Super-Resolution Neural Operator

Min Wei*  Xuesong Zhang*†
Beijing University of Posts and Telecommunications, China
{mw, xuesong_zhang}@bupt.edu.cn

Abstract

We propose Super-resolution Neural Operator (SRNO), a deep operator learning framework that can resolve high-resolution (HR) images at arbitrary scales from the low-resolution (LR) counterparts. Treating the LR-HR image pairs as continuous functions approximated with different grid sizes, SRNO learns the mapping between the corresponding function spaces. From the perspective of approximation theory, SRNO first embeds the LR input into a higher-dimensional latent representation space, trying to capture sufficient basis functions, and then iteratively approximates the implicit image function with a kernel integral mechanism, followed by a final dimensionality reduction step to generate the RGB representation at the target coordinates. The key characteristics distinguishing SRNO from prior continuous SR works are: 1) the kernel integral in each layer is efficiently implemented via the Galerkin-type attention, which possesses non-local properties in the spatial domain and therefore benefits the grid-free continuum; and 2) the multilayer attention architecture allows for the dynamic latent basis update, which is crucial for SR problems to “hallucinate” high-frequency information from the LR image. Experiments show that SRNO outperforms existing continuous SR methods in terms of both accuracy and running time. Our code is at https://github.com/2y7c3/Super-Resolution-Neural-Operator.

1. Introduction

Single image super-resolution (SR) addresses the inverse problem of reconstructing high-resolution (HR) images from their low-resolution (LR) counterparts. In a data-driven way, deep neural networks (DNNs) learn the inversion map from many LR-HR sample pairs and have demonstrated appealing performances [4, 20, 21, 24, 29, 40, 41]. Nevertheless, most DNNs are developed in the configuration of single scaling factors, which cannot be used in scenarios requiring arbitrary SR factors [37, 38]. Recently, implicit neural functions (INF) [5, 18] have been proposed to represent images in arbitrary resolution, and paving a feasible way for continuous SR. These networks, as opposed to storing discrete signals in grid-based formats, represent signals with evaluations of continuous functions at specified coordinates, where the functions are generally parameterized by a multi-layer perceptron (MLP). To share knowledge across instances instead of fitting individual functions for each signal, encoder-based methods [5, 15, 26] are proposed to retrieve latent codes for each signal, and then a decoding MLP is shared by all the instances to generate the required output, where both the coordinates and the corresponding latent codes are taken as input. However, the point-wise behavior of MLP in the spatial dimensions results in limited performance when decoding various objects, particularly for high-frequency components [30, 32].
Neural operator is a newly proposed neural network architecture in the field of computational physics [17, 19, 23] for numerically efficient solvers of partial differential equations (PDE). Stemming from the operator theory, neural operators learn mappings between infinite-dimensional function spaces, which is inherently capable of continuous function evaluations and has shown promising potentials in various applications [9, 13, 27]. Typically, neural operator consists of three components: 1) lifting, 2) iterative kernel integral, and 3) projection. The kernel integrals operate in the spatial domain, and thus can explicitly capture the global relationship constraining the underlying solution function of the PDE. The attention mechanism in transformers [36] is a special case of kernel integral where linear transforms are first exerted to the feature maps prior to the inner product operations [17]. Tremendous successes of transformers in various tasks [6, 22, 34] have shown the importance of capturing global correlations, and this is also true for SR to improve performance. [8].

In this paper, we propose the super-resolution neural operator (SRNO), a deep operator learning framework that can resolve HR images from their LR counterparts at arbitrary scales. As shown in Fig.1, SRNO learns the mapping between the corresponding function spaces by treating the LR-HR image pairs as continuous functions approximated with different grid sizes. The key characteristics distinguishing SRNO from prior continuous SR works are: 1) the kernel integral in each layer is efficiently implemented via the Galerkin-type attention, which possesses non-local properties in the spatial dimensions and have been proved to be comparable to a Petrov-Galerkin projection [3]; and 2) the multilayer attention architecture allows for the dynamic latent basis update, which is crucial for SR problems to “hallucinate” high-frequency information from the LR image. When employing same encoders to capture features, our method outperforms previous continuous SR methods when dealing with out-of-distribution scaling factors. Inspired by INF, LIIF [5] takes continuous coordinates and latent variables as inputs, and employs an MLP to achieve outstanding performances for both in-distribution and out-of-distribution factors. In contrast, LTE [18] transforms input coordinates into the Fourier domain and uses the dominant frequencies extracted from latent variables to address the spectral bias problem [30, 32]. In a nutshell, treating images as RGB-valued functions and sharing the implicit function space are the keys to the success of LIIF-like works [5, 18]. Nevertheless, a purely local decoder, like MLP, is not able to accurately approximate arbitrary images, although it is rather sensitive to the input coordinates.

