NGSWIN: N-GRAM SWIN TRANSFORMER FOR EFFICIENT SINGLE IMAGE SUPER-RESOLUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

In single image super-resolution (SISR), many deep learning-based methods suffer from intensive computational operations. In addition, while Swin Transformer-based methods such as SwinIR established state-of-the-art results, they still hold the problem of ignoring the broad regions when computing window self-attention (WSA) to reconstruct high-frequency information. In this paper, we propose the efficient NGswin network, which is the first attempt in history to introduce N-Gram to deep learning in images. For text analysis, N-Gram is a sequence of consecutive characters or words, but in an image, we define N-Gram as neighboring local windows (in WSA of Swin Transformer) which interact with each other by *sliding-WSA*. We propose N-Gram interaction, SCDP bottleneck, and a pooling-cascading mechanism, which enable the network to consider broad regions beneficial to recovering the degraded neighbor pixels. Moreover, we employ a hierarchical encoder with patch-merging, uni-Gram embedding, and a compact decoder to NGswin to enhance the network efficiency. Experimental results show that the proposed model achieves competitive performance in terms of PSNR and SSIM scores with fewer operations (Mult-Adds) compared to other methods.

1 INTRODUCTION

Deep learning-based methods have improved performance in single image super-resolution (SISR), which aims to reconstruct high-resolution (HR) images from low-resolution (LR) images. However, recent state-of-the-art (Chen et al., 2021; Mei et al., 2021; Liang et al., 2021; Zhang et al., 2022) and lightweight SR networks (Lu et al., 2021; Zhang et al., 2021; Du et al., 2022) require intensive computations. Although the network having small number of parameters is practically more applicable, the contemporary semi-conductor system can tolerate a certain level of memory consumption (e.g., 1M parameters, 4MB) to fetch the network weights from the memory unit (SSD or RAM) to the processing unit (CPU or GPU) (Han et al., 2021). If keeping the parameters around that level, fewer operations are important for real-world applications concerning time. Meanwhile, as Swin Transformer (Liu et al., 2021) integrates locality of convolutional neural network and long-range dependency of Vision Transformer (Dosovitskiy et al., 2020) by window self-attention (WSA), some studies have employed it to restore HR images (Liang et al., 2021; Fang et al., 2022). Nevertheless, Swin Transformer has a critical limitation that the information of the broad regions in neighboring local windows would not be utilized for inferring high-frequency information, due to the shift size.

To overcome these problems, we propose a efficient approach, N-Gram Swin Transformer (**NGswin**). As illustrated in Fig. 1, NGswin is composed of five components: a shallow network, three hierarchical encoder stages with NSTBs (N-Gram Swin Transformer Blocks), SCDP Bottleneck (pixel-Shuffle, Concatenation, Depth-wise convolution, Point-wise projection), one decoder stage with NSTBs, and Reconstruction module. We adapt WSA of Swin Transformer, hierarchical architecture of U-Net (Ronneberger et al., 2015), and cascading mechanism of CARN (Ahn et al., 2018). To the best of our knowledge, this is the first attempt in history to introduce N-Gram to deep learning in the vision domain. As stated in He et al. (2022), images have heavy spatial redundancy, which means that some degraded pixels can be recovered from contextual information of neighboring pixels or patches. Similar to N-Gram language models that take into account the longer span of neighboring words beyond a single word, the longer context can enhance image reconstruction as well as text analysis. To take advantage of the extensive information from neighboring pixels,

