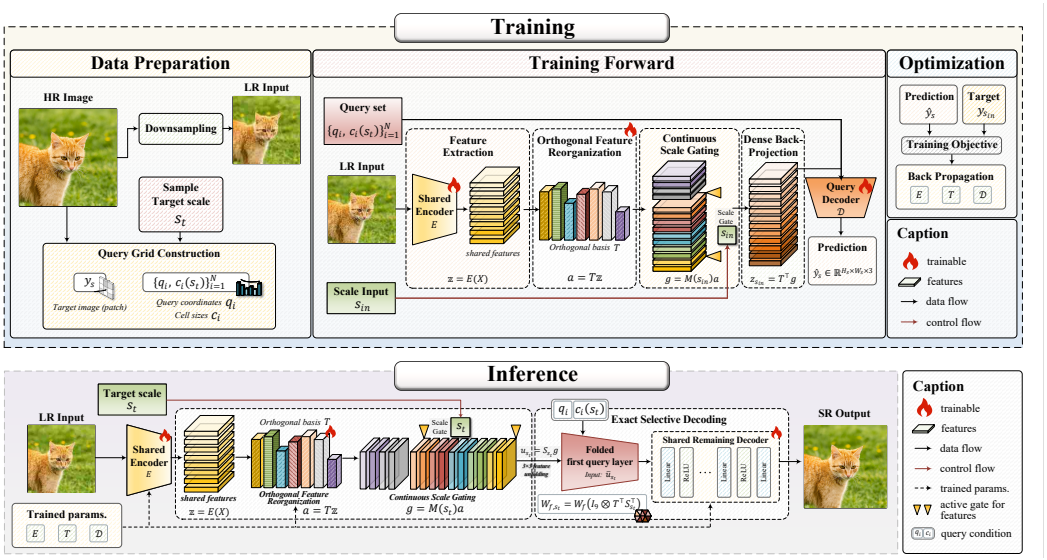

Figure 2: SOLA-SR training and inference. Training optimizes the shared encoder, orthogonal basis, and decoder with dense back-projection. Inference folds the selected subspace mapping into the first query layer. Flame icons mark trained modules; scale gating follows a fixed schedule.

3.3 Orthogonal Feature Reorganization

To reduce the loss of useful information that can arise from directly truncating the encoder's feature channels, we learn an orthogonal transformation that redistributes information across channels before scale-dependent selection. Orthogonal feature reorganization learns a shared channel basis in which compact prefixes can support reconstruction at lower scales and additional groups can be exposed at higher scales.

The transform $a = Tz$ acts on channels independently at every spatial location while preserving the spatial layout and the information in the full representation. With a learnable strictly upper-triangular U, we form skew-symmetric A and the Cayley transform:

$$A = U - U^\top, \qquad T = (I + A)^{-1}(I - A), \qquad T^\top T = I. \tag{1}$$

T starts at the identity. Orthogonality preserves full-representation norm and invertibility without mixing locations. Here "ordered" is structural: the fixed gate opens groups as nested prefixes, so the active set at a lower scale is a subset of that at any higher scale; this is set by the gate schedule rather than a learned channel-importance score, and we do not claim a semantic or frequency ordering.

3.4 Continuous Scale Gating

A shared channel basis alone does not determine which feature groups should participate at each scale. To link feature usage to the target magnification, we formulate a continuous scale-gating rule that progressively increases the weights of successive channel groups, allowing the active representation to expand with scale without an additional learned router. Continuous scale gating maps the target magnification to an ordered set of channel weights, allowing adjacent scales to share a compact representation while progressively exposing additional feature groups.

Let C_0 channels always be active, followed by G groups of C_g channels, so $C = C_0 + GC_g$. Between endpoints $s_0 < s_1$, release coordinate r and group weights α_j are

$$r(s) = \mathrm{clip}\left(G\frac{s - s_0}{s_1 - s_0}, 0, G\right), \qquad \alpha_j(s) = \mathrm{clip}(r(s) - j, 0, 1). \tag{2}$$

The diagonal gate is

$$M(s) = \mathrm{diag}(\mathbf{1}_{C_0}, \alpha_0(s)\mathbf{1}_{C_g}, \ldots, \alpha_{G-1}(s)\mathbf{1}_{C_g}). \tag{3}$$

Here $\mathbf{1}_k$ contains k ones. The dense readout returns to the decoder's original basis:

$$z_s = T^\top g = T^\top M(s)Tz. \tag{4}$$

Weights vary continuously while active supports remain nested; a partially weighted group counts as active, so $k_s = C_0 + C_g\lceil r(s)\rceil$. The target output grid is independent of group release. Endpoint settings are in Appendix E.

3.5 Exact Selective Decoding

To avoid redundant computation on inactive channels, we derive an exact selective decoding scheme that folds the subspace reconstruction map into the first query layer, reducing its feature input width while preserving algebraic equivalence to dense decoding. Feature selection alone does not reduce computation unless the decoder execution is modified accordingly.

For a feature vector $z(p) \in \mathbb{R}^C$, let $S_s \in \{0,1\}^{k_s \times C}$ select all nonzero-gated channels in order. Then $S_s^\top S_s M(s) = M(s)$, including partial weights, and

$$u_s(p) = S_s M(s) T z(p), \qquad z_s(p) = T^\top S_s^\top u_s(p). \tag{5}$$

With $B_s = T^\top S_s^\top$, the selection identity yields $B_s u_s(p) = z_s(p)$; gate weights are already in u_s.

For a 3×3 neighborhood, let $\bar{z}_s$ and $\bar{u}_s$ concatenate the nine full and selected feature vectors in neighborhood-major order. Then $\bar{z}_s = (I_9 \otimes B_s)\bar{u}_s$, and a first-layer feature block $W_f \in \mathbb{R}^{d \times 9C}$ folds to $W_{f,s} \in \mathbb{R}^{d \times 9k_s}$:

$$W_{f,s} = W_f(I_9 \otimes T^\top S_s^\top), \qquad W_f \bar{z}_s = W_{f,s}\bar{u}_s. \tag{6}$$

Here $\otimes$ is the Kronecker product. The channel-major permutation and implementation contraction are given in Appendix D; feature buffers and weight columns must use the same ordering.

With relative-coordinate/cell input v, its weights W_v, and bias b, the activation is $h = \sigma(W_{f,s}\bar{u}_s + W_v v + b)$. Later layers and local ensemble are unchanged. Exact receives $9k_s$ rather than $9C$ feature values without reconstructing a zero-padded query. Dense and Exact share trained weights and gates; their algebraic equality does not require bitwise identity across floating-point kernels.

Equation 6 gates selected features; the cached implementation equivalently folds fractional weights into the first-layer block and prepares it per request. The fully open branch bypasses folding. Detailed reuse and tensor layout are in Appendix F.

Computation and reuse. For N_s output pixels, L ensemble queries, and first-layer width d, selection saves $9LN_s d(C - k_s)$ first-layer MACs; this is not an end-to-end speedup estimate. The saving vanishes when $k_s = C$. Other costs and reuse scopes are separated in Table 15.

The additional runtime check in Appendix I tests the implemented tensor layout and folded execution without changing this algebraic derivation.

3.6 Training Objective

The encoder, orthogonal basis, and decoder are jointly trained on dense gated outputs using robust pixel, perceptual, mean-color, and gradient losses; the monotonic scale gate is fixed:

$$\mathcal{L} = \mathcal{L}_{\text{char}} + \lambda_p \mathcal{L}_{\text{perc}} + \lambda_c \mathcal{L}_{\text{color}} + \lambda_g \mathcal{L}_{\text{grad}}. \tag{7}$$

The Charbonnier pixel term and frozen-AlexNet LPIPS (Zhang et al., 2018a) are supplemented by color-mean and Sobel-gradient ℓ_1 terms (Appendix D). Gradients update the encoder, decoder, and Cayley parameters. Exact inference folds these trained weights without retraining.

4 Experiments

4.1 Experimental Details

We evaluate frozen models on common LR/HR inputs. All experiments are run on a single NVIDIA RTX 3090 GPU with 24 GB of memory. Training and inference recipes differ by method, so cross-method rankings are descriptive rather than architecture-only comparisons. Integer evaluation covers 657 image–scale pairs across Set5, Set14, B100, and Urban100. Additional-scale evaluation covers 219 images per scale; its main table equally weights the four dataset means, while the dataset-resolved entries appear in Appendix B. Mechanism and implementation diagnostics use fixed development weights; their protocols are given in the Appendix.

4.2 Comparison with Existing ASSR Methods

Table 1: Multi-method integer-scale comparison on Set5 (5 images), Set14 (14), B100 (100), and Urban100 (100). Each dataset contains nine columns: $\times 2$, $\times 3$, and $\times 4$, each reporting Y-PSNR (dB), Y-SSIM, and LPIPS-Alex. Each value is averaged over images within one dataset at one scale. Dark, medium, and pale red mark the first, second, and third distinct displayed values within each dataset, scale, and metric; ties share rank. Fixed-scale controls are unranked. [1]CiaoSR uses native tiling; [2]GSASR uses the labelled-head adapter with a row chunk of 16.

Method	Venue	Set5									Set14								
		$\times 2$			$\times 3$			$\times 4$			$\times 2$			$\times 3$			$\times 4$		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Meta-SR/RDN(2019)	CVPR 2019	38.259	0.9614	0.0549	34.746	0.9298	0.1220	32.509	0.8991	0.1711	34.040	0.9209	0.0885	30.588	0.8472	0.1988	28.848	0.7884	0.2767
ArbSR–ArbRCAN(2021)	ICCV 2021	38.284	0.9616	0.0544	34.761	0.9303	0.1216	32.515	0.8985	0.1689	34.087	0.9218	0.0878	30.645	0.8483	0.2003	28.848	0.7883	0.2754
LIIF–RDN(2021)	CVPR 2021	38.185	0.9613	0.0553	34.672	0.9295	0.1227	32.507	0.8992	0.1701	33.973	0.9212	0.0881	30.522	0.8472	0.1975	28.792	0.7875	0.2768
LTE–RDN(2022)	CVPR 2022	38.248	0.9615	0.0550	34.724	0.9301	0.1223	32.601	0.9001	0.1714	34.102	0.9215	0.0877	30.574	0.8476	0.1984	28.863	0.7890	0.2764
CiaoSR–RDN[1](2023)	CVPR 2023	38.335	0.9619	0.0553	34.876	0.9308	0.1205	32.680	0.9012	0.1689	34.237	0.9227	0.0873	30.660	0.8485	0.1978	28.939	0.7907	0.2746
SRNO–RDN(2023)	CVPR 2023	38.333	0.9620	0.0553	34.835	0.9308	0.1209	32.674	0.9011	0.1700	34.260	0.9235	0.0868	30.693	0.8495	0.1978	28.952	0.7909	0.2741
AnySR–EDSR(2024)	TIP 2024	38.057	0.9609	0.0577	34.490	0.9282	0.1226	32.362	0.8976	0.1744	33.703	0.9194	0.0940	30.448	0.8443	0.2050	28.737	0.7854	0.2796
DIIF(2024)	ICME 2024	37.600	0.9595	0.0622	33.917	0.9236	0.1271	31.596	0.8872	0.1690	33.264	0.9157	0.1035	30.041	0.8375	0.2165	28.257	0.7747	0.2779
GSASR–RDN[2](2025)	ICCV 2025	38.397	**0.9623**	0.0521	**34.907**	**0.9314**	0.1194	**32.835**	**0.9029**	0.1679	34.166	0.9228	0.0834	30.740	**0.8512**	0.1927	28.959	**0.7923**	0.2710
HIIF–RDN(2025)	CVPR 2025	38.327	0.9619	0.0546	34.828	0.9310	0.1205	32.718	0.9014	0.1684	**34.323**	**0.9244**	0.0858	30.763	0.8504	0.1980	**28.982**	0.7913	0.2737
EDSR (fixed)(2017)	CVPRW 2017	38.230	0.9614	0.0543	34.685	0.9296	0.1230	32.504	0.8991	0.1712	33.959	0.9204	0.0887	30.539	0.8467	0.1983	28.814	0.7881	0.2752
RCAN (fixed)(2018b)	ECCV 2018	38.303	0.9617	0.0538	34.765	0.9303	0.1214	32.650	0.9005	0.1694	34.140	0.9218	0.0871	30.629	0.8478	0.2001	28.848	0.7886	0.2733
RDN (fixed)(2018c)	CVPR 2018	38.265	0.9616	0.0538	34.714	0.9298	0.1242	32.463	0.8992	0.1700	34.001	0.9213	0.0874	30.562	0.8470	0.2018	28.793	0.7871	0.2785
SOLA-SR	This work	**38.409**	**0.9623**	**0.0399**	34.883	0.9311	**0.0627**	32.816	0.9027	**0.1002**	34.304	0.9241	**0.0692**	**30.778**	0.8509	**0.1172**	28.965	0.7921	**0.1798**

