Image Super-Resolution with Non-Local Sparse Attention

Yiqun Mei, Yuchen Fan, Yuqian Zhou
University of Illinois at Urbana-Champaign

Abstract

Both Non-Local (NL) operation and sparse representation are crucial for Single Image Super-Resolution (SISR). In this paper, we investigate their combinations and propose a novel Non-Local Sparse Attention (NLSA) with dynamic sparse attention pattern. NLSA is designed to retain long-range modeling capability from NL operation while enjoying robustness and high-efficiency of sparse representation. Specifically, NLSA rectifies non-local attention with spherical locality sensitive hashing (LSH) that partitions the input space into hash buckets of related features. For every query signal, NLSA assigns a bucket to it and only computes attention within the bucket. The resulting sparse attention prevents the model from attending to locations that are noisy and less-informative, while reducing the computational cost from quadratic to asymptotic linear with respect to the spatial size. Extensive experiments validate the effectiveness and efficiency of NLSA. With a few non-local sparse attention modules, our architecture, called non-local sparse network (NLSN), reaches state-of-the-art performance for SISR quantitatively and qualitatively.

1. introduction

Single Image Super-Resolution (SISR) has attracted great attention in recent years. In general, the goal of SISR is to reconstruct a high-resolution image given its low-resolution counterpart. Due to the ill-posed nature of SISR task, a variety of image priors [12, 14, 24, 36, 40, 46] were proposed as the regularizers, including the most representative sparse and non-local priors, which are the focus of this paper.

For decades, sparsity constraints have been well-explored as a powerful driving forces for many image reconstruction problems [4, 7, 19], especially SISR [46]. With sparse coding, images are well-expressed as the sparse linear combinations of atoms in a predefined over-complete dictionary such as wavelet [11] and curvelet [9] functions. Combining with exemplar-based approaches, sparse representation developed the dictionary using raw image patches [46] or learned semantic feature patches from the degraded image itself [17, 19] or external datasets[47]. As the deep Convolution Neural Networks (CNNs) for SISR emerges, the non-linearity activation between layers embraces the benefits of sparsity prior. Dong et al. propose the SRCNN [16] to first successfully bridge convolution to classic sparse coding where the ReLU activation roughly enforced 50% sparsity by zeroing-out all negative entries. Recently, Fan et al. [21] go beyond that by explicitly imposing sparsity constraints upon hidden neurons and conclude that sparsity in feature representation is indeed beneficial and favorable. It is widely proven that the sparsity constraints lead to high efficiency by largely decreasing the number of elements to represent images. It also yields a more powerful and robust expression in handling inverse problems theoretically [10, 20] and practically.

Another widely-explored image prior is the Non-Local (NL) prior. For SISR, adopting Non-Local Attention becomes a more prevalent way [37, 51] to utilize the image self-similarity prior that small patterns tend to recur within the same image [5]. NL operation searches for those similar patterns globally, and selectively sums over those correlated features to enhance the representation. Though Non-Local Attention is intuitive and promising to fuse features, directly applying it for SISR task will encounter some issues that cannot be ignored. First, the receptive field of features in deeper layers tend to be global, thus the mutual-correlation computation among deep features are not that accurate [33]. Second, global NL attention requires the computation of feature mutual-similarity among all the pixel locations. It results in quadratic computational cost with respect to image size. To alleviate the above mentioned problems, one strategy is to limit the NL searching range within a local neighbourhood. But it reduces computational cost at the expense of missing much global information.

In this paper, for the specific SISR task, we aim to enforce sparsity in the Non-Local attention module, as well as largely reduce its computational cost. Specifically, we propose a novel Non-Local Sparse Attention (NLSA) and embed it into a residual network baseline like EDSR [32] to form a Non-Local Sparse Network (NLSN). To force the sparsity of the NLSA, we spatially partition the deep feature pixels into different groups (termed attention buckets.
in this paper). The feature pixels inside the same bucket are considered content closely-correlated. We then apply the Non-Local (NL) operation within the bucket that the query pixel belongs to, or across adjacent buckets after sorting. We achieve this by building the partition approach upon Locality Sensitive Hashing (LSH) research [23] that searches for similar elements which produce maximum inner product.

The proposed NLSA will make it possible to reduce the computational complexity of NL from quadratic to asymptotic linear with respect to spatial dimensions. Searching similar cues within a smaller content-correlated bucket will also make the module attend to locations which are more informative and related. As a result, NLSA retains global modeling ability of the standard NL operation, while enjoying robustness and efficiency from its sparse representation. In summary, the main contributions of our paper are:

- We propose to enforce sparsity in Non-Local operation for SISR task via a novel Non-Local Sparse Attention (NLSA) module. The sparsity constraint forces the module to focus on correlated and informative area while ignoring unrelated and noisy contents.