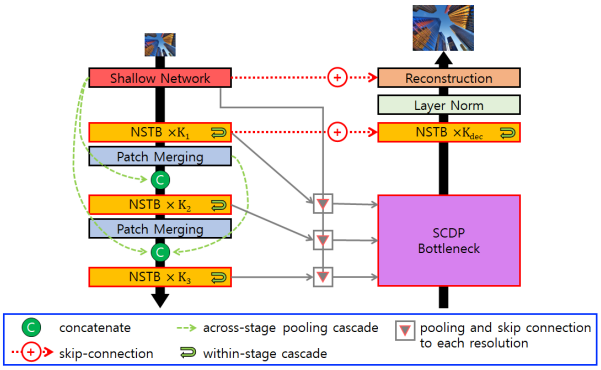


Figure 1: Overall architecture of NGswin. We adopt asymmetric encoder-decoder U-Net architecture. NSTB stands for N-Gram Swin Transformer Block. SCDP Bottleneck is composed of pixel-Shuffle, Concatenation, Depth-wise convolution, and Point-wise projection. It is a variant of bottleneck of U-Net, which takes all outputs of encoder stages including the initial shallow network.

the output of proposed N-Gram interactions by *sliding-WSA* is added to each local window before WSA. To decrease self-attention operations in N-Gram interaction, uni-Grams are embedded by a group convolutional layer before interacting with each other. We demonstrate that N-Gram context utilizes the information in broader regions for restoring the degraded pixels.

The main contributions of this paper are summarized as follows:

(1) We propose N-Gram interactions in the image domain by *sliding-WSA*, a variant of bottleneck structure, and pooling-cascading mechanism, all of which are crucial for achieving competitive performance in the efficient SISR on $\times 2$, $\times 3$, and $\times 4$ upscaling tasks.

(2) For a more efficient network while keeping a tolerable size of parameters, we exploit the hierarchical encoder with *patch-merging* and an asymmetrically smaller decoder. We also embed uni-Gram context by group convolution for efficient calculation of N-Gram interactions.

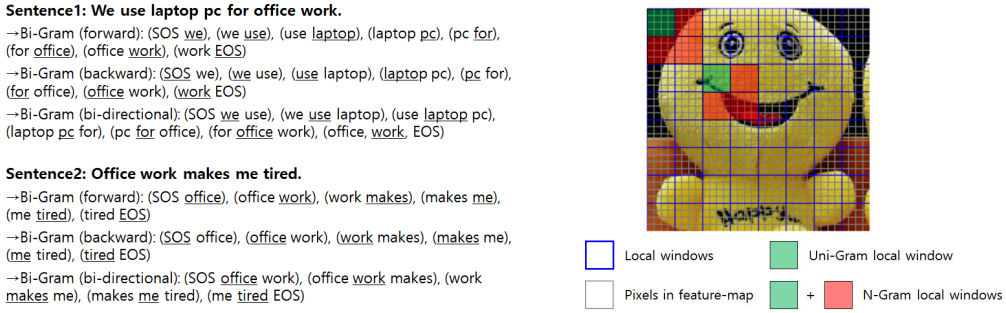
2 Related Work

2.1 Efficient Single Image Super-Resolution (SISR)

Many SISR studies have striven to increase network efficiency. CARN (Ahn et al., 2018) introduced cascading residual blocks. IMDN (Hui et al., 2019) used information multi-distillation and selective feature fusion. RFDN (Liu et al., 2020) proposed residual feature distillation for lightening IMDN. ESRT (Lu et al., 2021) combined convolutional neural network (CNN) and channel-reducing Transformer (Vaswani et al., 2017). SwinIR-light (Liang et al., 2021) additionally appended CNN on Swin Transformer backbone (Liu et al., 2021). SRPN-lite (Zhang et al., 2021) applied the network pruning technique (Reed, 1993) on EDSR-baseline (Lim et al., 2017), a CNN-based lightweight SR model. Most recently, ELAN-light (Zhang et al., 2022) utilized group-wise multi-scale self-attention. Although these methods successfully reduced the number of parameters, they still required intensive operations, which is a drawback for real-world applications. Since contemporary semiconductors can tolerate a certain memory level (Han et al., 2021) (e.g., around 1M parameters, about 4MB in size), computational operation is a key factor for evaluating the network efficiency.