Method	Venue	B100									Urban100								
		$\times 2$			$\times 3$			$\times 4$			$\times 2$			$\times 3$			$\times 4$		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Meta-SR/RDN(2019)	CVPR 2019	32.372	0.9015	0.1393	29.293	0.8096	0.2757	27.760	0.7426	0.3648	33.051	0.9365	0.0553	28.944	0.8678	0.1381	26.709	0.8056	0.2068
ArbSR–ArbRCAN(2021)	ICCV 2021	32.393	0.9023	0.1382	29.321	0.8111	0.2739	27.752	0.7433	0.3603	33.095	0.9372	0.0544	28.989	0.8692	0.1368	26.638	0.8041	0.2053
LIIF–RDN(2021)	CVPR 2021	32.327	0.9013	0.1449	29.268	0.8101	0.2777	27.750	0.7424	0.3642	32.892	0.9355	0.0568	28.827	0.8666	0.1385	26.685	0.8043	0.2085
LTE–RDN(2022)	CVPR 2022	32.365	0.9018	0.1427	29.309	0.8106	0.2747	27.775	0.7433	0.3628	33.056	0.9368	0.0551	28.982	0.8689	0.1375	26.816	0.8076	0.2056
CiaoSR–RDN[1](2023)	CVPR 2023	32.403	0.9023	0.1407	29.344	0.8115	0.2739	27.832	0.7454	0.3594	33.301	0.9389	0.0533	29.175	0.8717	0.1334	27.101	0.8143	0.1970
SRNO–RDN(2023)	CVPR 2023	32.423	0.9028	0.1381	29.360	0.8117	0.2715	27.824	0.7451	0.3574	33.336	0.9392	0.0520	29.140	0.8717	0.1334	26.982	0.8122	0.1997
AnySR–EDSR(2024)	TIP 2024	32.211	0.9002	0.1457	29.165	0.8072	0.2799	27.657	0.7402	0.3639	32.278	0.9296	0.0630	28.455	0.8579	0.1498	26.456	0.7965	0.2157
DIIF(2024)	ICME 2024	31.882	0.8962	0.1683	28.833	0.7997	0.3060	27.339	0.7300	0.3673	31.380	0.9219	0.0777	27.493	0.8388	0.1847	25.491	0.7670	0.2553
GSASR–RDN[2](2025)	ICCV 2025	32.433	0.9030	0.1359	29.362	**0.8132**	0.2697	27.846	**0.7472**	0.3592	**33.535**	**0.9408**	0.0506	**29.354**	**0.8761**	0.1293	27.154	**0.8179**	0.1960
HIIF–RDN(2025)	CVPR 2025	**32.473**	**0.9036**	0.1360	29.400	0.8131	0.2710	**27.856**	0.7466	0.3578	33.401	0.9398	0.0519	29.273	0.8731	0.1333	27.130	0.8146	0.1970
EDSR (fixed)(2017)	CVPRW 2017	32.369	0.9022	0.1389	29.275	0.8100	0.2774	27.730	0.7424	0.3619	32.991	0.9362	0.0552	28.828	0.8664	0.1388	26.651	0.8040	0.2044
RCAN (fixed)(2018b)	ECCV 2018	32.403	0.9027	0.1374	29.312	0.8106	0.2755	27.750	0.7434	0.3583	33.198	0.9374	0.0533	29.024	0.8690	0.1344	26.751	0.8069	0.1970
RDN (fixed)(2018c)	CVPR 2018	32.347	0.9020	0.1397	29.262	0.8095	0.2765	27.721	0.7421	0.3637	32.903	0.9357	0.0551	28.806	0.8656	0.1394	26.619	0.8031	0.2060
SOLA-SR	This work	32.452	0.9033	**0.1117**	**29.411**	0.8130	**0.1608**	27.846	0.7468	**0.2303**	33.517	0.9406	**0.0477**	29.333	0.8758	**0.1001**	**27.165**	**0.8179**	**0.1601**

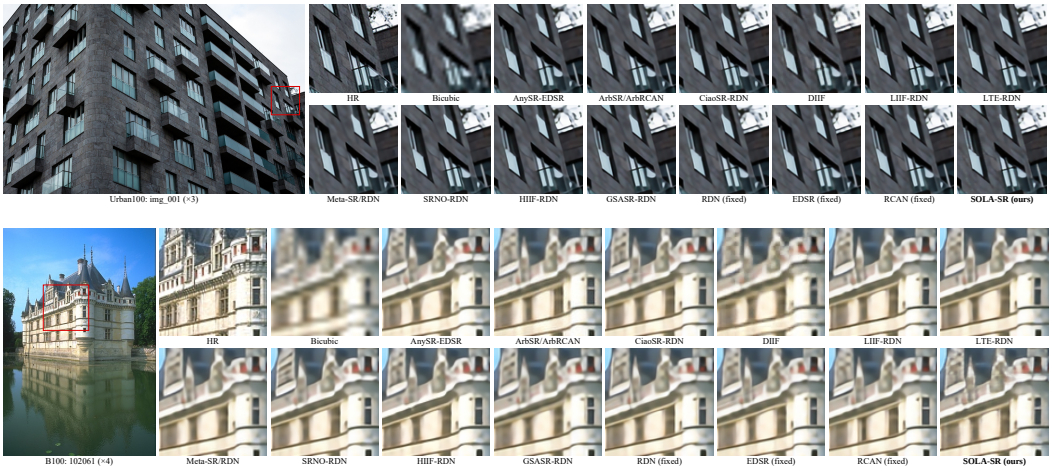

Figure 3: Integer-scale visual comparisons on Urban100 img_001 ($\times 3$) and B100 102061 ($\times 4$). Red boxes mark the compared regions.

Table 1 compares SOLA-SR with ten external ASSR methods, including AnySR (Zhan et al., 2024) and DIIF (He & Jin, 2024), and fixed-scale RDN (Zhang et al., 2018c), EDSR (Lim et al., 2017), and RCAN (Zhang et al., 2018b) controls. Figure 3 complements these quantitative results with representative reconstructions.

Across the integer scales, SOLA-SR attains PSNR on par with the strongest baselines and the lowest LPIPS. Since LPIPS is also part of the training objective (Eq. 7), this gap should be read with the PSNR results rather than as an isolated gain of the representation. The more distinctive benefits are structural: one encoding is reused across scales (Table 3) and decoding compute shrinks at lower scales, as analyzed below.

Fine-grained and extended-scale comparisons. Table 2 reports eleven methods at ten randomly sampled scales: 1.2345, 1.65, 2.3456, 2.75, 3.4567, 4.012345, 5.6789, 10.9, 20.4, and 35.2. It averages Set5, Set14, B100, and Urban100 equally at each scale; Appendix B retains all dataset-resolved results. LR/HR geometry uses MATLAB-compatible bicubic antialiasing and uint8 rounding; realized horizontal and vertical scales reflect integer output dimensions. Measurements beyond the configured range assess larger output grids, not additional latent-group release. Figure 4 shows four non-integer-scale comparisons.

Table 2: Four-dataset equal-weight mean quality at ten scales. Each scale has PSNR (Y, dB), SSIM (Y), and LPIPS-Alex subcolumns. Dark, medium, and pale red mark the first, second, and third distinct displayed values; ties share rank.

Method	Venue	×1.2345			×1.65			×2.3456			×2.75			×3.4567			×4.012345			×5.6789			×10.9			×20.4			×35.2		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Meta-SR/RDN[2019]	CVPR 2019	32.062	0.9209	0.0186	32.257	0.9228	0.0466	30.748	0.8891	0.1204	28.462	0.8454	0.1620	28.892	0.8288	0.2192	27.910	0.7989	0.2586	26.755	0.7334	0.3457	23.459	0.5969	0.5025	21.024	0.5057	0.6437	19.305	0.4678	0.7291
ArbSR–ArbRCAN[2021]	ICCV 2021	31.827	0.9170	0.0178	31.587	0.9137	0.0481	30.260	0.8802	0.1236	28.464	0.8441	0.1651	28.608	0.8251	0.2185	27.900	0.7962	0.2564	26.184	0.7192	0.3636	22.799	0.5705	0.6051	20.370	0.4890	0.7720	18.753	0.4622	0.8265
LIIF–RDN[2021]	CVPR 2021	31.814	0.9167	0.0181	31.564	0.9134	0.0489	30.230	0.8793	0.1260	28.393	0.8427	0.1676	28.514	0.8234	0.2218	27.841	0.7955	0.2585	26.485	0.7314	0.3383	23.476	0.6082	0.4976	21.149	0.5188	0.6469	19.383	0.4759	0.7496
LTE–RDN[2022]	CVPR 2022	31.799	0.9164	0.0179	31.577	0.9134	0.0485	30.244	0.8798	0.1244	28.410	0.8432	0.1658	28.548	0.8243	0.2204	27.896	0.7968	0.2579	26.517	0.7322	0.3412	23.451	0.6038	0.5112	21.043	0.5123	0.6729	19.289	0.4727	0.7757
SRNO–RDN[2023]	CVPR 2023	31.811	0.9165	0.0178	31.608	0.9139	0.0483	30.299	0.8810	0.1229	28.456	0.8445	0.1637	28.621	0.8264	0.2170	27.944	0.7989	0.2541	26.617	0.7352	0.3434	23.604	0.6120	0.5161	21.235	0.5199	0.6638	19.413	0.4748	0.7612
CiaoSR–RDN[2023]	CVPR 2023	31.340	0.9101	0.0296	30.883	0.8992	0.0574	29.984	0.8747	0.1340	28.556	0.8456	0.1665	28.580	0.8253	0.2210	27.912	0.7982	0.2566	26.676	0.7372	0.3328	23.636	0.6154	0.4872	21.198	0.5208	0.6257	19.413	0.4747	0.7116
AnySR–EDSR[2024]	TIP 2024	31.802	0.9165	0.0181	31.534	0.9130	0.0509	30.183	0.8779	0.1297	28.370	0.8411	0.1711	28.423	0.8205	0.2252	27.738	0.7922	0.2621	26.369	0.7267	0.3433	23.345	0.6032	0.4968	20.949	0.5121	0.6478	19.252	0.4704	0.7465
DIIF[2024]	ICME 2024	31.520	0.9151	0.0235	31.285	0.9111	0.0624	29.938	0.8721	0.1364	28.283	0.8363	0.1850	28.084	0.8090	0.2312	27.370	0.7782	0.2644	25.913	0.7068	0.3446	23.087	0.5786	0.5197	20.878	0.4987	0.6516	19.229	0.4658	0.7243
HIIF–RDN[2025]	CVPR 2025	31.805	0.9165	0.0180	31.604	0.9138	0.0472	30.298	0.8812	0.1219	28.448	0.8449	0.1634	28.641	0.8272	0.2172	27.986	0.7999	0.2537	26.663	0.7379	0.3315	23.618	0.6169	0.4856	21.206	0.5225	0.6338	19.348	0.4752	0.7304
GSASR–RDN[2025]	ICCV 2025	31.319	0.9062	0.0184	31.342	0.9127	0.0467	30.219	0.8855	0.1186	27.622	0.8327	0.1618	28.433	0.8241	0.2156	27.928	0.8007	0.2530	26.703	0.7402	0.3325	23.624	0.6164	0.4865	21.123	0.5123	0.6455	18.864	0.4481	0.7589
SOLA-SR	Ours	**32.236**	**0.9219**	**0.0145**	**32.437**	**0.9235**	**0.0417**	**30.958**	**0.8903**	**0.1161**	**28.680**	**0.8466**	**0.1592**	**29.072**	**0.8300**	**0.2114**	**28.142**	**0.8014**	**0.2482**	**26.940**	**0.7408**	**0.3249**	**23.848**	**0.6186**	**0.4787**	**21.393**	**0.5239**	**0.6207**	**19.558**	**0.4775**	**0.7071**