- We achieve the feature sparsity by first grouping the feature pixels and only conducting Non-Local operations within the group named attention bucket. We adopt the Locality Sensitive Hashing (LSH) for grouping and assign each group a Hash code. The proposed approach significantly reduces the computational complexity from quadratic to asymptotic linear.

- Without any bells and whistles, a few NLSA modules can drive a fairly simple ResNet backbone to state-of-the-arts. Extensive experiments demonstrate the advantages of NLSA over the standard Non-Local Attention (NLA).

2. Related Work

2.1. Sparse representation.

In this section, we briefly review the key concepts of the sparse representation. Formally, Suppose \( x_1, x_2, ..., x_n \in \mathbb{R}^d \) are \( n \) known examples in an over-complete dictionary \( \mathcal{D}^{d \times n} \) \( (d < n) \). For a query signal \( y \in \mathbb{R}^d \), exemplar-based approaches [12, 46] represent it as a weighted sum of the elements in \( \mathcal{D} \):

\[
y = \alpha_1 x_1 + \alpha_2 x_2 + \ldots + \alpha_n x_n \tag{1}
\]

where \( \alpha_i \) is a coefficient with respect to the \( x_i \). Let \( \alpha = [\alpha_1, \alpha_2, ..., \alpha_n] \), then Eq. 1 can be written as:

\[
y = D\alpha \tag{2}
\]

Eq. 2 yields an undetermined linear system. Solving \( \alpha \) becomes an ill-posed problem. To alleviate this, sparse representation assumes that \( y \) should be sparsely represented, i.e., \( \alpha \) should be sparse:

\[
y = D\alpha, \quad s.t. \|\alpha\|_0 \leq k \tag{3}
\]

Where \( \|\cdot\|_0 \) and \( k \) counts and bounds the number of non-zero elements in \( \alpha \), respectively. Given the sparsity constraint, optimization methods like OMP [38] can effectively approximate the solution of Eq 3. The resulting sparse representation has been proven to be extremely powerful in the field of image reconstruction [19, 47, 53]. Motivated by their success, we are inspired to incorporate sparse representation into non-local attention.

2.2. Non-Local Attention (NAL) for image SR.

Non-local operation assumes that small patches tend to re-occur within the same image, which has been well-demonstrated to be a strong prior for natural images [5]. Non-local approaches were designed to utilize these self-recurrences to recover underlying signals. Non-local operation has been widely applied in many image restoration problems, such as super-resolution [22], denoising [1, 2, 8, 13], and inpainting [18]. Wang et al. [45] first bridges classic non-local filtering to self-attention methods [43] for machine translation and further introduces Non-Local Attention (NLA) into deep neural networks to capture global semantic relationships for high-level tasks. For image super-resolution, recent approaches, such as NLRN [33], SAN [15], RNAN [51], and CSNBN [37], demonstrate considerable benefits of exploring long-range feature correlations by adopting NL attention. However, the existing NLA designs for SISR task are either limited to a local neighbourhood, or largely computational resource-consuming. Motivated by recent progress [29, 39, 44] on self-attention methods for language modeling, we propose Non-Local Sparse Attention (NLSA) to embrace the long-range information as well as reducing the complexity.

3. Non-Local Sparse Attention (NLSA)

3.1. General Form of Sparse Attention

As discussed above, the merits of Non-Local Attention for image SR often come at a price of limiting its searching range. To alleviate the issue, we propose to bridge standard NLA to exemplar-based approaches and then break the tie by imposing sparsity constraint.

Non-Local Attention. In general, a Non-Local Attention enhances an input feature map \( X \in \mathbb{R}^{h \times w \times c} \) by summarizing information from all positions. For illustration purpose, we reshape \( X \) into an 1-D feature \( X \in \mathbb{R}^{n \times c} \) where \( n = hw \). Given a query location \( i \), the corresponding
shows some examples of the attention bucket. Figure 1. Examples of the Attention Bucket in 2D spatial space. Given a query at location \(i\) and an index set \(\delta_i\), where the \(\delta_i\) decides on the group of locations to compute the non-locally fused features. Darker blue regions in the figure form the attention bucket. \(\delta_i = \{j\}\) indicates using the full-range pixels like in the standard NL attention. \(\delta_i = \{j | |j - i| < L\}\) indicates a local neighbourhood constrained attention span. While \(\delta_i = \{j | h(x_i) = h(x_j)\}\) is the proposed hash-based attention bucket.