**Neural Operators.** Recently, a novel neural network architecture, Neural Operator (NO), was proposed for discretization invariant solutions of PDEs via infinite-dimensional operator learning [10, 17, 19, 23]. Neural operators only need to be trained once and are capable of transferring solutions between differently discretized meshes while keeping a fixed approximation error. A valuable merit of NO is that it does not require knowledge of the underlying PDE, which allows us to introduce it by the following abstract form,

\[(L_o u)(x) = f(x), \quad x \in D, \]
\[u(x) = 0, \quad x \in \partial D, \tag{1}\]

where \( u : D \rightarrow \mathbb{R}^{d_u} \) is the solution function residing in the Banach space \( U \), and \( L : A \rightarrow L(U; U^*) \) is an operator-valued functional that maps the coefficient function \( a \in A \) of the PDE to \( f \in U^* \), the dual space of \( U \). As in many cases the inverse operator of \( L \) even does not exist, NO seeks a feasible operator \( \mathcal{G} : A \rightarrow U, a \mapsto u \), directly mapping the coefficient to the solution within an acceptable tolerance.

The operator \( \mathcal{G} \) is numerically approximated by training a neural network \( \mathcal{G}_\theta : A \rightarrow U \), where \( \theta \) are the trainable parameters. Suppose we have \( N \) pairs of observations \( \{a_j, u_j\}_{j=1}^N \) where the input functions \( a_j \) are sampled from probability measure \( \mu \) compactly supported on \( A \), and \( u_j = \mathcal{G}(a_j) \) are used as the supervisory output functions. The infinitely dimensional operator learning problem \( \mathcal{G} \leftarrow \mathcal{G}_\theta \) thus is associated with the empirical-risk minimization problem [35]. In practice, we actually measure the approximation loss using the sampled observations \( u^{(j)}_o \)
and \( a_0^{(j)} \), which are the direct results of discretization:

\[
\min_{\theta} \mathbb{E}_{a \sim \mu} \| \mathcal{G}(a) - \mathcal{G}_0(a) \|_{\mathcal{U}} \\
\approx \min_{\theta} \frac{1}{N} \sum_{j=1}^{N} || u_0^{(j)} - \mathcal{G}_0(a_0^{(j)}) ||_{\mathcal{U}}.  \tag{2}
\]

Similar to classical feedforward neural networks (FFNs), the NO is of an iterative architecture. For ease of exposition, suppose \( \mathcal{A} \) is defined on the bounded domain \( D \subset \mathbb{R}^d \), and the inputs and outputs of the intermediate layers are all vector-valued functions, with dimension \( d_z \). Then \( \mathcal{G}_0 : \mathcal{A} \to \mathcal{U} \) can be formulated as follows:

\[
z_0(x) = \mathcal{L}(x, \alpha(x)), \tag{3}
\]

\[
z_{t+1}(x) = \sigma(W_t z_t(x) + (\mathcal{K}_t(z_t; \Phi))(x)), \tag{4}
\]

\[
u(x) = \mathcal{P}(z_T(x)), \tag{5}
\]

where \( \mathcal{L} : \mathbb{R}^{d_a+d} \to \mathbb{R}^{d_z} \), and \( \mathcal{P} : \mathbb{R}^{d_z} \to \mathbb{R}^{d_u} \) are the local lifting and projection functions respectively, mapping the input \( \alpha \) to its first layer hidden representation \( z_0 \) and the last layer hidden representation \( z_T \) back to the output function \( u \). \( W : \mathbb{R}^{d_z} \to \mathbb{R}^{d_z} \) is a point-wise linear transformation, and \( \sigma : \mathbb{R}^{d_z} \to \mathbb{R}^{d_z} \) is the non-linear activation function.