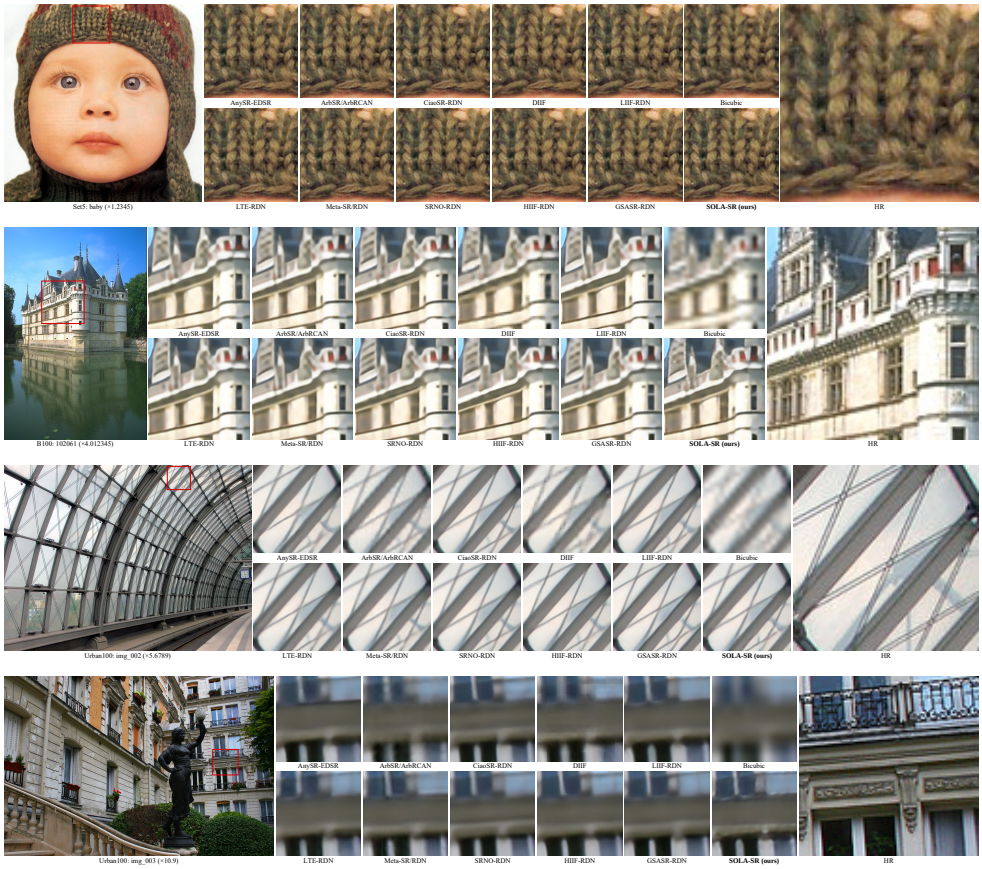

Figure 4: Non-integer-scale visual comparisons: Set5 baby (×1.2345), B100 102061 (×4.012345), Urban100 img_002 (×5.6789), and Urban100 img_003 (×10.9). Red boxes identify the shared crop; the rightmost panel enlarges the HR crop.

Continuous 200-scale inference. Table 3 compares independent encoding and encoding reuse for a 128×128 LR input at scales 2.01–4.00 (step 0.01). Timing, memory, and attribution details are in Appendix G.

Table 3: Continuous 200-scale inference ($\times 2.01$–$\times 4.00$): (a) independent encoding at every scale; (b) single-encoding reuse across scales, counted once. Dark, medium, and pale red mark the first, second, and third distinct displayed values within each panel, where lower is better; ties share rank. "-" marks an unsupported execution mode; †,‡ are unranked.

Method	Venue	Seq. (s)↓	Mean (ms)↓	P50 (ms)↓	P95 (ms)↓	Alloc. (MiB)↓	Res. (MiB)↓
Meta-SR/RDN[2019]	CVPR 2019	11.200	56.00	54.30	68.50	588.5	832.0
ArbSR–ArbRCAN[2021]	ICCV 2021	**9.613**	**48.07**	**44.81**	**51.76**	1395.3	23574.0
LIIF–RDN[2021]	CVPR 2021	19.576	97.88	95.17	135.92	364.8	646.0
LTE–RDN[2022]	CVPR 2022	20.185	100.92	98.62	140.84	**341.2**	**436.0**
SRNO–RDN[2023]	CVPR 2023	19.342	96.71	92.68	133.11	3032.4	23660.0
CiaoSR–RDN†[2023]	CVPR 2023	283.499	1417.49	1270.24	2308.54	1619.3	2332.0
DIIF[2024]	ICME 2024	10.730	53.65	53.60	56.83	1161.2	1646.0
AnySR–EDSR[2024]	TIP 2024	12.674	63.37	59.60	99.20	2954.3	23480.0
HIIF–RDN[2025]	CVPR 2025	34.802	174.01	168.35	261.45	2859.6	23688.0
GSASR–RDN‡[2025]	ICCV 2025	98.161	490.81	483.31	620.04	633.0	940.0
SOLA-SR	This work	10.920	54.60	53.10	73.80	388.5	1024.0

Method	Venue	Seq. (s)↓	Mean (ms)↓	P50 (ms)↓	P95 (ms)↓	Alloc. (MiB)↓	Res. (MiB)↓
Meta-SR/RDN[2019]	CVPR 2019	**5.100**	**25.50**	**24.20**	**38.60**	588.5	832.0
ArbSR–ArbRCAN[2021]	ICCV 2021	-	-	-	-	-	-
LIIF–RDN[2021]	CVPR 2021	12.042	60.21	57.54	97.60	364.8	646.0
LTE–RDN[2022]	CVPR 2022	12.603	63.02	60.42	102.31	**341.2**	**436.0**
SRNO–RDN[2023]	CVPR 2023	-	-	-	-	-	-
CiaoSR–RDN†[2023]	CVPR 2023	277.718	1388.59	1238.42	2278.79	1619.3	2332.0
DIIF[2024]	ICME 2024	-	-	-	-	-	-
AnySR–EDSR[2024]	TIP 2024	-	-	-	-	-	-
HIIF–RDN[2025]	CVPR 2025	27.278	136.39	130.28	223.30	2859.6	23688.0
GSASR–RDN‡[2025]	ICCV 2025	89.682	448.41	441.20	576.61	633.0	940.0
SOLA-SR	This work	9.020	45.10	43.60	73.50	388.5	1024.0

4.3 Analysis of Scale-Ordered Representation

Orthogonal basis ablation. Table 4 compares the learned basis with identity-gated and fixed-random controls on 20 development images. Three learned-basis seeds report uncertainty; the training and crop protocol is in Appendix J.

Table 4: Orthogonal-basis study; mean ± sample standard deviation across three learned-basis seeds. Dark, medium, and pale blue mark the first, second, and third distinct displayed means; ties share rank.

Configuration	RGB PSNR ↑	RGB SSIM ↑	LPIPS ↓
Identity + scale gate	24.565	0.7841	0.2107
Fixed random orthogonal + gate	21.367	0.7626	0.1771
Learned orthogonal + gate	**30.969 ± 0.005**	**0.8789 ± 0.0001**	**0.1204 ± 0.0002**

Scale contribution analysis. Table 5 reports how the opened suffix's output and gradient responses correlate with scale, together with the residual in $Q^\top Q - I$. These diagnostics do not identify a unique semantic or frequency ordering; the sweep is specified in Appendix J.

Table 5: Scale response and orthogonality.

Seed	Output–scale r	Gradient–scale r	$\max \lvert Q^\top Q - I \rvert$
101	0.9727	0.9717	2.703e-07
202	0.9705	0.9681	1.908e-07
303	0.9755	0.9651	3.441e-07

4.4 Analysis of Selective Decoding

Table 6 checks Dense–Exact numerical agreement on the same fixed weights and gates across 657 image–scale outputs; metric agreement is not a bound on per-pixel error. The endpoint hash check is described in Appendix E.

Table 6: Dense–Exact agreement.

Measurement	Observed value
Max absolute ΔRGB PSNR (dB)	3.8147e-06
Max absolute ΔRGB SSIM	2.563e-06
Max absolute ΔY PSNR (dB)	3.8147e-06
Max absolute ΔY SSIM	5.96046e-07
Max absolute ΔLPIPS-Alex	5.96046e-08
×4 identical float32 output hashes	219/219

Paired cache implementation comparison. Tables 7 and 8 compare Dense, original Exact, and revised Exact on fixed paired weights. The revision reduces Exact's median 200-scale sequence time by 0.35% independently and 0.79% with encoding reuse, but peak allocated memory remains above Dense. Stage-local peaks are nonadditive. The full protocol and contrasts are in Appendix J.

Table 7: Cache timing (s) and memory (MiB). Dark, medium, and pale red mark the first, second, and third distinct displayed values; ties share rank.

Implementation	Indep. mean	Indep. median	Reuse mean	Reuse median	Peak alloc.	Peak reserv.
Dense	19.575	19.571	12.078	12.080	**368.83**	**690.0**
Exact, original cache	18.394	18.387	10.902	10.903	392.04	1038.0
Exact, revised cache	**18.320**	**18.322**	**10.822**	**10.817**	392.04	1038.0

Table 8: Query-stage peak memory (MiB). Dark, medium, and pale red mark the first, second, and third distinct displayed values; ties share rank.

Implementation	×2.5 query peak (MiB)	×4.0 query peak (MiB)
Dense	364.68	**364.84**
Exact, original cache	**326.51**	**364.84**
Exact, revised cache	**326.51**	368.86

Execution-stage decomposition. In a separate 20-image diagnostic at ×2.5, Exact reduces median wall time from 82.718 to 73.567 ms and counted work from 501.663 to 478.952 GMAC. Six-scale timing, memory, counting scope, and the 30,000-query interface appear in Appendix H; the folded-execution check is in Appendix I.

4.5 Representation Organization vs. Latent Generation

Table 9 separates HR-conditioned oracle capacity from deployable LR-conditioned generation. Against their respective deterministic references, the oracle gains 1.289 dB PSNR and lowers LPIPS by 0.01786, whereas block-causal transport loses 0.338 dB despite lowering LPIPS by 0.00411. The 20-scene design and paired confidence intervals are in Appendix J.

Table 9: Representation capacity and LR-conditioned latent generation; separate study-specific controls. Dark, medium, and pale blue mark the first, second, and third distinct displayed values among deployable study-B PSNR/LPIPS rows; ties share rank. Oracle rows are unranked and retention columns are descriptive.

Configuration	RGB PSNR ↑	LPIPS ↓	Gradient retention	Laplacian retention
A: Deterministic SOLA-SR	32.916	0.0711	0.7214	0.7357
A: HR posterior (oracle)	34.205	0.0532	0.7740	0.7884
B: Deterministic SOLA-SR	**32.583**	0.0710	0.7290	0.7305
B: HR posterior (oracle)	33.845	0.0561	0.7801	0.7837
B: Flat latent transport	32.201	0.0670	0.7434	0.7472
B: Independent-block transport	32.153	0.0686	0.7456	0.7516
B: Block-causal transport	32.245	**0.0669**	0.7447	0.7502

5 Conclusion

SOLA-SR organizes a shared image encoding into scale-ordered latent subspaces and selects them continuously with the requested magnification. Orthogonal channel reorganization and a folded first query layer provide deterministic arbitrary-scale reconstruction without an iterative sampler. Across the evaluated settings, quality and inference-efficiency comparisons show the possibilities and limits of this design. The paired analyses further separate what the representation can retain from what an LR-conditioned generator can recover. Generalization beyond the evaluated models, image sets, and scale range remains to be established.

AI USE STATEMENT

Generative AI tools were used to aid writing, translation, and polishing; to support literature retrieval and discovery; to assist research ideation and execution, including experiment-workflow planning, code drafting, and debugging; and to draft manuscript sections. They also assisted with figure prototyping and the preparation of LaTeX and tables. Experimental measurements were obtained from model runs, not generated by AI tools. The authors take responsibility for the scientific claims, experimental evidence, software, citations, and final manuscript.