output response \(y_i \in \mathbb{R}^c\) can be expressed as:

\[
y_i = \sum_{j=1}^{n} \frac{f(x_i, x_j)}{\sum_{j=1}^{n} f(x_i, x_j)} g(x_j)
\]

where \(x_i, x_j\) and \(x_j\) are pixel-wise features at location \(i, j\) and \(j\) on \(X\). \(f(., .)\) measures the mutual similarity and \(g(.)\) is a feature transformation function. Eq. 4 can be viewed as an exemplar-based approach (Eq. 2) by setting \(D = [g(x_1), ..., g(x_n)] \in \mathbb{R}^{c \times n}\) and \(\alpha_i = [f(x_i, x_1), ..., f(x_i, x_n)] \in \mathbb{R}^n\), i.e., \(y_i = D\alpha_i\).

**Sparsity Constraints on Non-Local Attention.** Given Eq. 4, a sparsity constraint can be imposed on the Non-Local Attention by limiting the number of non-zero entries of \(\alpha\) up to a constant \(k\). Therefore, a general form of Non-Local Attention with Sparse Constraints can be derived as:

\[
y_i = D\alpha_i \quad \text{s.t.} \quad ||\alpha_i||_0 \leq k
\]

where \(\delta_i\) indexes non-zero elements of \(\alpha_i\), i.e., \(\delta_i = \{j | \alpha_i[j] \neq 0\}\), where \(\alpha_i[j]\) denotes the \(j\)-th element in \(\alpha_i\). With sparse attention, the computational cost can be largely saved by ignoring elements with zero coefficients.

**Attention Bucket.** Notice that the index set \(\delta_i\) indicates the group of pixel locations where a given query should attend to. In another word, \(\delta_i\) constrains the identified locations where the Non-Local Attention can be computed from. In this paper, we define this group of locations as in an Attention Bucket. Figure 1 shows some examples of the attention bucket under different \(\delta_i\). For example, standard non-local attention spans over all the possible locations, which makes the aggregated feature noisy and less informative. If an attention spans a local neighborhood of length \(L\), this specifies a window \(\delta_i = \{j | |j - i| < L\}\). In this case, some long-range context cannot be effectively aggregated.

Intuitively, a more powerful sparse attention is expected to cover locations that are the most informative and closely related at a global scale, resulting that ignoring other elements brings no harm to the performance. A naive way is to rank all mutual similarities and then use the top \(k\) entries. However, it requires forming a full attention first which brings no efficiency improvement. In the following sections, we will show how we form the attention bucket for each query \(i\) by globally modelling the attention with high efficiency.

### 3.2. Attention Bucket from Locality Sensitive Hashing (LSH)

As discussed above, a desired attention should not only keep sparse but also incorporate the most relevant elements. In this section, we propose to adopt the Spherical Locality Sensitive Hashing (LSH) \([3, 42]\) to form the desired attention bucket containing global and correlated elements with the query element. Specifically, we propose to spatially partition the embedding space into buckets of similar features depending on their angular distances. Consequently, even when the attention keeps sparse by only spanning over one bucket, it could still capture most of the correlated elements.

Recall that a hashing scheme is locality sensitive if nearby elements are at high possibility to fall into the same hash bucket (hash code) whereas distant ones are not. The spherical LSH is an instance of LSH designed for angular distance. One can intuitively think it as randomly rotating a cross-polytope inscribed into a hyper-sphere, as shown in the top branch of Figure 2. The hash function projects a tensor onto the hyper-sphere and the closest polytope vertex is selected as its hash code. Thus, if two vectors have a small angular distance, they are likely to fall in the same hash bucket, which is also the defined attention bucket.

Formally, suppose we want to get \(m\) hash buckets, we have to first project the target tensor onto a hyper-sphere and randomly rotate it with a matrix \(A \in \mathbb{R}^{c \times m}\), a sampled random rotation matrix with i.i.d. Gaussian entries, i.e.,

\[
\hat{x} = A \frac{x}{\|x\|_2}
\]

The hash code, or the assigned hash bucket, is defined as \(h(x) = \arg \max_i (\hat{x})\). After hashing all elements, we manage to partition the space into buckets of correlated elements, and the attention bucket of \(x_i\) can be identified by the index set \(\delta_i = \{j | h(x_j) = h(x_i)\}\).

In practice, the spherical LSH is simultaneously performed for all elements with batch matrix multiplication, which only adds negligible computational cost. Knowing which bucket to attend in advance, the model can achieve high-efficiency and robustness by ignoring other noisy or less-correlated partitions.