Although the PDE in (1) point-wisely defines the behavior of the solution function \( u(x) \), the solution operator \( \mathcal{G} \) we are seeking should exhibit the non-local property such that \( \mathcal{G}(a) \) can approximate \( u \) everywhere rather than locally. For this purpose, NO may employ the kernel integral operators \( \mathcal{K}_t : \{ z_t : D_t \to \mathbb{R}^{d_z} \} \to \{ z_{t+1} : D_{t+1} \to \mathbb{R}^{d_z} \} \) to maintain the continuum in the spatial domain, and one of the most adopted forms [19] is defined as

\[
(K_t(z_t; \Phi))(x) = \int_D K_t(x, y; \Phi) z_t(y) dy, \quad \forall x \in D, \tag{6}
\]

where the kernel matrix \( K_t : \mathbb{R}^{d_z+d} \to \mathbb{R}^{d_z \times d_z} \) is parameterized by \( \Phi \).

3. Super-resolution Neural Operator

All operations of NO are defined in function space and the training data are just samples at coordinates \( x_i \) irrespective of the discretization sizes. This property inspires us to employ the NO’s architecture for continuous SR. Different from NO that aims at mapping between infinite-dimensional function spaces, we define SRNO in the sense of mappings between two approximation spaces of finite-dimensional but continuous functions. This is reasonable because imaging optics inevitably limit the highest possible frequency in the radiance field under observation. With this configuration, we can design the architecture of SRNO using the well-developed theoretical tool, the Galerkin-type method that is widely used in the filed of Finite element [7].

**Problem Formulation.** Let \( (\mathcal{H}, \langle \cdot, \cdot \rangle_\mathcal{H}) \) denote a Hilbert space equipped with the inner-product structure, which is continuously embedded in the space of continuous functions \( C^0(\Omega) \), with \( \Omega \subset \mathbb{R}^2 \) a bounded domain. An image is defined as a vector-valued function \( \mathcal{H} \ni f : \Omega \to \mathbb{R}^3 \). Assume we can access the function values of \( f \) at the coordinates \( \{ x_i \}_{i=1}^{n_n} \subset \Omega \) with the biggest discretiza-

Since our data \( a^{(k)} \) and \( u^{(k)} \) are functions, to work with them numerically, we assume access only to their pointwise form. \( \mathbf{a} \) and \( \mathbf{u} \) are functions, to work with grid size \( h_f \), where \( \mathbf{a} \) and \( \mathbf{u} \) are functions, to work with grid size \( h_f \) and \( h_i \).

**Lifting.** The neural operators from PDEs usually use a simple pointwise function, \( \mathbf{L} \) in (3), to expand the input channels. But for the SR problem, a deep feature encoder \( \mathbf{E}_\psi \) (with trainable parameters \( \psi \)) and spatial interpolations should be considered in the lifting operation, due to the complexity of natural images and the discretization inconsistency between LR and HR image functions. Another critical ingredient in Lifting is the incorporation scheme of coordinates. Prior study [14] demonstrates that deep CNNs can implicitly learn to encode the information about absolute positions. As shown in Fig.2 (c), the position information of the grid points \( \mathbf{a} \) have been implicitly encoded by \( \mathbf{E}_\psi \) employing CNNs. Therefore, we only need to explicitly construct the coordinate features \( \mathbf{a}(x) \) with the fractional part of coordinate \( x \) inside a grid. In order to reduce the blocky artifacts resulting from direct interpolation of the LR feature maps, we propose to concatenate the features weighted by the corresponding bilinear interpolation factors \( s_l \). Compared to the local ensemble trick in [5], our method takes shorter running time and overcomes the over-smoothing problem. We reformulate the lifting operation in (3) as:

\[
z_0(x) = \mathbf{L}(\mathbf{c}, \{s_l \cdot \mathbf{E}_\psi(a(\hat{x}_l)) \}, \delta_l(x))_{l=1}^4,
\]

where \( x \) are coordinates of HR image functions, \( \hat{x}_l \) are the coordinates of the four neighbors of \( x \), \( s_l = (2/r_x, 2/r_y) \) represents a \( r_x \times r_y \) local area in HR images with \( r_x \) and \( r_y \) the scaling factors, and \( \mathbf{L} \colon \mathbb{R}^{4 \times (d_2 + 2)} \rightarrow \mathbb{R}^{d_z} \) is an local linear transformation function, with \( d_c \) the number of channels after \( \mathbf{E}_\psi \).