2.2 N-Gram Language Model in Deep Learning

N-Gram is a sequence of consecutive characters or words, of which the length (or size) N is typically set to 2 or 3 (Majumder et al., 2002). The N-Gram language model (LM) considering a longer span of context in sentences was operating well statistically. Although this method seemed dominated by recent deep learning-based LM, some researchers still adopted N-Gram to effectively understand text data even in the deep learning methods. Sent2Vec (Pagliardini et al., 2017) used N-Gram embeddings by averaging word-embedding to learn sentence embedding. To learn the sen-



(a) N-Gram in text (b) N-Gram in image

Figure 2: N-Gram in text and image ($N = 2$). **(a)** SOS and EOS signify the start and the end of the sentence. Underlined words in groups are the target words and non-underlined words are N-Gram neighbors of the target word. **(b)** Each local window (referred from SwinV1&V2) is defined as uni-Gram. The right/lower local windows are defined as forward N-Gram neighbors.

tence representation better, Lopez-Gazpio et al. (2019) computed word N-Gram context by recurrent neural network (RNN) and passed it to the attention layer. ZEN (Diao et al., 2019; Song et al., 2021) trained a BERT-styled (Devlin et al., 2018) N-Gram encoder for all possible character N-Grams from Chinese or Arabic lexicon to convey salient N-Grams to another encoder, the character encoder.

2.3 Swin Transformer V1 & V2

Swin Transformer (Liu et al., 2021) (SwinV1) proposed window self-attention (WSA) that computes self-attention only within non-overlapping local windows, to avoid quadratic time-complexity to the resolution of feature-map. To compensate for WSA that lacks interaction across windows, SwinV1 also proposed shifted window self-attention (SWSA) in consecutive layers. The revised version (Liu et al., 2022) (SwinV2) modified SwinV1. First, for the advanced model capacity with milder optimization, SwinV2 introduced residual post normalization and scaled cosine attention, instead of pre-normalization configuration and scaled-dot-product attention. Second, for better transfer-learning regardless of the size of images or windows, it replaced the relative position bias with the log-spaced continuous position bias. As we find that SwinV2 backbone is more effective than SwinV1 for NGswin, we adopt SwinV2 rather than SwinV1.

Limitations. The shift size ($\lfloor \frac{M}{2} \rfloor$ on paper; M: window size) in SwinV1 and V2 is critically vulnerable, in that the shift size could not be over M. It causes the considerably broad regions in neighboring local windows to be ignored when computing (S)WSA. Though the weakness can be compensated in the deeper layers, information helpful for reconstruction task cannot be utilized for producing high-frequency information in the shallower layers taking HR inputs.

3 Methodology

3.1 Definition of N-Gram Context in Image

N-Gram in text. As shown in Fig. 2(a), the N-Gram language model views the consecutive forward, backward, or bi-directional words as N-Gram of the target word. The words are independent with each other for uni-Gram (i.e., word-embedding), but they interact with each other by RNN or Attention when considering N-Gram. In contrast, an N-Gram composed of the same words (e.g., "office work" in Fig. 2(a)) never interact with the other N-Gram combinations.

N-Gram in image. Similarly, N-Gram in an image should have the aforementioned properties. Thus, we define a uni-Gram as a non-overlapping local window of Swin Transformer, within which the pixels of feature-maps interact with each other by self-attention. N-Gram is defined as the larger window including neighbors of each uni-Gram. As depicted in Fig. 2(b), setting N-Gram size N to 2 indicates a bi-Gram that combines a local window (green area) and its neighboring windows (red areas) at lower-right. The interaction within the N-Gram will be explained in sec. 3.3.