### 3.3. Non-Local Sparse Attention

Once the attention bucket index set \(\delta_i\) for query location \(i\) is determined, the proposed Non-Local Sparse Attention
Figure 2. The proposed Non-Local Sparse Attention (NLSA). The upper branch partitions the input features into buckets via Spherical Locality Sensitive Hashing (LSH). The bottom branch computes attention for every query within each bucket or between adjacent buckets after sorting the buckets by the hash code. The module embraces the advantages of Non-Local Attention (modeling globally), and the benefits of sparsity and hashing (high efficiency).

(NLSA) can be easily derived from Eq 6. Specifically, as shown in Figure 2, NLSA assigns each pixel-wise feature in X to a bucket sharing the same hash code based on their content relevance, and only the corresponding bucket elements contribute to the output. In the following, we describe some techniques used in our practical implementation.

**Dealing with Unbalanced Bucketing.** Ideally, given totally m buckets, each hash bucket will equally contain \( \frac{n}{m} \) elements. However, this may not hold in practice as buckets tend to be unbalanced. This also makes parallel computing very difficult. To overcome the difficulty, we first sort features by their bucket value (hash code), then the permutation is defined as \( \pi : i \rightarrow \pi(i) \). After knowing their new positions (denoted by the superscripts), we split them into chunks of size \( k \):

\[
C_j = [x^{jk+1}, x^{jk+2}, \ldots, x^{(j+1)k}] \tag{8}
\]

where \( C_j \) presents the \( j \)-th chunk. Consequently, the attention bucket of \( x_i \) is updated to the corresponding chunk,

\[
\delta_i = \text{Index}(C_j) \quad \text{if } \pi(i) \in [jk+1, (j+1)k] \tag{9}
\]

The above strategy is more friendly used to perform computation in parallel. Despite its merits, splitting the original buckets into fixed-size chunks as the updated attention buckets also brings a subtle issue: some new chunks may cross the original bucket boundaries, as shown in Figure 2.

Fortunately, this issue can be effectively alleviated by allowing attention to also span over adjacent chunks.

**Multi-round NLSA.** The nature of Spherical LSH indicates there is always a small chance that some correlated elements are incorrectly hashed into different hash buckets. Fortunately, this chance can be reduced by independently hashing multiple rounds and taking the union of all results. Motivated by this observation, we propose multi-rounds NLSA to make hashing process more robust. Let \( \delta_{r,i} \) denote the resulting attention bucket of \( x_i \) of the \( r \)-th hashing, and \( \text{Att}(x_i, \delta_{r,i}) \) be the associated sparse attention defined in Eq. 6, i.e.,

\[
\text{Att}(x_i, \delta_{r,i}) = \sum_{j \in \delta_{r,i}} \frac{f(x_i, x_j)}{\sum_{j' \in \delta_{r,i}} f(x_i, x_{j'})} g(x_j) \tag{10}
\]

Then the multi-round NLSA is defined as:

\[
x_i = \sum_r \sum_{j \in \delta_{r,i}} \frac{f(x_i, x_j)}{\sum_j f(x_i, x_j)} \text{Att}(x_i, \delta_{r,i}) \tag{11}
\]

Intuitively, multi-round NLSA is the weighted sum of each single round attention results, and the weight coefficient represents the normalized similarity between the query and the elements in its assigned hash bucket for each round. As a side effect, this augmentation linearly increases computational cost with respect to the total hash rounds. But we can still dynamically adjust this parameter during the evaluation time to study the trade-offs.
**Computational complexity.** We analyze the time complexity of the proposed NLSA. Given an input feature $X \in R^{n \times c}$, the cost for spherical LSH with $m$ buckets a matrix multiplication, which is $O(ncm)$. The cost for attention operation (Eq. 6) with sparsity constraint (size of the attention bucket) $k$ is $O(nck)$. The sorting operation of a sequence with length $n$ and $m$ distinct numbers (bucket number) adds an additional $O(nm)$ with quick sort (and can be further optimized with advanced sorting algorithms). Therefore, the overall computational cost of our non-local sparse attention is $O(nck + ncm + nm)$. Hashing for $r$ rounds will increase computational cost with a factor $r$, resulting in $O(rnck + rncm + rnm)$. NLSA only takes linear computational complexity with respect to the input spatial size.

**Instantiations.** To instantiate the non-local attention defined in Eq. 6, we choose embedded Gaussian for $f(\ldots)$, i.e. $f(x, y) = \exp(\theta(x)^T \phi(y))$, where $\theta$ and $\phi$ are learned linear projections. In this paper, we use one of its variants that sets $\theta = \phi$ to ensure the projected features are in the same subspace for better LSH. We also found sharing $\theta$ and $\phi$ did not hurt the performance, which will be verified in experiment section.