**Kernel integral.** The kernel integral operator, in (4), actually tells that we can identify the hidden representation \( z(x) \) of the underlying image function \( f(x) \) with distributions [28], inner products in the settings of SRNO. We first define the kernel \( K_{ij} \colon \mathbb{R}^{d_z \times d_z} \rightarrow \mathbb{R}^{d_z \times d_z} \) with respect to the input pair \((z(x), z(y))\), rather than like (6) depending on the spatial variables \((x, y)\). Furthermore, in order to more efficiently explore \( z(x) \) we can use multiple sets of test functions defining the distributions, which may immediately remind one of a single-head self-attention [17]. Denote \( z_i = z(x_i) \in \mathbb{R}^{d_z} \) for \( i = 1, \ldots, n_{h_f} \). The kernel integral operator can be approximated by the Monte-Carlo method (omitting the layer index):

\[
(K(z))(x) = \int_{\Omega} K(z(x), z(y))z(y)dy \\
\approx \sum_{i=1}^{n_{h_f}} K(z(x), z_i)z_i, \quad \forall x \in \Omega_{h_f},
\]

where

\[
K(z(x), z_i) = \exp \left( \frac{(W_{xz}z(x), W_{z_i z_i})}{\sqrt{d_z}} \right) \sum_{j=1}^{n_{h_f}} \exp \left( \frac{(W_{z_i z_i}, W_{z_i z_i})}{\sqrt{d_z}} \right) W_{z_i}.
\]

In the language of transformers, the matrices \( W_q, W_k, W_v \in \mathbb{R}^{d_z \times d_z} \) correspond to the queries, keys and values functions respectively. By allowing \( K \) to be the sum of multiple functions with separate trainable parameters, the multi-head self-attention can also be rewritten like in (9). For every observation \((a, u)\), this operation has a complexity of \( O(n_{h_f}^2 d_z) \), which is unaffordable for the SR problem where the sampling number \( n_{h_f} \) usually reaches to 10,000 or beyond. To overcome this issue, we employ the Galerkin-type attention operator which has a linear complexity \( O(n_{h_f} d_z^2) \). The approximation capacity of a linearized Galerkin-type attention has been proved to be comparable to a Petrov-Galerkin projection [3].

We suppose that a image function \( f \) is locally integrable, i.e., \( f \) is measurable and \( \int_{\Omega} |f| < \infty \) for every compact \( \Omega \subset \mathbb{R}^2 \). The idea behind distributions [28] is to identify \( f \) with \( f \phi \) by suitably choosing a “test function” \( \phi \), rather than by a series of separate evaluations \( f(x) \) at coordinates \( x \in \Omega \). Let \( r(x) = W_q z(x) \), \( k(x) = W_k z(x) \), and \( q(x) = W_v z(x) \) be the keys, queries and values functions respectively, which are \( d_z \)-dimensional vector-valued functions, with a subscript denoting the corresponding component. The kernel integral attention then can be written in a component-wise form \((j = 1, \ldots, d_z)\):

\[
((K(z))(x))_j = \sum_{l=1}^{d_z} \langle k_l, v_j \rangle q_l(x) \\
\approx \sum_{l=1}^{d_z} \left( \int_{\Omega} k_l(y) v_j(y)dy \right) q_l(x), \quad \forall x \in \Omega_{h_f}.
\]
Denote the evaluations \((z(x_1), \ldots, z(x_{n_h}))^T\) by \(z \in \mathbb{R}^{n_h \times d_z}\). Let \(Q = zW_q, K = zW_k, V = zW_v \in \mathbb{R}^{n_h \times d_z}\). The columns of \(Q/K/V\) contain the vector representations of the learned basis functions, spanning certain subspaces of the latent representation Hilbert spaces respectively.