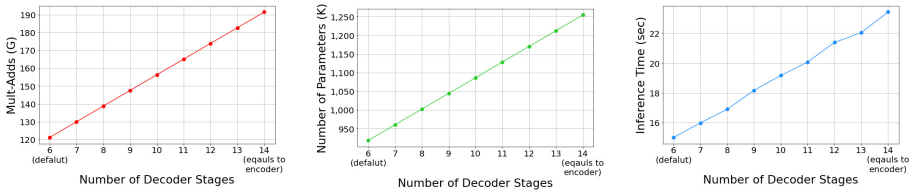


Figure 3: Efficiency of asymmetrically smaller decoder in terms of the operations, the number of parameters, and the inference time. Mult-Adds is evaluated on $\times 2$ task with a 1280×720 HR image. The inference time is estimated on doubling 10 LR images (640×360) into HR images (1280×720) on a single NVIDIA TITAN Xp GPU.

Table 1: Computational complexity of NGswin. **(left)** Comparisons of operations with state-of-the-art networks. **(right)** Number of operarions for each hierarchical encoder and a compact decoder. *res.* and *dep.* indicate the resolution of input and the number of NSTBs, respectively.

scale	**NGswin (ours)**	SwinIR-light[2]	ESRT	ELAN-light
x2	**140.4G**	243.7G	191.4G	168.4G
x3	**66.5G**	109.5G	96.4G	75.7G
x4	**36.4G**	61.7G	67.7G	43.2G

scale	stage	Mult-Adds	*res.*	*dep.*
$\times 2$	encoder 1	61.91G	64×64	6
	encoder 2	11.05G	32×32	4
	encoder 3	2.76G	16×16	4
	decoder	60.84G	64×64	6

3.2 Overall Architecture

As illustrated in Fig. 1, we adopt U-Net (Ronneberger et al., 2015) architecture; hierarchical encoder stages, a bottleneck layer, a decoder stage, and skip-connection from encoder to decoder with same feature resolution[1]. However, the encoder and decoder in our network are asymmetric, which indicates significantly smaller decoder stages are employed (He et al., 2022; Pang et al., 2022). This asymmetry highly enhances the efficiency, as shown in Fig. 3 and right of Tab. 1.

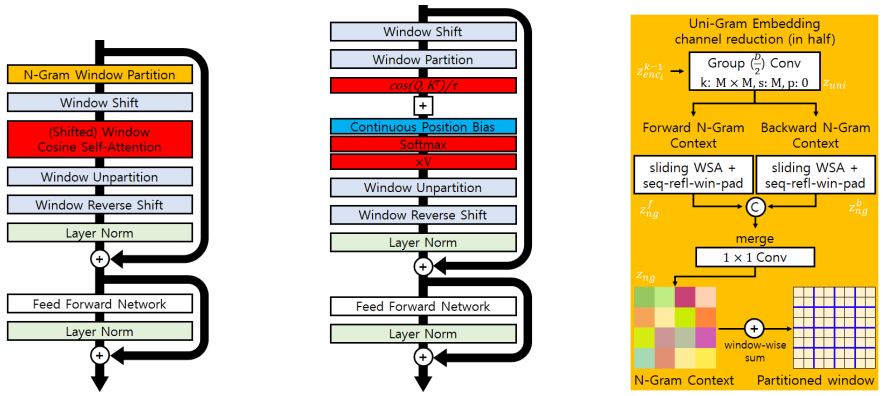
Encoder. Given a low-resolution (LR) image $I_{LR} \in \mathbb{R}^{3\times H\times W}$, a shallow network $G_s(\cdot)$ (Liang et al., 2021) extracts $z_s \in \mathbb{R}^{D\times H\times W}$, where H, W, and D stand for height, width, and network dimension (channels), respectively. This module provides stable optimization and easy mapping to feature space (Liang et al., 2021). z_s is passed through three encoder stages, each composed of K_i N-Gram Swin Transformer Blocks (NSTB, sec. 3.3) and a 2×2 *patch-merging* (except the last stage). $\{K_1, K_2, K_3\}$ is set to $\{6, 4, 4\}$ by default. The mapping function of k-th $(1 \sim K_i)$ NSTB in i-th $(1, 2, 3)$ encoder stage is formulated as:

$$z^k_{enc_i} = G^k_{enc_i}(z^{k-1}_{enc_i}),\ z^{k-1}_{enc_i} \in \mathbb{R}^{HW/(2^{i-1})^2\times D}, \tag{1}$$

where $z^0_{enc_1}$ equals to z_s, and $z^0_{enc_i}$ equals to $z_{enc_{i-1}}$, which results from downsampling $z^{K_{i-1}}_{enc_{i-1}}$. In other words, the first NSTB in the 2nd or 3rd stage takes the output of *patch-merging* in the previous stage as input. The *patch-merging* follows Swin Transformer (Liu et al., 2021), except the network dimension is decreased from $4D$ to D instead of $2D$. Since *patch-merging* reduces the resolutions, NGswin consumes much fewer Attention computations than state-of-the-art methods, as revealed in Tab. 1 (Liang et al., 2021; Lu et al., 2021; Zhang et al., 2022, respectively on the left table). Mult-Adds is evaluated on a 1280×720 HR image.

Following the global cascading in CARN (Ahn et al., 2018), we place a cascading mechanism (the light green dotted lines in Fig. 1) between all the encoder stages including the shallow network. However, the resolutions of features are halved as the stage goes deeper, so we place 2×2 max-pooling layers before concatenating the intermediary features, unlike CARN. Cascading gives the effect of reflecting the flow of the information and gradient in the previous layers, which helps the network to learn more meaningful representations and be easily optimized with few additional computations. We denote it as *across-stage-pooling-cascading*, distinguishing it from another cascading mechanism between NSTBs in the same stage (sec. 3.3). More detailed algorithms of cascading are provided in A.2.

[1] In this paper, "resolution" indicates height and width of features, excluding network dimension (channel).
[2] We correct Mult-Adds underestimated on a 1024×720 HR image in the original SwinIR paper.



(a) NSTB (proposed) (b) SwinV2 Block (c) N-Gram Window Partition

Figure 4: NSTB architecture. The sandwich between red blocks in **(b)** is same as the red block in **(a)**, except that NSTB does not utilize the continuous position bias (blue block in **(b)**). Instead, we use relative position bias following SwinV1. *cos* and τ in **(b)** are cosine-similarity and a learnable scalar. **(c)** is N-Gram window-partitioning, the top block of **(a)**. k, s, p, and M in **(c)** stand for kernel size, stride, padding, and local window size, respectively. The channel and resolution reduction through uni-Gram embedding enables the efficient WSA between N-Gram neighbors. Note that WSA weights are shared between forward and backward N-Gram context computations.

Bottleneck. All intermediary outputs from encoders including the shallow network are simultaneously taken as inputs, and they are mapped into $z_{scdp} \in \mathbb{R}^{HW \times D}$ by a bottleneck layer $G_{scdp}(\cdot)$. More detailed explanation is in sec. 3.4.

Decoder. z_{scdp} is fed into a single decoder stage $G_{dec}(\cdot)$, which is much smaller than the encoder. It contains K_{dec} NSTBs and a final layer-norm (LN) (Ba et al., 2016) which allows stable learning. The decoder NSTB architecture is the same as encoder NSTB. As shown in Fig. 1, the input of decoder is residually connected with $z^{K_1}_{enc_1}$ from the first encoder stage. z_s and decoder output $z_{dec} \in \mathbb{R}^{HW \times D}$ are added with a global skip-connection (Kim et al., 2016; Liang et al., 2021; Ahn et al., 2022). The global skip-connection boosts optimization and allows reconstruction module $G_{rec}(\cdot)$ to utilize both locality and long-range dependency for producing super-resolution images.