### 3.4. Non-Local Sparse Network (NLSN)

To demonstrate the effectiveness of the non-local sparse attention, we build our non-local sparse network (NLSN) upon a fairly simple EDSR [32] backbone, which consists of 32 residual blocks. As shown in Figure 3, the network uses total 5 attention blocks with one insertion after every 8 residual blocks. The network is trained solely with $L_1$ reconstruction loss.

### 4. Experiments

#### 4.1. Datasets and Metrics

Following [32, 52], we use DIV2K as our training dataset, which contains 800 training images. We test our approach on 5 standard benchmarks: Set5 [6], Set14 [48], B100 [34], Urban100 [27] and Manga109 [35]. We evaluate all the results using PSNR and SSIM metrics on Y channel in the transformed YCbCr space.

#### 4.2. Implementation and Training Details

For the non-local sparse attention, we set the attention bucket size (i.e. chunk size) $k = 144$. The corresponding number of hash buckets $m = \min(\frac{hw}{k}, 128)$ is dynamically determined by the division of input size $h \times w$ and $k$ but clipped at 128. The final non-local sparse network is built upon an EDSR backbone with 32-residual blocks and 5 additional NLSA blocks. We set all the convolutional kernel sizes to $3 \times 3$. All intermediate features have 256 channels the same as in EDSR, except for those embedded ones in the attention blocks, which have 64 channels. The last convolution layer transforms the deep features into a $n \times 3$-channel RGB image with 3 filters. By default, the model is trained and evaluated with NLSA of $r = 4$ rounds.

During training, we randomly crop $48 \times 48$ patches from the training examples and form a mini-batch of 16 images. The training images are further augmented via horizontal flipping and random rotation of 90, 180, and 270 degrees. We optimize the model by ADAM optimizer [28] with $\beta_1 = 0.9$, $\beta_2 = 0.99$ and $\epsilon = 10^{-8}$. The learning rate is set to $10^{-4}$ and reduced by 0.5 after 200 epochs. The final model is obtained after 1000 epochs. Our model is implemented using PyTorch and trained on Nvidia 1080ti GPUs.

#### 4.3. Comparisons with State-of-the-Arts

To demonstrate the effectiveness of our NLSA, we compare it with 12 state-of-the-arts including LapSRN [30], SR-MDNF [49], MemNet [41], EDSR [32], DBPN [25], RDN [52], RCAN [50], NLRN[33], RNAN [51], SRFBN [31], OISR [26] and SAN [15].

The quantitative results are shown in Table 1. Our NLSN achieves the best results on almost all benchmarks and all upsampling scales. In particular, when comparing with its backbone EDSR, adding additional NLSAs shows great superiority in improving performance and even makes EDSR outperform the very competitive RCAN and SAN. Especially, the proposed NLSN brings improvements around 0.2 dB in Set5 and Set14, 0.1 dB in B100 and more than 0.4 dB in Urban100 and Manga109. These performance gains show that NLSA succeeds in exploring extensive global cues for more accurate super-resolution. Moreover, when comparing with previous non-local approaches like NLRN and RNAN, our network shows a huge advance in all en-
Table 1. Quantitative results on benchmark datasets. Best and second best results are highlighted and underlined.