REFERENCES

Jiezhang Cao, Qin Wang, Yongqin Xian, Yawei Li, Bingbing Ni, Zhiming Pi, Kai Zhang, Yulun Zhang, Radu Timofte, and Luc Van Gool. CiaoSR: Continuous implicit attention-in-attention network for arbitrary-scale image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1796–1807, 2023. URL https://openaccess.thecvf.com/content/CVPR2023/html/Cao_CiaoSR_Continuous_Implicit_Attention-in-Attention_Network_for_Arbitrary-Scale_Image_Super-Resolution_CVPR_2023_paper.html. (Cited in Sections 1, 2.1, 1, 2, and 3.)

Du Chen, Liyi Chen, Zhengqiang Zhang, and Lei Zhang. Generalized and efficient 2D gaussian splatting for arbitrary-scale super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 26435–26445, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Chen_Generalized_and_Efficient_2D_Gaussian_Splatting_for_Arbitrary-scale_Super-Resolution_ICCV_2025_paper.html. (Cited in Sections 1, 2.1, 1, 2, and 3.)

Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8628–8638, 2021. URL https://openaccess.thecvf.com/content/CVPR2021/html/Chen_Learning_Continuous_Image_Representation_With_Local_Implicit_Image_Function_CVPR_2021_paper.html. (Cited in Sections 1, 1, 2.1, 3.1, 1, 2, 3, and G.)

Yiming Cui, Liang Li, Haibing Yin, Yuhan Gao, Xichun Sheng, and Chenggang Yan. Expert-teacher-student collaborative learning for domain adaptive object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 25557–25567, 2026. (Cited in Section 1.)

Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2): 295–307, 2016. (Cited in Section 1.)

Huiyuan Fu, Fei Peng, Xianwei Li, Yejun Li, Xin Wang, and Huadong Ma. Continuous optical zooming: A benchmark for arbitrary-scale image super-resolution in real world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3035–3044, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Fu_Continuous_Optical_Zooming_A_Benchmark_for_Arbitrary-Scale_Image_Super-Resolution_in_CVPR_2024_paper.html. (Cited in Section 2.1.)

Hongcan Guo, Qinyu Zhao, Yian Zhao, Shen Nie, Rui Zhu, Qiushan Guo, Feng Wang, Tao Yang, Hengshuang Zhao, Guoqiang Wei, and Yan Zeng. Continuous latent diffusion language model. *arXiv preprint arXiv:2605.06548*, 2026. URL https://arxiv.org/abs/2605.06548. (Cited in Section 1.)

Zongyao He and Zhi Jin. Dynamic implicit image function for efficient arbitrary-scale super-resolution. In *2024 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, 2024. doi: 10.1109/ICME57554.2024.10687852. URL https://doi.org/10.1109/ICME57554.2024.10687852. (Cited in Sections 1, 4.2, 2, and 3.)

Jintong Hu, Bin Xia, Bin Chen, Wenming Yang, and Lei Zhang. GaussianSR: High fidelity 2D gaussian splatting for arbitrary-scale image super-resolution. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(4):3554–3562, 2025. doi: 10.1609/aaai.v39i4.32369. URL https://ojs.aaai.org/index.php/AAAI/article/view/32369. (Cited in Sections 1 and 2.1.)

Xuecai Hu, Haoyuan Mu, Xiangyu Zhang, Zilei Wang, Tieniu Tan, and Jian Sun. Meta-SR: A magnification-arbitrary network for super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1575–1584, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Hu_Meta-SR_A_Magnification-Arbitrary_Network_for_Super-Resolution_CVPR_2019_paper.html. (Cited in Sections 1, 2.1, 1, 2, and 3.)

Yuxuan Jiang, Ho Man Kwan, Tianhao Peng, Ge Gao, Fan Zhang, Xiaoqing Zhu, Joel Sole, and David Bull. HIIF: Hierarchical encoding based implicit image function for continuous super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2289–2299, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Jiang_HIIF_Hierarchical_Encoding_based_Implicit_Image_Function_for_Continuous_Super-resolution_CVPR_2025_paper.html. (Cited in Sections 1, 2.1, 1, 2, and 3.)

Jaeha Kim, Junghun Oh, and Kyoung Mu Lee. Beyond image super-resolution for image recognition with task-driven perceptual loss. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2651–2661, 2024a. URL https://openaccess.thecvf.com/content/CVPR2024/html/Kim_Beyond_Image_Super-Resolution_for_Image_Recognition_with_Task-Driven_Perceptual_Loss_CVPR_2024_paper.html. (Cited in Section 1.)

Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1646–1654, 2016. (Cited in Section 1.)

Minsu Kim, Jaewon Lee, Byeonghun Lee, Sunghoon Im, and Kyong Hwan Jin. Implicit neural image stitching with enhanced and blended feature reconstruction. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 4087–4096, 2024b. URL https://openaccess.thecvf.com/content/WACV2024/html/Kim_Implicit_Neural_Image_Stitching_With_Enhanced_and_Blended_Feature_Reconstruction_WACV_2024_paper.html. (Cited in Section 2.1.)

Jaewon Lee and Kyong Hwan Jin. Local texture estimator for implicit representation function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1929–1938, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Lee_Local_Texture_Estimator_for_Implicit_Representation_Function_CVPR_2022_paper.html. (Cited in Sections 1, 2.1, 1, 2, and 3.)

Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017. (Cited in Sections 1, 1, and 4.2.)

Antoine Mercier, Ruan Erasmus, Yashesh Savani, Manik Dhingra, Fatih Porikli, and Guillaume Berger. Efficient neural supersampling on a novel gaming dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 296–306, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Mercier_Efficient_Neural_Supersampling_on_a_Novel_Gaming_Dataset_ICCV_2023_paper.html. (Cited in Section 1.)

Reyhaneh Neshatavar, Mohsen Yavartanoo, Sanghyun Son, and Kyoung Mu Lee. ICF-SRSR: Invertible scale-conditional function for self-supervised real-world single image super-resolution. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1557–1567, 2024. URL https://openaccess.thecvf.com/WACV2024. (Cited in Section 2.1.)

Daniel Panangian and Ksenia Bittner. Can location embeddings enhance super-resolution of satellite imagery? In *Proceedings of the Winter Conference on Applications of Computer Vision*, pp. 6136–6145, 2025. URL https://openaccess.thecvf.com/content/WACV2025/html/Panangian_Can_Location_Embeddings_Enhance_Super-Resolution_of_Satellite_Imagery_WACV_2025_paper.html. (Cited in Section 1.)

Sejin Park, Sangmin Lee, Kyong Hwan Jin, and Seung-Won Jung. IM-LUT: Interpolation mixing look-up tables for image super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14317–14325, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Park_IM-LUT_Interpolation_Mixing_Look-Up_Tables_for_Image_Super-Resolution_ICCV_2025_paper.html. (Cited in Section 2.1.)

Jacob Shermeyer and Adam Van Etten. The effects of super-resolution on object detection performance in satellite imagery. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019. URL https://openaccess.thecvf.com/content_CVPRW_2019/html/EarthVision/Shermeyer_The_Effects_of_Super-Resolution_on_Object_Detection_Performance_in_Satellite_CVPRW_2019_paper.html. (Cited in Section 1.)

Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Shi. 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*, pp. 1874–1883, 2016. (Cited in Section 1.)

Yi Ting Tsai, Yu Wei Chen, Hong-Han Shuai, and Ching-Chun Huang. Arbitrary-resolution and arbitrary-scale face super-resolution with implicit representation networks. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 4270–4279, 2024. URL https://openaccess.thecvf.com/content/WACV2024/html/Tsai_Arbitrary-Resolution_and_Arbitrary-Scale_Face_Super-Resolution_With_Implicit_Representation_Networks_WACV_2024_paper.html. (Cited in Section 2.1.)

Longguang Wang, Yingqian Wang, Zaiping Lin, Jungang Yang, Wei An, and Yulan Guo. Learning a single network for scale-arbitrary super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4801–4810, 2021. URL https://openaccess.thecvf.com/content/ICCV2021/html/Wang_Learning_a_Single_Network_for_Scale-Arbitrary_Super-Resolution_ICCV_2021_paper.html. (Cited in Sections 1, 2.1, 1, 2, and 3.)

Min Wei and Xuesong Zhang. Super-resolution neural operator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18247–18256, 2023. URL https://openaccess.thecvf.com/content/CVPR2023/html/Wei_Super-Resolution_Neural_Operator_CVPR_2023_paper.html. (Cited in Sections 1, 2.1, 1, 2, and 3.)

Shao-Yu Weng, Hsuan Yuan, Yu-Syuan Xu, Ching-Chun Huang, and Wei-Chen Chiu. Best of both worlds: Learning arbitrary-scale blind super-resolution via dual degradation representations and cycle-consistency. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1547–1556, 2024. URL https://openaccess.thecvf.com/content/WACV2024/html/Weng_Best_of_Both_Worlds_Learning_Arbitrary-Scale_Blind_Super-Resolution_via_Dual_WACV_2024_paper.html. (Cited in Section 2.1.)

Bartlomiej Wronski, Ignacio Garcia-Dorado, Manfred Ernst, Damien Kelly, Michael Krainin, Chia-Kai Liang, Marc Levoy, and Peyman Milanfar. Handheld multi-frame super-resolution. *ACM Transactions on Graphics*, 38, 2019. URL https://research.google/pubs/handheld-multi-frame-super-resolution/. (Cited in Section 1.)

Hanlin Wu, Ning Ni, and Libao Zhang. Scale-aware dynamic network for continuous-scale super-resolution. *arXiv preprint arXiv:2110.15655*, 2021. URL https://arxiv.org/abs/2110.15655. (Cited in Section 1.)

Wengyi Zhan, Mingbao Lin, Chia-Wen Lin, and Rongrong Ji. AnySR: Realizing image super-resolution as any-scale, any-resource. *IEEE Transactions on Image Processing*, 33:6564–6578, 2024. doi: 10.1109/TIP.2024.3497807. URL https://doi.org/10.1109/TIP.2024.3497807. (Cited in Sections 1, 4.2, 2, and 3.)

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018a. (Cited in Sections 3.6 and D.)

Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *European Conference on Computer Vision*, 2018b. (Cited in Sections 1, 1, and 4.2.)

Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2472–2481, 2018c. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_Residual_Dense_Network_CVPR_2018_paper.html. (Cited in Sections 1, 3.1, 1, and 4.2.)

A Framework Comparison

Figure 5 contrasts the source of the conceptual inspiration with the proposed image-reconstruction pipeline. COLA-DLM models continuous text latents and then generates tokens conditionally; SOLA-SR organizes one image encoding into scale-dependent channel subspaces and reconstructs pixels deterministically. The correspondence concerns representation organization, not an identity between the two generation tasks. The rightmost panels are schematic, not measured channel attribution.

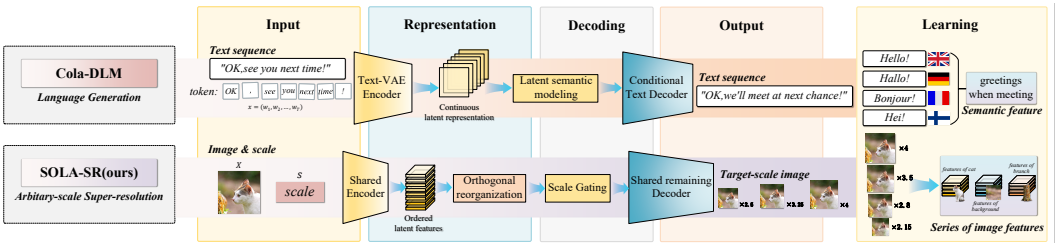

Figure 5: Conceptual comparison of COLA-DLM and SOLA-SR. Schematic panels illustrate different representation goals, not measured feature attribution.

B Dataset-Resolved Additional-Scale Results

The main text displays four-dataset equal-weight means. Table 10 retains the underlying per-dataset results in four continued blocks, without averaging across datasets.