\[
z = Q(\tilde{K}^T \tilde{V})/n_{h_z}, \quad (12)
\]

where \(\tilde{K} = \text{Ln}(K), \tilde{V} = \text{Ln}(V)\), with \(\text{Ln}(\cdot)\) the layer normalization. The \(j\)-th component of \(\tilde{K}^T \tilde{V}\) contains the coefficients for the linear combination of the basis vectors \(\{q_j\}_{j=1}^{d_z}\) to form the output \(z\). Consequently, the global correlations are reflected through the components of \(\tilde{K}^T \tilde{V}\).

As a result, SRNO can utilize all information effectively where correlations are reflected through the components of \(\tilde{K}^T \tilde{V}\).

The columns of \(Q, K, V\) are normalized. The \(\text{Ln}\) normalization. Consequently, the global correlations are reflected through the components of \(\tilde{K}^T \tilde{V}\).

**4. Experiments**

**4.1. Training**

**Dataset.** We used the DIV2K dataset \([1]\) for network training, while the DIV2K validation set \([1]\) and four benchmark datasets, including Set5 \([2]\), Set14 \([39]\), B100 \([25]\) and Urban100 \([12]\), for evaluation. Peak signal-to-noise ratio (PSNR) is used as the evaluation metric. Similar to \([5,11]\), we cropped image boundaries when computing PSNRs.

**Implementation Details.** Numerically, we have to explore the underlying continuous image function through access to its point-wise evaluations on the HR and LR grids. Let \(B\) be the batch size and \(n_{h_z}\) be the LR sampling counts. We first sample \(B\) random scales \(r^{(i)}\) following a uniform distribution \(U(1, 4)\), and then crop \(B\) patches of sizes \(\{\sqrt{n_{h_z}r^{(i)}} \times \sqrt{n_{h_z}r^{(i)}}\}_{i=1}^{B}\) from the HR training images (one per each). We find \(n_{h_z} = 128^2\) is sufficient for SRNO to represent most natural image patches and fix this choice in the sequel. The LR counterparts are downsampled using bicubic interpolation with the corresponding \(r^{(i)}\).

In order to keep a consistent number of supervision points for the LR patches of different scales in a single batch, we randomly sample \(128^2\) HR pixels and calculate the corresponding fractional coordinates on the coarse grid associated with \(r^{(i)}\), as is done in \([5]\). We use an L1 loss \([21]\) and the Adam \([16]\) optimizer with an initial learning rate \(4 \times 10^{-5}\) and the maximum \(4 \times 10^{-4}\). All models are trained for 1000 epochs with batch size 64, and the learning rate decays by the cosine annealing after a warm-up phase of 50 epochs.

**4.2. Evaluation**

**Quantitative result.** On the DIV2K validation set, Table 1 presents a quantitative comparison among our SRNO and three existing arbitrary-scale SR methods, MetaSR \([5,11]\), LIIF \([5]\), and LTE \([18]\). Results when EDSR-baseline \([21]\), and RDN \([41]\) are used as encoders are displayed in the top and bottom rows, respectively. One can observe that SRNO provides the best results over all scale factors, irrespective of the encoder employed. Additionally, on in-distribution scales \((\times 2, \times 3, \times 4)\), our method outperforms earlier works by wide margins.

As RDN facilitates better reconstruction accuracy than EDSR-baseline in Tab.1, we choose RDN as the encoder in the comparisons over the four benchmark datasets. The results are listed in Tab.2, where SRNO obtains all the best performances. Note that only SRNO produces substantial improvements over RDN on every in-distribution scale, although the latter trains separate models for each scale. As the experimental settings in Tab.1 and 2 only differ in the decoder part, we conclude that the Galerkin-type attention mechanism employed in SRNO does contribute to better function approximation capability.