<table>
<thead>
<tr>
<th>Method</th>
<th>Scale</th>
<th>Set5</th>
<th>SSIM</th>
<th>Set14</th>
<th>SSIM</th>
<th>B100</th>
<th>Urban100</th>
<th>SSIM</th>
<th>Manga109</th>
<th>SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>LapSRN [30]</td>
<td>×2</td>
<td>37.52</td>
<td>0.9591</td>
<td>33.08</td>
<td>0.9130</td>
<td>31.08</td>
<td>0.8950</td>
<td>30.41</td>
<td>0.9101</td>
<td>37.27</td>
</tr>
<tr>
<td>MemNet [41]</td>
<td>×2</td>
<td>37.78</td>
<td>0.9597</td>
<td>33.28</td>
<td>0.9142</td>
<td>32.08</td>
<td>0.8978</td>
<td>31.31</td>
<td>0.9195</td>
<td>37.72</td>
</tr>
<tr>
<td>SRMDNF [49]</td>
<td>×2</td>
<td>37.79</td>
<td>0.9601</td>
<td>33.32</td>
<td>0.9159</td>
<td>32.05</td>
<td>0.8985</td>
<td>31.33</td>
<td>0.9204</td>
<td>38.07</td>
</tr>
<tr>
<td>DBPN [25]</td>
<td>×2</td>
<td>38.09</td>
<td>0.9600</td>
<td>33.85</td>
<td>0.9190</td>
<td>32.27</td>
<td>0.9000</td>
<td>32.55</td>
<td>0.9324</td>
<td>38.89</td>
</tr>
<tr>
<td>RDN [52]</td>
<td>×2</td>
<td>38.24</td>
<td>0.9614</td>
<td>34.01</td>
<td>0.9212</td>
<td>32.34</td>
<td>0.9017</td>
<td>32.89</td>
<td>0.9353</td>
<td>39.18</td>
</tr>
<tr>
<td>RCAN [50]</td>
<td>×2</td>
<td>38.27</td>
<td>0.9614</td>
<td>34.12</td>
<td>0.9216</td>
<td>32.41</td>
<td>0.9027</td>
<td>33.34</td>
<td>0.9384</td>
<td>39.44</td>
</tr>
<tr>
<td>NLRN [33]</td>
<td>×2</td>
<td>38.00</td>
<td>0.9603</td>
<td>33.46</td>
<td>0.9159</td>
<td>32.19</td>
<td>0.8992</td>
<td>31.81</td>
<td>0.9249</td>
<td>–</td>
</tr>
<tr>
<td>RNAN [51]</td>
<td>×2</td>
<td>38.17</td>
<td>0.9611</td>
<td>33.87</td>
<td>0.9207</td>
<td>32.32</td>
<td>0.9014</td>
<td>32.73</td>
<td>0.9340</td>
<td>39.23</td>
</tr>
<tr>
<td>SRFBN [31]</td>
<td>×2</td>
<td>38.11</td>
<td>0.9609</td>
<td>33.82</td>
<td>0.9196</td>
<td>32.29</td>
<td>0.9010</td>
<td>32.62</td>
<td>0.9328</td>
<td>39.08</td>
</tr>
<tr>
<td>OISR [26]</td>
<td>×2</td>
<td>38.21</td>
<td>0.9612</td>
<td>33.94</td>
<td>0.9206</td>
<td>32.36</td>
<td>0.9019</td>
<td>33.03</td>
<td>0.9365</td>
<td>–</td>
</tr>
<tr>
<td>SAN [15]</td>
<td>×2</td>
<td>38.31</td>
<td>0.9620</td>
<td>34.07</td>
<td>0.9213</td>
<td>32.42</td>
<td>0.9028</td>
<td>33.10</td>
<td>0.9370</td>
<td>39.32</td>
</tr>
<tr>
<td>EDSR [32]</td>
<td>×2</td>
<td>38.11</td>
<td>0.9602</td>
<td>33.92</td>
<td>0.9195</td>
<td>32.30</td>
<td>0.9013</td>
<td>32.93</td>
<td>0.9351</td>
<td>39.10</td>
</tr>
<tr>
<td>NLSN (ours)</td>
<td>×2</td>
<td>38.34</td>
<td>0.9618</td>
<td>34.08</td>
<td>0.9231</td>
<td>32.43</td>
<td>0.9027</td>
<td>33.42</td>
<td>0.9394</td>
<td>39.59</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Scale</th>
<th>Set5</th>
<th>SSIM</th>
<th>Set14</th>
<th>SSIM</th>
<th>B100</th>
<th>Urban100</th>
<th>SSIM</th>
<th>Manga109</th>
<th>SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>LapSRN [30]</td>
<td>×3</td>
<td>33.82</td>
<td>0.9227</td>
<td>29.87</td>
<td>0.8320</td>
<td>28.82</td>
<td>0.7980</td>
<td>27.07</td>
<td>0.8280</td>
<td>32.21</td>
</tr>
<tr>
<td>MemNet [41]</td>
<td>×3</td>
<td>34.09</td>
<td>0.9248</td>
<td>30.00</td>
<td>0.8350</td>
<td>28.96</td>
<td>0.8001</td>
<td>27.56</td>
<td>0.8376</td>
<td>32.51</td>
</tr>
<tr>
<td>SRMDNF [49]</td>
<td>×3</td>
<td>34.12</td>
<td>0.9254</td>
<td>30.04</td>
<td>0.8382</td>
<td>28.97</td>
<td>0.8025</td>
<td>27.57</td>
<td>0.8398</td>
<td>33.00</td>
</tr>
<tr>
<td>RDN [52]</td>
<td>×3</td>
<td>34.71</td>
<td>0.9296</td>
<td>30.57</td>
<td>0.8468</td>
<td>29.26</td>
<td>0.8093</td>
<td>28.80</td>
<td>0.8653</td>
<td>34.13</td>
</tr>
<tr>
<td>RCAN [50]</td>
<td>×3</td>
<td>34.74</td>
<td>0.9299</td>
<td>30.65</td>
<td>0.8482</td>
<td>29.32</td>
<td>0.8111</td>
<td>29.09</td>
<td>0.8702</td>
<td>34.44</td>
</tr>
<tr>
<td>NLRN [33]</td>
<td>×3</td>
<td>34.27</td>
<td>0.9266</td>
<td>30.16</td>
<td>0.8374</td>
<td>29.06</td>
<td>0.8026</td>
<td>27.93</td>
<td>0.8453</td>
<td>–</td>
</tr>
<tr>
<td>RNAN [51]</td>
<td>×3</td>
<td>34.66</td>
<td>0.9290</td>
<td>30.52</td>
<td>0.8462</td>
<td>29.26</td>
<td>0.8090</td>
<td>28.75</td>
<td>0.8646</td>
<td>34.25</td>
</tr>
<tr>
<td>SRFBN [31]</td>
<td>×3</td>
<td>34.70</td>
<td>0.9292</td>
<td>30.51</td>
<td>0.8461</td>
<td>29.24</td>
<td>0.8084</td>
<td>28.73</td>
<td>0.8641</td>
<td>34.18</td>
</tr>
<tr>
<td>OISR [26]</td>
<td>×3</td>
<td>34.72</td>
<td>0.9297</td>
<td>30.57</td>
<td>0.8470</td>
<td>29.29</td>
<td>0.8103</td>
<td>28.95</td>
<td>0.8680</td>
<td>–</td>
</tr>
<tr>
<td>SAN [15]</td>
<td>×3</td>
<td>34.75</td>
<td>0.9300</td>
<td>30.59</td>
<td>0.8476</td>
<td>29.33</td>
<td>0.8112</td>
<td>28.93</td>
<td>0.8671</td>
<td>34.30</td>
</tr>
<tr>
<td>EDSR [32]</td>
<td>×3</td>
<td>34.65</td>
<td>0.9280</td>
<td>30.52</td>
<td>0.8462</td>
<td>29.25</td>
<td>0.8093</td>
<td>28.80</td>
<td>0.8653</td>
<td>34.17</td>
</tr>
<tr>
<td>NLSN (ours)</td>
<td>×3</td>
<td>34.85</td>
<td>0.9306</td>
<td>30.70</td>
<td>0.8485</td>
<td>29.34</td>
<td>0.8117</td>
<td>29.25</td>
<td>0.8726</td>
<td>34.57</td>
</tr>
</tbody>
</table>