Table 10: Additional-scale quality, block 1 of 4. Each entry averages one dataset at one requested scale; Y-PSNR (dB), Y-SSIM, and LPIPS-Alex are kept separate. Scales beyond the configured range assess extrapolation with the available latent capacity.

Method	Venue	Set5									Set14								
		×1.2345			×1.65			×2.3456			×1.2345			×1.65			×2.3456		
		Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓
Meta-SR/RDN	CVPR 2019	36.504	0.9525	0.0103	35.671	0.9448	0.0268	34.512	0.9407	0.0779	33.514	0.9340	0.0151	31.356	0.9154	0.0511	30.608	0.8770	0.1276
LIIF–RDN	CVPR 2021	36.515	0.9490	0.0083	35.050	0.9450	0.0312	34.469	0.9404	0.0797	33.510	0.9343	0.0141	31.353	0.9150	0.0496	30.524	0.8753	0.1302
ArbSR–ArbRCAN	ICCV 2021	36.502	0.9489	0.0081	35.067	0.9451	0.0309	34.485	0.9408	0.0788	33.515	0.9346	0.0138	31.387	0.9153	0.0498	30.549	0.8758	0.1314
LTE–RDN	CVPR 2022	36.511	0.9487	0.0082	35.072	0.9450	0.0309	34.458	0.9405	0.0796	33.502	0.9341	0.0136	31.360	0.9150	0.0494	30.548	0.8758	0.1292
SRNO–RDN	CVPR 2023	36.527	0.9487	0.0081	35.103	0.9454	0.0316	34.536	0.9412	0.0796	33.534	0.9345	0.0137	31.410	0.9158	0.0500	30.612	0.8773	0.1289
CiaoSR–RDN	CVPR 2023	36.460	0.9528	0.0217	34.815	0.9432	0.0333	34.365	0.9396	0.0835	33.048	0.9313	0.0174	30.462	0.9017	0.0632	30.124	0.8711	0.1431
AnySR–EDSR	TIP 2024	36.514	0.9488	0.0083	35.038	0.9450	0.0320	34.440	0.9402	0.0811	33.521	0.9345	0.0139	31.326	0.9148	0.0530	30.462	0.8738	0.1366
DIIF	ICME 2024	35.866	0.9481	0.0120	34.717	0.9446	0.0377	34.085	0.9368	0.0795	33.046	0.9309	0.0195	31.173	0.9122	0.0616	30.176	0.8680	0.1379
HIIF–RDN	CVPR 2025	36.529	0.9490	0.0083	35.079	0.9452	0.0312	34.534	0.9411	0.0788	33.567	0.9347	0.0139	31.391	0.9157	0.0497	30.628	0.8775	0.1289
GSASR–RDN	ICCV 2025	35.287	0.9295	0.0103	35.702	0.9455	0.0293	33.250	0.9354	0.0781	31.622	0.9266	0.0158	30.592	0.8991	0.0497	29.628	0.8666	0.1254
SOLA-SR	Ours	**36.684**	**0.9530**	**0.0063**	**35.747**	**0.9459**	**0.0236**	**34.748**	**0.9417**	**0.0768**	**33.647**	**0.9356**	**0.0103**	**31.548**	**0.9162**	**0.0464**	**30.685**	**0.8785**	**0.1224**

Method	Venue	B100									Urban100								
		×1.2345			×1.65			×2.3456			×1.2345			×1.65			×2.3456		
		Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓
Meta-SR/RDN	CVPR 2019	29.541	0.8916	0.0283	34.542	0.9390	0.0693	29.687	0.8525	0.1906	28.687	0.9057	0.0209	27.460	0.8922	0.0392	28.183	0.8864	0.0855
LIIF–RDN	CVPR 2021	28.980	0.8781	0.0301	34.096	0.9369	0.0713	29.154	0.8417	0.1994	28.250	0.9053	0.0198	25.757	0.8568	0.0436	26.772	0.8600	0.0949
ArbSR–ArbRCAN	ICCV 2021	28.999	0.8786	0.0299	34.147	0.9375	0.0692	29.170	0.8425	0.1922	28.293	0.9060	0.0195	25.748	0.8568	0.0426	26.837	0.8617	0.0921
LTE–RDN	CVPR 2022	28.960	0.8780	0.0300	34.124	0.9371	0.0710	29.165	0.8420	0.1957	28.224	0.9050	0.0197	25.752	0.8567	0.0428	26.807	0.8609	0.0932
SRNO–RDN	CVPR 2023	28.958	0.8779	0.0299	34.170	0.9376	0.0696	29.191	0.8428	0.1925	28.224	0.9050	0.0197	25.750	0.8569	0.0418	26.858	0.8627	0.0905
CiaoSR–RDN	CVPR 2023	28.894	0.8703	0.0499	33.778	0.9341	0.0762	29.398	0.8454	0.1977	26.958	0.8858	0.0294	24.478	0.8179	0.0569	26.049	0.8427	0.1115
AnySR–EDSR	TIP 2024	28.970	0.8782	0.0302	34.029	0.9363	0.0728	29.123	0.8406	0.2000	28.203	0.9047	0.0199	25.741	0.8560	0.0459	26.707	0.8570	0.1009
DIIF	ICME 2024	29.031	0.8780	0.0384	33.478	0.9308	0.0957	28.965	0.8353	0.2110	28.136	0.9032	0.0240	25.770	0.8567	0.0545	26.524	0.8484	0.1171
HIIF–RDN	CVPR 2025	28.935	0.8776	0.0302	34.256	0.9383	0.0666	29.184	0.8434	0.1900	28.190	0.9047	0.0197	25.689	0.8561	0.0414	26.846	0.8630	0.0899
GSASR–RDN	ICCV 2025	29.570	0.8922	0.0271	31.516	0.9140	0.0723	29.739	0.8530	0.1871	28.798	0.8767	0.0204	27.559	0.8924	0.0355	28.260	0.8868	0.0838
SOLA-SR	Ours	**29.771**	**0.8925**	**0.0247**	**34.728**	**0.9392**	**0.0628**	**29.924**	**0.8539**	**0.1843**	**28.843**	**0.9064**	**0.0168**	**27.725**	**0.8929**	**0.0341**	**28.475**	**0.8873**	**0.0810**

Dark, medium, and pale red mark the first, second, and third distinct displayed values; ties share rank. Rankings are within one dataset, scale, and metric, using displayed precision. Ten external ASSR methods and SOLA-SR form the 11-method ranking; fixed-scale controls are unranked. Training recipes differ; ranks do not establish architectural causality or statistical significance.

C Additional Visual Comparisons

These figures retain the cases moved from the main text. The integer layout contains a full HR image and red ROI at left, with the HR crop and Bicubic in the upper method row and SOLA-SR at the end of the lower row. The non-integer layout places the full HR image at left and the enlarged HR crop at right; the red ROI is shared across methods.

Table 10: Additional-scale quality (continued), block 2 of 4. Each entry averages one dataset at one requested scale; Y-PSNR (dB), Y-SSIM, and LPIPS-Alex are kept separate. Scales beyond the configured range assess extrapolation with the available latent capacity.

Method	Venue	Set5									Set14								
		×2.75			×3.4567			×4.012345			×2.75			×3.4567			×4.012345		
		Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓
Meta-SR/RDN	CVPR 2019	31.449	0.9137	0.1099	30.723	0.8961	0.1436	30.346	0.8844	0.1741	28.976	0.8415	0.1736	28.531	0.8038	0.2418	27.922	0.7788	0.2825
ArbSR–ArbRCAN	ICCV 2021	31.543	0.9143	0.1081	30.823	0.8963	0.1459	30.427	0.8824	0.1727	29.016	0.8408	0.1788	28.429	0.8025	0.2433	27.968	0.7773	0.2818
LIIF–RDN	CVPR 2021	31.472	0.9134	0.1095	30.707	0.8948	0.1480	30.316	0.8818	0.1747	28.922	0.8394	0.1772	28.332	0.8011	0.2435	27.878	0.7763	0.2823
LTE–RDN	CVPR 2022	31.477	0.9136	0.1089	30.734	0.8953	0.1472	30.400	0.8827	0.1760	28.963	0.8400	0.1764	28.347	0.8015	0.2434	27.919	0.7775	0.2825
CiaoSR–RDN	CVPR 2023	31.553	0.9144	0.1088	30.789	0.8964	0.1468	30.400	0.8837	0.1739	29.077	0.8411	0.1763	28.421	0.8034	0.2418	27.980	0.7791	0.2810
SRNO–RDN	CVPR 2023	31.527	0.9144	0.1079	30.772	0.8962	0.1458	30.367	0.8834	0.1742	29.030	0.8415	0.1764	28.437	0.8039	0.2411	27.971	0.7790	0.2803
AnySR–EDSR	TIP 2024	31.482	0.9132	0.1108	30.630	0.8934	0.1494	30.232	0.8800	0.1795	28.917	0.8382	0.1826	28.316	0.7996	0.2477	27.841	0.7745	0.2855
DIIF	ICME 2024	31.351	0.9103	0.1120	30.484	0.8880	0.1422	30.023	0.8726	0.1714	28.763	0.8331	0.1904	28.111	0.7918	0.2425	27.563	0.7640	0.2758
HIIF–RDN	CVPR 2025	31.511	0.9144	0.1082	30.743	0.8962	0.1466	30.372	0.8835	0.1741	29.027	0.8419	0.1766	28.427	0.8040	0.2419	27.993	0.7794	0.2803
GSASR–RDN	ICCV 2025	30.151	0.9015	0.1079	29.613	0.8809	0.1474	30.388	0.8849	0.1735	27.695	0.8208	0.1725	28.591	0.8045	0.2369	27.863	0.7792	0.2778
SOLA-SR	Ours	**31.742**	**0.9152**	**0.1039**	**30.977**	**0.8969**	**0.1400**	**30.645**	**0.8857**	**0.1685**	**29.221**	**0.8427**	**0.1715**	**28.669**	**0.8049**	**0.2326**	**28.156**	**0.7804**	**0.2721**

Method	Venue	B100									Urban100								
		×2.75			×3.4567			×4.012345			×2.75			×3.4567			×4.012345		
		Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓
Meta-SR/RDN	CVPR 2019	27.619	0.7946	0.2456	28.497	0.7765	0.3199	26.697	0.7277	0.3688	25.802	0.8319	0.1188	27.818	0.8389	0.1716	26.673	0.8045	0.2088
ArbSR–ArbRCAN	ICCV 2021	27.700	0.7947	0.2476	28.227	0.7738	0.3156	26.806	0.7254	0.3645	25.597	0.8268	0.1259	26.954	0.8279	0.1690	26.398	0.7995	0.2067
LIIF–RDN	CVPR 2021	27.676	0.7937	0.2543	28.190	0.7724	0.3210	26.762	0.7243	0.3682	25.501	0.8243	0.1292	26.825	0.8253	0.1747	26.407	0.7997	0.2089
LTE–RDN	CVPR 2022	27.669	0.7938	0.2503	28.212	0.7731	0.3181	26.755	0.7246	0.3667	25.531	0.8255	0.1277	26.900	0.8274	0.1729	26.511	0.8025	0.2064
CiaoSR–RDN	CVPR 2023	27.703	0.7947	0.2490	28.268	0.7749	0.3163	26.804	0.7265	0.3635	25.889	0.8323	0.1319	26.843	0.8266	0.1789	26.464	0.8033	0.2078
SRNO–RDN	CVPR 2023	27.680	0.7946	0.2463	28.249	0.7746	0.3136	26.765	0.7259	0.3614	25.585	0.8276	0.1242	27.027	0.8311	0.1677	26.672	0.8072	0.2005
AnySR–EDSR	TIP 2024	27.642	0.7924	0.2542	28.117	0.7704	0.3210	26.690	0.7222	0.3674	25.441	0.8206	0.1367	26.629	0.8184	0.1828	26.190	0.7919	0.2162
DIIF	ICME 2024	27.614	0.7890	0.2754	27.833	0.7612	0.3254	26.598	0.7145	0.3637	25.404	0.8128	0.1624	25.910	0.7951	0.2148	25.296	0.7617	0.2468
HIIF–RDN	CVPR 2025	27.670	0.7952	0.2453	28.280	0.7759	0.3138	26.784	0.7272	0.3621	25.583	0.8280	0.1236	27.115	0.8326	0.1665	26.794	0.8094	0.1982
GSASR–RDN	ICCV 2025	27.079	0.7807	0.2465	27.882	0.7700	0.3153	26.790	0.7282	0.3635	25.561	0.8279	0.1204	27.647	0.8409	0.1629	26.669	0.8107	0.1971
SOLA-SR	Ours	**27.784**	**0.7957**	**0.2442**	**28.654**	**0.7772**	**0.3122**	**26.887**	**0.7284**	**0.3594**	**25.971**	**0.8326**	**0.1170**	**27.989**	**0.8412**	**0.1607**	**26.882**	**0.8111**	**0.1928**

Dark, medium, and pale red mark the first, second, and third distinct displayed values; ties share rank. Rankings are within one dataset, scale, and metric, using displayed precision. Ten external ASSR methods and SOLA-SR form the 11-method ranking; fixed-scale controls are unranked. Training recipes differ; ranks do not establish architectural causality or statistical significance.