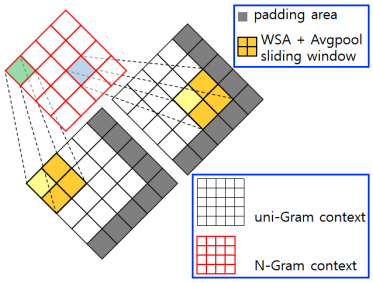
Reconstruction. Following Kim et al. (2016); Lim et al. (2017); Ahn et al. (2018; 2022); Liang et al. (2021), the final reconstruction module contains a convolutional layer that adjusts dimension, an upsampling pixel-shuffler (Shi et al., 2016), and a convolutional layer that produces the super-resolution image $I_{SR} \in \mathbb{R}^{3 \times rH \times rW}$, where r is an upscale factor (e.g., $\times 4$). For more, see A.3.

3.3 N-Gram Swin Transformer Block (NSTB)

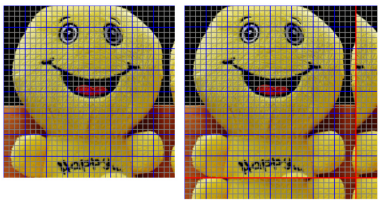
As shown in Fig. 4(a) and (b), NSTB shares the most components with SwinV2 (Liu et al., 2022) block, except for the continuous position bias proposed for scalable fine-tuning on the larger image or window size. Since SR models are commonly trained on fixed size (64×64) of image patches, we employ relative position bias of SwinV1 (Liu et al., 2021). In window partitioning (the top of Fig. 4(a)), the novel N-Gram context algorithm is adapted by following 4 steps, as illustrated in Fig. 4(c). As previously mentioned, the input to k-th NSTB in i-th encoder stage is $z^{k-1}_{enc_i}$.

First, $z^{k-1}_{enc_i}$ is embedded into uni-Gram ($N = 1$) context $z_{uni} \in \mathbb{R}^{\frac{D}{2} \times w_h \times w_w}$ by $M \times M$ group convolution (stride: M, padding: 0, groups: $\frac{D}{2}$). M is the window size. $w_h (= \frac{h}{M})$ and $w_w (= \frac{w}{M})$ represent the number of windows in height and width. h and w are the resolution of $z^{k-1}_{enc_i}$.

Second, the $N \times N$ pixels in each N-Gram ($N > 1$) of z_{uni} interact with each other by window self-attention (WSA) and $N \times N$ average-pooling, thereby producing the forward N-Gram feature z^f_{ng}. For this step, *sliding-WSA* (Fig. 5(a)) is implemented as *sliding-window-convolution* done in CNN. It computes self-attention in each $N \times N$ window, rather than convolution. As illustrated in Fig. 5(b), $(N - 1)$ size of paddings are applied on the lower-right side of z_{uni} by sequentially



(a) forward N-Gram sliding WSA + Avgpool



(b) sequentially-reflected-window-padding (*seq-refl-win-pad*)

Figure 5: **(a)** As the window slides through uni-Gram context, WSA and avg-pool are applied in each window, to get N-Gram context. **(b)** (**left**) Before padding. (**right**) After padding. The areas at the right/lower side out of the red lines are padded areas. In both cases **(a)** and **(b)**, $N = 2$.

reflected window padding (*seq-refl-win-pad*). Based on the outermost low/right windows, it uses the upper/left $(N-1)$ rows/columns of windows as padding values. It allows some uni-Grams to interact with their padded neighbors, instead of trivial "zero" padding values. Subsequently, backward N-Gram feature z^b_{ng} can be obtained by reversed *seq-refl-win-pad* (i.e., upper/left side padding). When calculating bi-directional N-Gram features, the *sliding-WSA* weights are shared. Note that since image is 2D data, our N-Gram can be seen from four directions (lower-right, lower-left, upper-right, upper-left), unlike text that can see max bi-direction. But, as shown in Tab. 5, trade-off between performance and efficiency is optimized when N-Gram neighbors are seen from bi-direction.