Tries. This is mainly because NLSA only attends to content-correlated locations, which yields a more accurate correlation estimation. It is worth noting that all these benefits come only at an expense of adding a small amount of computation, which roughly equals to a few convolution operations, demonstrating that our NLSA is indeed effective and efficient. Visual results on Urban100 are shown in Figure 4. Our NLSN effectively restores the image details by efficiently utilizing global similar patches.

4.4. Ablation Study

In this section, we conduct controlled experiments to analyze the proposed NLSA. We build the baseline model with 16 residual blocks. For each attention variants, we insert a corresponding block after the 8-th residual block.

Size $k$ of the Attention Bucket. As discussed above, the sparsity of NLSA is controlled by the size of the attention bucket $k$ (chunk size). And if most correlated elements are successfully identified, a small $k$ should be sufficient for producing high quality super resolution. Here we investigate the effects of different $k$ and compare it with standard Non-Local Attention that have local window receptive fields covering exactly $k$ elements. Specifically, we set $k = \{5^2, 10^2, 15^2, 20^2, 25^2, 30^2, 40^2, 50^2\}$. As shown in Figure 5, the performance of NLSA peaks at $k = 10^2$, which significantly outperforms the NLA of the same cov-
Figure 4. Visual comparison for 4× SR on Urban100 dataset.

Figure 5. Ablation study on size $k$ of the attention bucket.

When further enlarging the attention bucket, NLSA becomes much denser. Its performance also begins to approximate the standard NLA as expected. This is mainly because a larger $k$ reduces the effectiveness of LSH by decreasing the number of hash bits and also is more likely to make chunks across multiple bucket boundaries. In an extreme case when $k$ equals to the number of image pixels, NLSA will be just identical to the standard Non-Local Attention.

**Similarity Grouping via Spherical LSH.** Spherical LSH partitions the space into attention buckets of correlated elements and it plays a key role in NLSA. We investigate its effectiveness by comparing it with other partition options. We first compare it with random hashing, i.e., elements are assigned to random buckets. As shown in Table 2, random hashing brings no evident improvement over the baseline as expected. In contrast, Spherical LSH achieves more than 0.1 dB improvements over the random version and baseline.