Table 10: Additional-scale quality (continued), block 3 of 4. Each entry averages one dataset at one requested scale; Y-PSNR (dB), Y-SSIM, and LPIPS-Alex are kept separate. Scales beyond the configured range assess extrapolation with the available latent capacity.

Method	Venue	Set5									Set14								
		×5.5			×5.6789			×10.9			×5.5			×5.6789			×10.9		
		Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓
Meta-SR/RDN	CVPR 2019	28.749	0.8433	0.2296	29.561	0.8438	0.2406	25.086	0.7035	0.3767	26.421	0.7183	0.3599	26.880	0.7125	0.3696	23.598	0.5838	0.5104
LIIF–RDN	CVPR 2021	28.605	0.8379	0.2251	28.982	0.8397	0.2336	25.132	0.7223	0.3731	26.348	0.7103	0.3566	26.616	0.7097	0.3636	23.569	0.5914	0.5105
ArbSR–ArbRCAN	ICCV 2021	28.342	0.8254	0.2444	28.507	0.8242	0.2535	24.065	0.6614	0.4958	26.275	0.7042	0.3669	26.371	0.7008	0.3791	22.972	0.5620	0.6042
LTE–RDN	CVPR 2022	28.640	0.8380	0.2293	28.960	0.8390	0.2378	25.037	0.7151	0.3870	26.352	0.7103	0.3606	26.633	0.7096	0.3692	23.599	0.5891	0.5240
CiaoSR–RDN	CVPR 2023	28.772	0.8419	0.2219	29.223	0.8442	0.2293	25.255	0.7282	0.3601	26.459	0.7141	0.3543	26.724	0.7121	0.3641	23.743	0.5973	0.5062
SRNO–RDN	CVPR 2023	28.759	0.8408	0.2328	29.131	0.8420	0.2426	25.246	0.7239	0.4000	26.415	0.7120	0.3603	26.698	0.7117	0.3693	23.702	0.5955	0.5267
AnySR–EDSR	TIP 2024	28.458	0.8340	0.2309	28.874	0.8360	0.2412	24.885	0.7131	0.3747	26.248	0.7061	0.3610	26.512	0.7055	0.3705	23.515	0.5887	0.5110
DIIF	ICME 2024	28.118	0.8176	0.2238	28.304	0.8160	0.2324	24.577	0.6763	0.4099	26.056	0.6947	0.3484	26.142	0.6920	0.3556	23.268	0.5700	0.5158
GSASR–RDN	ICCV 2025	28.700	0.8440	0.2213	29.361	0.8478	0.2267	25.112	0.7247	0.3550	25.964	0.7076	0.3549	26.589	0.7129	0.3624	23.726	0.5974	0.5094
HIIF–RDN	CVPR 2025	28.689	0.8408	0.2244	29.085	0.8436	0.2313	25.218	0.7308	0.3633	26.439	0.7140	0.3536	26.762	0.7133	0.3611	23.716	0.5982	0.5028
SOLA-SR	Ours	**28.970**	**0.8445**	**0.2181**	**29.673**	**0.8482**	**0.2222**	**25.474**	**0.7315**	**0.3506**	**26.570**	**0.7193**	**0.3459**	**27.012**	**0.7136**	**0.3546**	**23.938**	**0.5984**	**0.4993**

Method	Venue	B100									Urban100								
		×5.5			×5.6789			×10.9			×5.5			×5.6789			×10.9		
		Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓
Meta-SR/RDN	CVPR 2019	26.297	0.6709	0.4571	26.163	0.6643	0.4666	23.792	0.5499	0.6184	23.670	0.7117	0.2955	24.416	0.7130	0.3059	21.358	0.5505	0.5045
LIIF–RDN	CVPR 2021	25.823	0.6636	0.4516	26.187	0.6663	0.4594	23.865	0.5567	0.6171	23.581	0.7015	0.2902	24.156	0.7098	0.2966	21.337	0.5625	0.4897
ArbSR–ArbRCAN	ICCV 2021	25.789	0.6587	0.4620	25.994	0.6589	0.4723	23.428	0.5368	0.6862	23.601	0.6914	0.3360	23.863	0.6930	0.3494	20.732	0.5216	0.6343
LTE–RDN	CVPR 2022	25.827	0.6639	0.4549	26.219	0.6668	0.4635	23.850	0.5542	0.6305	23.627	0.7042	0.2882	24.256	0.7133	0.2944	21.320	0.5570	0.5035
CiaoSR–RDN	CVPR 2023	25.877	0.6665	0.4484	26.281	0.6699	0.4561	23.941	0.5600	0.6144	23.772	0.7120	0.2760	24.475	0.7226	0.2819	21.603	0.5759	0.4680
SRNO–RDN	CVPR 2023	25.867	0.6658	0.4532	26.259	0.6690	0.4612	23.932	0.5587	0.6316	23.749	0.7092	0.2932	24.378	0.7180	0.3004	21.535	0.5700	0.5059
AnySR–EDSR	TIP 2024	25.742	0.6610	0.4527	26.096	0.6635	0.4605	23.804	0.5548	0.6172	23.439	0.6938	0.2948	23.995	0.7016	0.3011	21.175	0.5561	0.4843
DIIF	ICME 2024	25.608	0.6504	0.4398	25.819	0.6505	0.4486	23.627	0.5420	0.6048	23.175	0.6681	0.3329	23.387	0.6687	0.3418	20.878	0.5260	0.5481
GSASR–RDN	ICCV 2025	25.713	0.6650	0.4510	26.261	0.6714	0.4581	23.959	0.5624	0.6165	23.603	0.7098	0.2771	24.600	0.7289	0.2827	21.701	0.5810	0.4652
HIIF–RDN	CVPR 2025	25.865	0.6674	0.4463	26.299	0.6711	0.4535	23.953	0.5615	0.6117	23.770	0.7124	0.2748	24.508	0.7238	0.2800	21.586	0.5771	0.4644
SOLA-SR	Ours	**26.385**	**0.6713**	**0.4369**	**26.359**	**0.6722**	**0.4449**	**24.122**	**0.5631**	**0.6030**	**23.859**	**0.7131**	**0.2707**	**24.716**	**0.7292**	**0.2778**	**21.856**	**0.5813**	**0.4620**

Dark, medium, and pale red mark the first, second, and third distinct displayed values; ties share rank. Rankings are within one dataset, scale, and metric, using displayed precision. Ten external ASSR methods and SOLA-SR form the 11-method ranking; fixed-scale controls are unranked. Training recipes differ; ranks do not establish architectural causality or statistical significance.

Table 10: Additional-scale quality (continued), block 4 of 4. Each entry averages one dataset at one requested scale; Y-PSNR (dB), Y-SSIM, and LPIPS-Alex are kept separate. Scales beyond the configured range assess extrapolation with the available latent capacity.

Method	Venue	Set5						Set14					
		×20.4			×35.2			×20.4			×35.2		
		Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓
Meta-SR/RDN	CVPR 2019	21.927	0.5857	0.5495	19.996	0.5598	0.6509	21.056	0.4965	0.6290	19.185	0.4487	0.7144
LIIF–RDN	CVPR 2021	22.164	0.6028	0.5396	20.063	0.5681	0.6640	21.139	0.5080	0.6393	19.243	0.4566	0.7352
ArbSR–ArbRCAN	ICCV 2021	20.963	0.5585	0.6820	19.220	0.5516	0.7491	20.437	0.4809	0.7582	18.687	0.4422	0.8090
LTE–RDN	CVPR 2022	21.969	0.5947	0.5684	19.928	0.5652	0.6934	21.076	0.5023	0.6622	19.178	0.4528	0.7581
SRNO–RDN	CVPR 2023	22.234	0.6040	0.5582	20.077	0.5659	0.6709	21.233	0.5089	0.6510	19.260	0.4559	0.7480
CiaoSR–RDN	CVPR 2023	22.103	0.6001	0.5147	20.041	0.5633	0.6270	21.190	0.5100	0.6174	19.262	0.4549	0.6970
AnySR–EDSR	TIP 2024	21.859	0.5880	0.5632	19.994	0.5595	0.6718	20.938	0.5032	0.6309	19.158	0.4522	0.7291
DIIF	ICME 2024	21.773	0.5743	0.5700	19.898	0.5573	0.6607	20.936	0.4898	0.6427	19.147	0.4457	0.7099
HIIF–RDN	CVPR 2025	22.141	0.6022	0.5275	19.892	0.5622	0.6356	21.211	0.5125	0.6284	19.198	0.4559	0.7192
GSASR–RDN	ICCV 2025	21.769	0.5644	0.5648	19.390	0.5171	0.7050	21.201	0.5097	0.6309	18.573	0.4353	0.7482
SOLA-SR	Ours	**22.356**	**0.6051**	**0.5107**	**20.254**	**0.5688**	**0.6233**	**21.320**	**0.5132**	**0.6159**	**19.397**	**0.4568**	**0.6950**

Method	Venue	B100						Urban100					
		×20.4			×35.2			×20.4			×35.2		
		Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓	Y-PSNR↑	Y-SSIM↑	LPIPS↓
Meta-SR/RDN	CVPR 2019	21.795	0.4846	0.7208	20.082	0.4445	0.7807	19.319	0.4560	0.6753	17.957	0.4183	0.7705
LIIF–RDN	CVPR 2021	21.880	0.4935	0.7295	20.175	0.4512	0.8016	19.415	0.4710	0.6791	18.051	0.4278	0.7975
ArbSR–ArbRCAN	ICCV 2021	21.328	0.4749	0.8221	19.601	0.4401	0.8601	18.750	0.4418	0.8255	17.504	0.4147	0.8877
LTE–RDN	CVPR 2022	21.843	0.4895	0.7502	20.107	0.4484	0.8215	19.285	0.4626	0.7109	17.943	0.4245	0.8297
SRNO–RDN	CVPR 2023	21.914	0.4930	0.7483	20.172	0.4496	0.8141	19.558	0.4738	0.6978	18.141	0.4276	0.8118
CiaoSR–RDN	CVPR 2023	21.919	0.4947	0.7175	20.192	0.4504	0.7688	19.579	0.4784	0.6532	18.156	0.4300	0.7535
AnySR–EDSR	TIP 2024	21.741	0.4901	0.7298	19.958	0.4458	0.7993	19.258	0.4669	0.6672	17.898	0.4240	0.7858
DIIF	ICME 2024	21.709	0.4814	0.7064	20.042	0.4429	0.7616	19.095	0.4493	0.6873	17.828	0.4172	0.7651
HIIF–RDN	CVPR 2025	21.934	0.4963	0.7244	20.174	0.4517	0.7942	19.536	0.4791	0.6550	18.129	0.4311	0.7725
GSASR–RDN	ICCV 2025	21.917	0.4961	0.7278	19.378	0.4106	0.7947	19.604	0.4789	0.6587	18.113	0.4294	0.7878
SOLA-SR	Ours	**22.138**	**0.4973**	**0.7045**	**20.236**	**0.4527**	**0.7575**	**19.759**	**0.4798**	**0.6518**	**18.346**	**0.4316**	**0.7525**

Dark, medium, and pale red mark the first, second, and third distinct displayed values; ties share rank. Rankings are within one dataset, scale, and metric, using displayed precision. Ten external ASSR methods and SOLA-SR form the 11-method ranking; fixed-scale controls are unranked. Training recipes differ; ranks do not establish architectural causality or statistical significance.