**Qualitative result.** Figure 3 illustrates the visual results...
obtained from two images in the Urban100 and DIV2K validation datasets. Both the regions marked with a rectangle are rich in high-frequency structures. The bicubic interpolations exhibit severe aliasing artifacts, which obviously have been transferred into the latent codes of LIIF [5] as it directly relies on the interpolation of the feature maps and the bilinear ensemble. Thanks to the local Fourier frequency estimation step, LTE [18] performs somewhat better in partial regions (e.g. the top-right corner of the building in the second row), but aliasing still dominates on the whole. In contrast, SRNO successfully recovers the fine structures, which again verifies the importance of the global integral kernels in capturing the correct overall structures.

Figure 4 shows two examples of scene text images on non-integer scales. In the first row, only SRNO can clearly recover the word "ANTIQUES". And in the second row, only SRNO can consistently restore the two e’s. Obviously, the Galerkin-type attention attributes to this global consistency.

4.3. Ablation Study

Data sampling. The quantity of sampling points is crucial in our super-resolution operator learning architecture. Figure 5 shows that almost all scales perform better when using additional pixel samples. However, the benefit becomes less as the numbers of samples increases. 96×96 or 128×128 points would be good trade-offs between training time and performance. Figure 6 shows that more sampling points can more accurately represent an RGB-based function, which facilitates to learn the map between two approximation spaces. Using the DIV2K (800 images) and Flickr2K (2650 images) datasets, we also train an EDSR-baseline-MLP with position embedding, denoted by MLP\(^p\).

We can see that our SRNO (64×64/96×96/128×128) obtains better performance for scales ×3, ×4, ×6, ×8, trained with only 800 images in DIV2K. The curve for MLP(128×128) in Fig.5 reflects the performance advantage of SRNO(128×128), both trained with DIV2K dataset. Random and sequential sampling results are presented in Supplementary.
<table>
<thead>
<tr>
<th>Method</th>
<th>In-distribution</th>
<th>Out-of-distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\times 2$</td>
<td>$\times 3$</td>
</tr>
<tr>
<td>Bicubic</td>
<td>31.01</td>
<td>28.22</td>
</tr>
<tr>
<td>EDSR-baseline [21]</td>
<td>34.55</td>
<td>30.90</td>
</tr>
<tr>
<td>EDSR-baseline-LIIF [5]</td>
<td>34.67</td>
<td>30.96</td>
</tr>
<tr>
<td>EDSR-baseline-LTE [18]</td>
<td>34.72</td>
<td>31.02</td>
</tr>
<tr>
<td>EDSR-baseline-SRNO (ours)</td>
<td>34.85</td>
<td>31.11</td>
</tr>
</tbody>
</table>

Table 1. Quantitative comparison on DIV2K validation set (PSNR (dB)). The best performance are bolded. EDSR-baseline trains separate models for the three in-distribution scales. The rest methods use a single model for all scales, and are trained with continuous random scales uniformly sampled in $\times 1-\times 4$.

<table>
<thead>
<tr>
<th>Method</th>
<th>In-distribution</th>
<th>Out-of-distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\times 2$</td>
<td>$\times 3$</td>
</tr>
<tr>
<td>RDN [41]</td>
<td>38.24</td>
<td>34.71</td>
</tr>
<tr>
<td>RDN-MetaSR [5, 11]</td>
<td>38.22</td>
<td>34.63</td>
</tr>
<tr>
<td>RDN-LIIF [5]</td>
<td>38.17</td>
<td>34.68</td>
</tr>
<tr>
<td>RDN-LTE [18]</td>
<td>38.23</td>
<td>34.72</td>
</tr>
<tr>
<td>RDN-SRNO (ours)</td>
<td>38.32</td>
<td>34.84</td>
</tr>
</tbody>
</table>

Table 2. Quantitative comparison on benchmark datasets (PSNR (dB)). The best performances are in bold. RDN trains separate models for the three in-distribution scales. The rest methods use a single model for all scales, and are trained with continuous random scales uniformly sampled in $\times 1-\times 4$.