**Table 2. Comparison of partition strategies on Set5 (×2).**

<table>
<thead>
<tr>
<th>Partition</th>
<th>Baseline</th>
<th>Local(q=12)</th>
<th>Random</th>
<th>Spherical LSH</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
<td>37.78</td>
<td>37.83</td>
<td>37.79</td>
<td>37.92</td>
</tr>
</tbody>
</table>

Table 2. Comparison of partition strategies on Set5 (×2).
We also implement the local window strategy, where elements are gathered purely depending on locations. We set the window size \( q = 12 \), which equals to the attention range \( (k = 144) \) for fair comparison. Table 2 shows Spherical LSH is superior than local window strategy. This indicates Spherical LSH indeed effectively identifies more useful global cues beyond a local neighborhood.

Multi-round NLSA As discussed above, hashing for more rounds improves the robustness of NLSA but at a price of linearly increasing computational cost. Fortunately, the attention rounds \( r \) can be flexibly adjusted during testing. Results of the models trained and evaluated with different rounds are presented in Table 3. The results indicate that increasing the hashing rounds at either training or evaluation can constantly improve super-resolution accuracy. As expected, the best result is achieved with the largest round number during both training and testing, but it also yields the worst computational cost.

Shared linear projection in embedded Gaussian. NLSA uses shared linear projections in embedded Gaussian to estimate pair-wise similarity, i.e., \( \theta \neq \phi \). We investigate its effect on a standard Non-Local Attention. As shown in Table 4, the model with shared linear projection produces comparable and even slightly better results than the one not shared. In other words, this modification does not bring any harm to the performance.

Table 3. Ablation study on the attention rounds on Set5 \((\times 2)\). \( r \) denotes the number of attention rounds.

<table>
<thead>
<tr>
<th>Method</th>
<th>( r = 1 )</th>
<th>( r = 2 )</th>
<th>( r = 4 )</th>
<th>( r = 8 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>37.86</td>
<td>37.87</td>
<td>37.88</td>
<td>37.89</td>
</tr>
<tr>
<td>Test</td>
<td>37.87</td>
<td>37.88</td>
<td>37.89</td>
<td>37.90</td>
</tr>
</tbody>
</table>

Table 4. Effect of sharing linear projection on Set5 \((\times 2)\).

<table>
<thead>
<tr>
<th>Linear Projection</th>
<th>Baseline</th>
<th>( \theta \neq \phi )</th>
<th>( \theta = \phi )</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
<td>37.78</td>
<td>37.86</td>
<td>37.87</td>
</tr>
</tbody>
</table>

![Figure 6. Effect of size of receptive fields on Set5 \((\times 2)\).](image)

Robustness. Unlike standard Non-Local Attention, our NLSA is inherently robust to receptive fields, since it avoids attending to less-informative and less-correlated locations. As shown in Figure 6, both NLSA and NLA are confined to local windows. It suggests that, for a same window size, our NLSA constantly outperforms NLA. This proves that NLSA captures more accurate correlations and is more robust to noisy information. Further increasing the receptive fields also slightly improves NLSA as it can capture additional information from longer-range contexts. In contrast, the performance of Non-Local Attention gradually drops as more information is taken into account, which hurts its correlation estimation.

Efficiency. We compare the NLSA with standard Non-Local Attention in terms of computational efficiency. Table 5 reports the incremental computational cost and the associated performances. The input is assumed to have spatial size of \( 100 \times 100 \) and both input and output channels are set to 64. We also add an entry of normal \( 3 \times 3 \) convolution operation for better illustration. As shown in Table 5, NLSA significantly reduces the computational cost of the NLA while obtaining superior performance. For example, the most efficient single round NLSA-r1 has a similar computational cost of convolution, but achieves comparable performances with standard NL operation. The best result is achieved by NLSA-r8 with 8 rounds attention, which is still roughly 3 times more efficient than the standard Non-Local Attention. More comparisons of the efficiency are provided in the supplementary material.

5. Conclusion

In this paper, we propose a novel Non-Local Sparse Attention (NLSA) for deep single image super-resolution networks, that simultaneously embraces the benefits of sparse representation and non-local operation. NLSA globally identifies the most informative locations to attend without paying any attention to unrelated regions, resulting in a robust and efficient global modeling operation. Further inserting it into deep networks, our non-local sparse network sets new state-of-the-arts on multiple benchmarks. Extensive evaluations suggest that our NLSA is a superior operation over the standard non-local attention and indeed beneficial for accurate image super resolution.
References


[23] Aristides Gionis et al. Similarity search in high dimensions via hashing. 2


[50] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In ECCV, 2018. 5, 6, 7