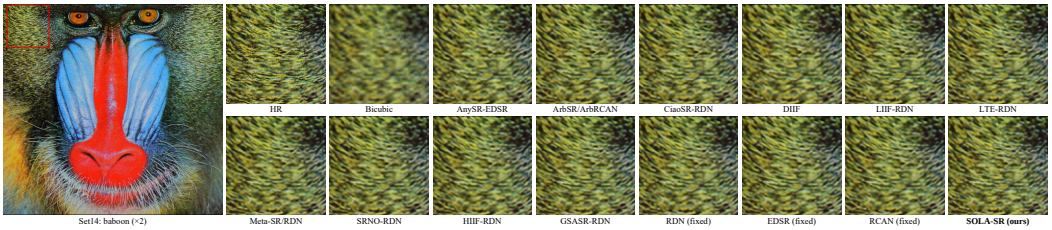

Figure 6: Additional integer-scale comparison: Set14 baboon (×2).

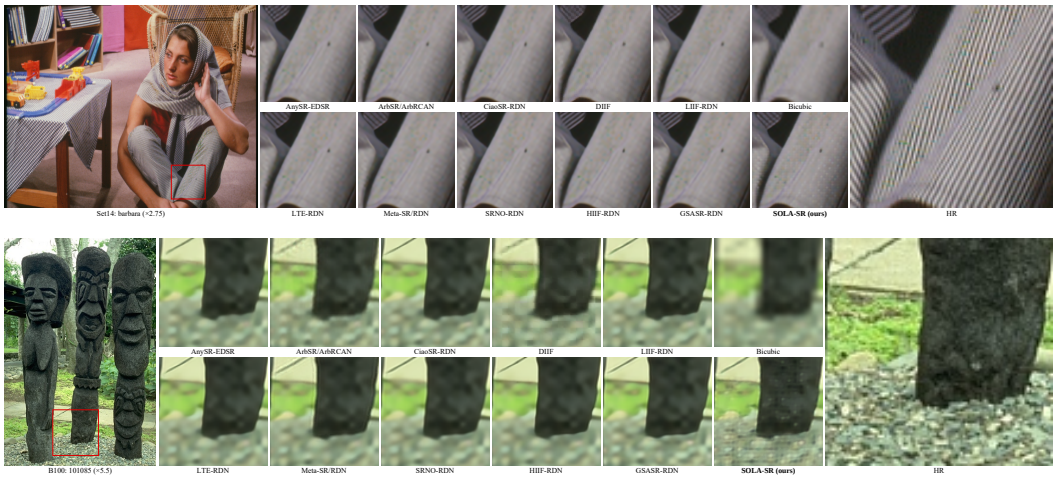

Figure 7: Additional non-integer comparisons: Set14 barbara (×2.75) and B100 101085 (×5.5).

D METHOD NOTATION AND DERIVATION DETAILS

Coordinate-query formulation. Arbitrary-scale SR first extracts spatial latent features and then decodes RGB values at continuous target coordinates. For LR image x and magnification s, the encoder produces $z = E(x) \in \mathbb{R}^{H\times W\times C}$. A query q identifies an output pixel and $c(s)$ its cell size; the decoder predicts $\hat{y}_s(q) = D(z, q, c(s))$. The output grid determines image dimensions, whereas scale-dependent selection determines which latent dimensions reach each query. Our RDN–LIIF instantiation retains the original spatial layout, relative coordinates, cell decoding, feature unfolding, and local ensemble.

Channel reorganization and gating. At each spatial location the Cayley-parameterized T mixes channels without mixing locations. It starts at the identity, preserves the norm and invertibility of the full representation, and is optimized through reconstruction rather than an independently estimated channel-importance score. A fixed clipped schedule releases G groups between s_0 and s_1; partially weighted groups remain in the active support. Hence continuously varying gate weights do not imply a continuous-valued number of input channels, and changing the output grid is independent of releasing another group.

Selection identity and tensor ordering. Because S_s contains the identity rows at every nonzero gate entry, $S_s^\top S_s M(s) = M(s)$, including partially activated groups. With $B_s = T^\top S_s^\top$, this yields

$$B_s u_s(p) = T^\top S_s^\top S_s M(s) T z(p) = T^\top M(s) T z(p) = z_s(p).$$

For the nine-neighbor query, $\bar{z}_s = [z_s(p_1)^\top, \dots, z_s(p_9)^\top]^\top \in \mathbb{R}^{9C}$ and $\bar{u}_s = [u_s(p_1)^\top, \dots, u_s(p_9)^\top]^\top \in \mathbb{R}^{9k_s}$. The Kronecker expression in Equation 6 is neighborhood-major: $\bar{z}_s = (I_9 \otimes B_s)\bar{u}_s$. If P_C and P_{k_s} permute the respective vectors to channel-major order, $P_C(I_9 \otimes B_s)P_{k_s}^\top = B_s \otimes I_9$. Feature buffers and weight columns must use the same ordering.

First-layer execution. Let $v \in \mathbb{R}^{d_v}$ collect relative-coordinate and cell inputs ($d_v = 4$ when both are enabled), $W_v \in \mathbb{R}^{d\times d_v}$, and $b \in \mathbb{R}^d$. The activation is $h = \sigma(W_{f,s}\bar{u}_s + W_v v + b)$. The coordinate/cell weights, bias, activation, later layers, and local ensemble are unchanged. Exact processes $9k_s$ feature values rather than $9C$ and never reconstructs a zero-padded full-width query input. Dense and Exact are algebraically equivalent for fixed trained weights and identical gates, not separately trained networks; different floating-point kernels need not give bitwise-equal outputs. Appendix F distinguishes feature-side and weight-side gate placement.

Computation scope. If N_s is the output-pixel count and L the local-ensemble queries per pixel, the first-layer dense and selective multiplications cost $LN_s d(9C + d_v)$ and $LN_s d(9k_s + d_v)$ MACs. Their difference, $9LN_s d(C - k_s)$, concerns the first layer only. Basis construction, transformation, folded-weight preparation, sampling, and memory traffic enter end-to-end execution; this first-layer saving vanishes at $k_s = C$. Table 15 separates these costs and their reuse scopes.

Training terms. For a training pair (x, y_s), the dense gated path predicts $\hat{y}_s$. The Charbonnier term averages $\sqrt{(\hat{y}_s - y_s)^2 + \epsilon^2}$ over RGB pixels ($\epsilon > 0$). LPIPS uses a frozen AlexNet (Zhang et al., 2018a), while color and gradient losses are ℓ_1 differences of per-channel spatial means and Sobel responses. Gradients pass through the dense readout to the encoder, decoder, and Cayley parameters. The fixed scale-release schedule is not trained; inference uses the trained weights directly or folds them algebraically.

E GATE CONFIGURATION

The implementation provides a fully open gate at $\times 4$. The reported 64-channel configuration uses $C_0 = 32$, $G = 4$, $C_g = 8$, $s_0 = 2$, and $s_1 = 4$, giving $r(s) = \mathrm{clip}(2(s-2), 0, 4)$. Partially weighted groups count as active: all 64 dimensions are present for $s > 3.5$, whereas all gate weights reach one at $s = 4$. At $\times 4$, the code uses the encoder feature z directly instead of numerically forming $T^\top T z$. Larger output grids do not release additional latent groups beyond the fully open gate. This endpoint is an implementation setting within arbitrary-scale reconstruction, not a separate fixed-scale method.

F Gate Placement and Folded-Weight Implementation

Equation 6 applies the gate to selected features. To relate it to the cached implementation, define ungated selected features $\tilde{u}_s = S_s T z$ and the restricted gate $\Lambda_s = S_s M(s) S_s^\top \in \mathbb{R}^{k_s \times k_s}$. Then $u_s = \Lambda_s \tilde{u}_s$. In the neighborhood-major convention of the equation,

$$\widetilde{W}_{f,s} = W_{f,s}(I_9 \otimes \Lambda_s) = W_f(I_9 \otimes B_s \Lambda_s), \qquad W_{f,s}\bar{u}_s = \widetilde{W}_{f,s}\bar{\tilde{u}}_s.$$

The gate is applied once, either to the features or to the weights. The implementation uses channel-major feature unfolding, where the corresponding operator is $(B_s\Lambda_s) \otimes I_9$. For an output-channel index o, active ordered-basis channel index i retained in full-basis order, and neighborhood offset p, its contraction is

$$\widetilde{W}_{o,i,p} = m_i(s) \sum_{c=1}^{C} W_{o,c,p} T_{i,c},$$

with active indices retained in their original order. Relative-coordinate and cell columns and the bias are not transformed. This contraction avoids constructing the full Kronecker matrix.

The cached path retains the fixed model basis and, for each input, its encoded and transformed features. For each requested scale below four, it prepares the compact unfolded features and gate-weighted first-layer block, rather than retaining a bank of scale-specific folded blocks. At scales of four or greater, the query branch uses the original encoder feature without folding. The encoding routine nevertheless computes Tz before this branch is selected; that cost is not eliminated by the query bypass. Reusing a folded block across requests would require unchanged gate values as well as unchanged model weights, basis, active-index order, tensor layout, device and precision. Thus the support-level reuse permitted by the feature-gated equation is not evidence of that reuse in this implementation. Stage timing must include the preparation actually performed by the selected execution path.

G Continuous-Scale Timing Protocol and Attribution

Sequence and memory measurements. Table 3 reconstructs a fixed 128×128 LR image at 200 consecutive scales, $s = 2.01, 2.02, \ldots, 4.00$. Independent execution repeats encoding and decoding at every scale; cached execution reuses one image encoding when the native interface supports it, including that encoding cost once. The reference timing protocol specifies three fresh-process repeats and reports mean sequence time, frame-level P50/P95 over 600 measurements, and peak allocated/reserved memory. The cached mean includes one encoder call, whereas frame quantiles describe the scale-dependent forward calls. Allocated memory denotes live CUDA allocations; reserved memory denotes the allocator pool, not the model's required live memory. Training recipes and backbones remain method-specific, and rankings are descriptive orderings of the displayed measurements.

Attribution of execution savings. Encoding reuse, intermediate-buffer reuse, and selective query decoding provide distinct execution savings. LIIF already separates feature extraction from coordinate queries (Chen et al., 2021), so encoding reuse is not specific to SOLA-SR. Paired fixed-weight diagnostics distinguish these effects under corresponding execution modes. For 200-scale sequences, Exact used 0.616 s less than Dense with independent encoding and 1.241 s less with encoding reuse, averaged over 20 images. Additional cache preparation contributed separate reductions; these descriptive differences are not additive causal contributions. Appendix H provides the full contrasts and measurement conditions. Reduced first-layer MACs need not translate proportionally to sequence time or peak memory.

Scope of timing ranks. CiaoSR and GSASR retain their measured timing values but are excluded from timing ranks. This restriction does not remove their reconstruction-quality results. All 12 recorded CiaoSR repeats, including the cold-start follow-up, reported software thermal slowdown. Peak GPU core temperature was 88–89 °C; no hardware thermal-slowdown flag was recorded, and hotspot and memory temperatures were not collected. The available observations do not establish an unrestricted timing comparison.

For GSASR, stage-hash checks at $\times 2.5$, $\times 3.5$, and $\times 4.0$ in both tracks found identical inputs, padded inputs, encoded features, and independently evaluated semantic-head parameters across three fresh processes, but different output hashes in all six track–scale groups. This localizes divergence after the recorded feature/head witnesses to the rendering/output path; it does not establish a particular CUDA operator as its cause. The outputs were finite, but the predeclared bitwise-repeatability criterion remained unmet. These observations qualify timing comparability and are not evidence that either reconstruction method fails.

H Paired Runtime and Computation Diagnostics

The additional diagnostics use fixed SOLA-SR weights and 20 DIV2K development images (training IDs 0781–0800). Each image supplies a 128×128 LR input; execution uses RTX 3090, FP32, TF32 disabled, and query chunks of 30,000. Each configuration retains one first-complete process run, with image-level pairing rather than treating the 200 scales as independent samples. Configurations run in a fixed order. Software thermal slowdown was recorded during these measurements, so the timing contrasts are descriptive implementation observations rather than unrestricted causal speed estimates.