<table>
<thead>
<tr>
<th>Method</th>
<th>#Params (M)</th>
<th>#FLOPs (G)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDSR-baseline-LIIF</td>
<td>1.6</td>
<td>85.0</td>
</tr>
<tr>
<td>EDSR-baseline-LTE</td>
<td>1.7</td>
<td>75.3</td>
</tr>
<tr>
<td>EDSR-baseline-SRNO (ours)</td>
<td>2.0</td>
<td>65.8</td>
</tr>
<tr>
<td>RDN-LIIF</td>
<td>22.4</td>
<td>765.2</td>
</tr>
<tr>
<td>RDN-LTE</td>
<td>22.5</td>
<td>755.5</td>
</tr>
<tr>
<td>RDN-SRNO (ours)</td>
<td>22.8</td>
<td>746.0</td>
</tr>
</tbody>
</table>

Table 3. Comparison of model parameters and FLOPs. Metrics are measured under the setting of $128^2$ sample points.

**Other design choices.** In Tab.4, we retrain the following models using EDSR-baseline [21]. All the tests were conducted using a single NVIDIA RTX 3090. In comparing our interpolation-free method to SRNO(-i) using nearest-neighbor interpolation, we find that it works better and faster. It should be noted that we only generate the image once, as opposed to employing an MLP four times as in LIIF [5] and LTE [18]. Figure 7 shows that SRNO, compared to SRNO(-i), alleviates the blocky artifact, and compared to the local ensemble [18], overcomes the over-smoothing problem. By comparing SRNO with SRNO(-h), it can be shown that cutting the number of heads from 16 to 8 causes an increase in time costs because GPU’s parallelism is reduced. We see that reducing the number of basis functions and the iterative updating layers results in a considerable performance drop by comparing SRNO with SRNO(-w) and SRNO(-l). The number of basis functions ($d_z$) and the number of subspaces ($n_{heads}$) play an important role in the approximation power of our model.

**4.4. Dynamic Basis**

We investigate the dynamic basis updating phenomenon across the approximation subspaces in this section. We...
Table 4. Quantitative ablation study on design choices of SRNO. Evaluated on the Set14 (PSNR(dB)). -h/w/l refers to $n_{\text{heads}} = 8$, $d_z = 128$, and $T = 1$ correspondingly. -i refers to using nearest-neighborhood interpolation and position embedding. EDSR-baseline is used in lifting. The running time is averaged over Set14 on $\times 2/3/4/6/8$ SR. For each scale SR, we run every model three times and choose the minimum running time.

![Image showing ablation study results](image_url)

Figure 6. Effects of sampling global image structures. Test on $\times 8$ scale. EDSR-baseline is used as the encoder, while the data in parentheses indicate the pixel numbers sampled in the HR images during training.

![Image showing ablation study results](image_url)

Figure 7. Interpolation vs. interpolation-free of SRNO. Test on $\times 8$ scale. EDSR-baseline is used as an encoder. -i refers to using nearest-neighborhood interpolation and position embedding.

![Image showing ablation study results](image_url)

Figure 8. Dynamic basis update. The annotation $t = k$ refers to the iterative layer number. The pictures display one of the learned basis functions in $V$ and the latent representations $z$.

![Image showing ablation study results](image_url)

Figure 9. Ranks of the matrix $z$. Evaluated on the Set5 dataset.

SRNO would not be able to take any chances to enrich the basis for every individual image through optimizations. The increasing ranks of the latent representation matrix $z$, shown in Fig. 9, provide the evidence that Galerkin-type attention is becoming more complex than a straightforward linear combination of the existing bases. Due to the space limitation, we have provided a principled discussion on the dynamic basis update processes in the supplementary material.

5. Conclusion

In this paper, we proposed the Super-Resolution Neural Operator (SRNO) for continuous super-resolution. In SRNO, each image is seen as a function, and our method learns a map between finite-dimensional function spaces, which allows SRNO can be trained and generalize on different levels of discretization. First, in the Lifting, we use CNN-based encoders to capture feature maps from LR images and design a simple but efficient interpolation-free method, addressing the discretization inconsistency problem encountered in SR. Second, to approximate our target function in the Iterative Kernel Integration, we employ a linear attention operator that has been proved to be comparable to the Petrov-Galerkin projection. Finally, we map the last hidden representation to the output function. Experimental results show that our SRNO outperforms other arbitrary-scale SR methods in performance as well as computation time, and particularly in the capability of capturing the global image structures.
References