H.1 Encoding, caching, and selective execution

Table 11 separates repeated encoding from encode-once execution, Dense from Exact, and additional cache preparation within each path. LIIF provides an external encoding-reuse reference. Differences are paired by input image and reported as mean sequence-time reductions; these contrasts are not additive components of a single speedup.

Table 11: Paired 200-scale sequence-time differences on 20 development images. Positive values mean the first configuration takes longer.

Execution contrast	Mean difference (s)
Dense: repeated encoding − encode once	7.514
Exact: repeated encoding − encode once	8.140
LIIF: repeated encoding − encode once	8.179
Dense − Exact, independent encoding	0.616
Dense − Exact, encoding reuse	1.241
Dense: original − prepared cache	0.747
Exact: original − revised cache	0.137

Scales 2.01–4.00, FP32, query chunk 30,000; one first-complete run per configuration. Measurement conditions are specified in Appendix H.

H.2 Stage timing, counted work, and memory

Tables 12 and 13 give medians over the 20 images at each scale. End-to-end wall time is measured without stage instrumentation; stage CUDA times are synchronized diagnostics and are not additive. The memory columns give cell-scoped peak allocated and reserved memory, not full 200-scale-sequence peaks. Encoder and later-query Conv/Linear MACs use actual-shape forward hooks; first-affine, transform, and folded-weight MACs use the executed tensor dimensions. Counted work excludes basis construction, sampling, indexing, coordinate arithmetic, ensemble weighting, scalar gates, and memory traffic. Exact reduced allocated memory at the measured scales below four, but neither allocated nor reserved memory was uniformly lower across all six scales.

I Runtime Check of Folded Execution

We test Equation 6 with fixed weights, a fixed seeded 32×32 FP32 input (seed 20260924), and five specified scales. A one-hot layout check gives zero error between channel-major and neighborhood-major unfolding after the explicit permutation in Section F. Across the four non-endpoint scales, the maximum first-layer preactivation error is 4.77×10^{-7}. The maximum Dense/Exact output error is 1.73×10^{-6}, below the preset 10^{-5} tolerance. Revised and original Exact outputs are bitwise equal at all five scales. Table 14 records each check; it is an implementation witness for this fixture, not a reconstruction-quality test or a cross-platform empirical guarantee.

Table 12: Runtime decomposition at six scales; medians across 20 development images, in milliseconds.

Path	Scale	E2E wall	Encoder	Transform	Basis	Fold	First affine	Later query
Dense	2	67.183	37.382	0.613	–	–	6.775	14.910
Dense	2.25	74.255	38.510	0.486	–	–	7.904	17.863
Dense	2.5	82.718	39.192	0.492	–	–	9.989	22.186
Dense	3	100.994	39.134	0.496	–	–	13.787	31.072
Dense	3.5	123.192	39.066	0.494	–	–	18.760	42.031
Dense	4	147.593	39.056	–	–	–	24.236	54.602
Exact	2	59.694	37.066	0.114	0.418	0.182	4.996	14.816
Exact	2.25	66.706	37.995	0.093	0.300	0.156	6.228	17.821
Exact	2.5	73.567	38.871	0.095	0.306	0.155	7.818	22.249
Exact	3	91.947	38.380	0.095	0.309	0.157	11.900	30.946
Exact	3.5	116.493	39.013	0.094	0.310	0.157	17.400	41.936
Exact	4	146.740	38.879	–	–	–	24.083	54.477

E2E is measured without instrumentation; stage CUDA times use synchronization and are not additive. A dash indicates the stage is bypassed. See Appendix H for the separate first-complete protocol.

Table 13: Counted computation and peak memory for the same six-scale diagnostic.

Path	Scale	Encoder	Transform	Fold	First affine	Later query	Total	Allocated	Reserved
Dense	2	359.866	0.134	0.000000	38.923	51.741	450.664	397.42	606.00
Dense	2.25	359.866	0.134	0.000000	49.262	65.485	474.747	397.68	676.00
Dense	2.5	359.866	0.134	0.000000	60.817	80.845	501.663	398.32	750.00
Dense	3	359.866	0.134	0.000000	87.577	116.417	563.994	400.23	750.00
Dense	3.5	359.866	0.134	0.000000	119.202	158.457	637.659	401.32	750.00
Dense	4	359.866	0.000	0.000000	155.693	206.964	722.522	398.48	750.00
Exact	2	359.866	0.067	0.004719	19.596	51.741	431.275	282.76	422.00
Exact	2.25	359.866	0.067	0.005898	30.916	65.485	456.340	313.96	510.00
Exact	2.5	359.866	0.067	0.005898	38.168	80.845	478.952	315.30	560.00
Exact	3	359.866	0.067	0.007078	65.834	116.417	542.191	347.52	660.00
Exact	3.5	359.866	0.067	0.008258	104.405	158.457	622.803	377.24	830.00
Exact	4	359.866	0.000	0.000000	155.693	206.964	722.522	399.18	1032.00

Work columns are GMAC; the last two columns are cell-scoped peak MiB. Encoder/later Conv/Linear counts use actual-shape hooks; transform/fold/first-affine counts use tensor dimensions. Excluded operations and non-additive memory scopes are stated in Appendix H.

Table 14: Fixed-fixture runtime check of folded execution.

Scale	k_s	First-affine max error	Output max error	Exact cache bitwise
2.25	40	4.768e-07	6.258e-07	Yes
2.75	48	4.768e-07	9.537e-07	Yes
3.25	56	4.768e-07	8.345e-07	Yes
3.75	64	4.768e-07	1.729e-06	Yes
4.0	64	–	0.000e+00	Yes

Maximum absolute errors compare Dense and Exact. The last column compares revised and original Exact. At $\times 4$, the endpoint bypass is recorded separately; all active dimensions are already present at $\times 3.75$.

J Extended Experimental Details

Orthogonal-basis protocol. The basis study freezes a pretrained LIIF encoder and decoder (1,567,323 parameters) and optimizes only 2,016 Cayley coordinates for 3,000 steps with AdamW, learning rate 0.01, zero weight decay, and batch size 16. The learned basis is repeated with seeds 101, 202, and 303; fixed controls are reused. Evaluation uses 20 development images (DIV2K training IDs 0781–0800), 192×192 center patches, and eight off-grid scales $\{1.5, 1.75, 2.25, 2.5, 2.75, 3.25, 3.5, 3.75\}$.

Scale-contribution sweep. For each learned basis, 21 scales from 2.0 to 4.0 in increments of 0.1 measure output contribution and gradient response of the progressively opened suffix. Table 5 reports Pearson correlations and the maximum elementwise error in $Q^\top Q - I$. These measurements describe scale-dependent use, not a unique semantic or frequency ordering.

Dense–Exact equivalence. The equivalence study uses identical weights and scale gates from fixed paired weights. We pair 657 public image–scale outputs and also check float32-output hashes for

the endpoint path described in Appendix E. Table 6 reports numerical agreement for these weights. Metric agreement is not a per-pixel error bound.

Paired cache contrasts. Original Dense and Exact implementations and a revised Exact cache use the same fixed paired weights and 200-scale sequence, with three fresh-process repeats per implementation and execution track; there is no retraining. Revised Exact reduces median sequence time from 18.386882 to 18.322304 s for independent execution and from 10.902903 to 10.816993 s with encoding reuse (0.35% and 0.79%, respectively). These differences isolate this cache revision within Exact, not the entire benefit of selective decoding or a reconstruction-quality gain. Peak allocated memory remains above Dense in the 200-scale measurement.

Stage-local memory. Instrumented stage peaks are not additive and do not replace the full 200-frame memory measurements. Revised Exact uses less query-stage memory than Dense at $\times 2.5$ but more at $\times 4.0$. It retains the encoder feature and ordered representation simultaneously; older implementations did not expose all corresponding substages separately. Reduced input computation therefore does not imply universally lower peak VRAM.

Representation and generation protocol. Study A supplies an HR-conditioned posterior to a frozen deterministic reconstruction model, measuring what the latent representation can encode; the posterior is an oracle diagnostic, not an inference-time method. Study B compares flat, independent-block, and block-causal LR-conditioned latent transports, each with 57,808 parameters, 20,000 training steps, and eight sampling evaluations. Evaluation uses 20 development scenes (DIV2K training IDs 0761–0780), three fixed crop locations per scene, and nine scales from 2.0 to 4.0 in steps of 0.25, yielding 540 measurements per method. The prior study uses 192×192 HR patches, 48×48 LR patches, and 64×64 query crops. Statistics are recomputed from per-crop measurements; posterior and deterministic rows remain distinct references.

For the HR-conditioned posterior relative to its deterministic reconstruction reference, the image-level paired RGB PSNR difference is +1.289 dB (95% bootstrap CI: +1.079 to +1.492), and the LPIPS difference is -0.01786 (-0.02549 to -0.01141). For block-causal LR-conditioned transport relative to its deterministic reference, the corresponding differences are -0.338 dB (-0.455 to -0.251) and -0.00411 (-0.00649 to -0.00196). Confidence intervals use 2,000 image-level bootstrap draws with crops and scales nested within each image.

K Analytic Cost and Reference Protocol

The following tables retain the analytic computation accounting, reference measurement settings, and public sample counts from the original manuscript, documenting the measurement protocol for each study.

Table 15: Analytic computation accounting for Dense and Exact. These expressions are not measured runtime results. H, W, C describe the encoder feature map, N_s the output pixel count, L the local-ensemble multiplicity, and d the first query-layer width. One MAC denotes one multiply–accumulate.

Stage	MAC accounting	Reuse / execution scope
Encoder	Backbone-dependent; unchanged between paired paths	Once per image with encoding reuse; otherwise once per scale
Basis construction	Matrix-solve implementation-dependent	Model-static for fixed weights; cold preparation must be counted separately
Forward channel transform	HWC^2 for a dense Tz	Once per encoded feature in the cached path; still computed but unused by its fully open query branch
Dense back-projection	HWC^2 for a dense $T^\top g$	Scale-dependent; absent in Exact; full-open bypass avoids it
Exact folded feature weights	$9dCk_s$ for nine direct block products	Support-level reuse for feature-side gating; per-request preparation when the weight-gated folded path is used
First query layer: Dense	$LN_s d(9C + d_v)$	Per output grid
First query layer: Exact	$LN_s d(9k_s + d_v)$	Per output grid
Remaining query layers	$LN_s \sum_{j=1}^{m} d_{j-1} d_j$	Same widths in paired paths; $d_0 = d$, $d_m = 3$
Gate, selection, neighborhood preparation and output assembly	Not included in these dense-matrix MAC expressions	Arithmetic and memory costs require separate timing; not treated as zero-cost

The transform and folding expressions describe the stated direct operations, without materializing a dense Kronecker matrix. Caching and bypasses determine their execution counts. Folding the gate into weights additionally requires scalar weighting; support alone is not a sufficient reuse key when fractional gate values change. These scalar operations and memory traffic are outside the listed matrix MACs, not zero-cost operations. Peak memory values are not additive across stages; cache storage must be included in the end-to-end peak. MAC reductions alone do not establish a wall-clock improvement.

Table 16: Reference measurement protocol and settings.

Item	Setting
Hardware	NVIDIA RTX 3090, 24 GB; single-GPU evaluation
Main scales	$\times 2$, $\times 3$, $\times 4$
Degradation	Frozen bicubic LR/HR pairs; identical input hashes across methods
Quality	RGB and Y PSNR/SSIM; LPIPS-Alex; scale-dependent border shaving
Continuous efficiency	FP32; TF32 off; batch 1; 128×128 LR input
Timing	20 warm-ups; 3 fresh-process repeats; 200 scales per repeat
Timing scope	Synchronized forward execution; no file I/O, hashing, or telemetry inside timing
Memory	CUDA peak allocated and peak reserved reported separately

Reference measurement protocol. Models and training recipes are method-specific; mechanism studies use their own stated settings.

Table 17: Public test-image counts and measurement units.

Dataset	Images	Main pairs per method	Extended pairs per method
Set5	5	15	20
Set14	14	42	56
B100 (BSD100)	100	300	400
Urban100	100	300	400
Total	219	657	876

Main: three scales. Extended: four scales. Image-weighted aggregation gives 657 and 876 pairs per method, respectively.