Third, z^f_{ng} and z^b_{ng} are concatenated and, then merged by a 1×1 convolutional layer to produce N-Gram context $z_{ng} \in \mathbb{R}^{D \times w_h \times w_w}$.

Finally, z_{ng} is added window-wise to the partitioned windows from $z^{k-1}_{enc_i}$. It is worth noting that the reduction of channel and resolution by uni-Gram embedding makes N-Gram WSA more efficient. Considering the complexity of WSA ($4hwD^2 + 2M^2hwD$), halved D and M^2 times reduced hw highly decrease computational burdens. The flow after window partitioning follows that of SwinV2 Block: cyclic window-shift, scaled cosine self-attention, window unpartioning, reversed window-shift, layer-norm, residual connection, feed-forward network, layer-norm, and residual connection.

The cascading mechanism exists between NSTBs within a stage (*within-stage-cascading*). It doesn't require pooling layers and concatenation of intermediary features. Instead, the NSTBs are simply residually connected, in both the encoder and decoder. This approach is illustrated in Fig. 8(c).

3.4 SCDP Bottlneck

Most SISR models (Zhang et al., 2018; Niu et al., 2020; Chen et al., 2021; Liang et al., 2021) commonly never use the hierarchical encoder, which downsamples the resolutions of features after one stage. As demonstrated in Tab. 6, it is because getting SR images from the space preserving high-resolution information richly has a significant advantage over from the space preserving insufficiently. However, since the resolution in our encoder is reduced highly due to the *patch-merging*, only using $z^{K_3}_{enc_3} \in \mathbb{R}^{HW/4^2 \times D}$ (from the last NSTB in the last encoder stage) as an input to the bottleneck makes it more challenging to restore high-frequency information.

In contrast with the bottleneck in standard U-Net architecture (Ronneberger et al., 2015; Wang et al., 2022) that takes the output of the last encoder layer, our SCDP bottleneck takes all outputs of the shallow network and the last NSTB in each encoder stage. SCDP stands for pixel-**S**huffle, **C**oncatenation, **D**epth-wise convolution, and **P**oint-wise projection. The steps are as followed. **First**, the outputs of the last NSTB in each encoder stage are upsampled into the resolution identical to that of I_{LR}, by pixel-shuffle layer (Shi et al., 2016). Mapping function is formulated as:

$$z'_{enc_i} = G_{shuffle}(z^{K_i}_{enc_i} + z^{(i)}_s), z^{K_i}_{enc_i} \in \mathbb{R}^{\frac{HW}{(2^{i-1})^2} \times D}, z'_{enc_i} \in \mathbb{R}^{HW \times \frac{D}{(2^{i-1})^2}}, \quad (2)$$

where $z^{(i)}_s$ comes from downsizing z_s into the resolution of each $z^{K_i}_{enc_i}$ by iterative 2×2 maxpoolings followed by LeakyReLU non-linearity, and is residually added to each $z^{K_i}_{enc_i}$. **Second**, all z'_{enc_i}

Table 2: Comparison of efficient (lightweight) super-resolution results. NGswin-*me* reduces window size (8×8 by default) to 4×4 and is made **m**ore **e**fficient. D2K stands for DIV2K dataset we use to train NGswin. DF2K indicates a merged dataset of D2K and Flickr2K (Timofte et al., 2017) containing 800 + 2,650 HR-LR image pairs. 291 images dataset is from Yang et al. (2010), Arbelaez et al. (2010). The best, second best, and third best performances are in red, blue, and underline.

Method	Scale	Training Dataset	Mult-Adds	#Params	Set5		Set14		BSD100		Urban100		Manga109	
					PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
EDSR-baseline	×2	D2K	316.3G	1,370K	37.99	0.9604	33.57	0.9175	32.16	0.8994	31.98	0.9272	38.54	0.9769
MemNet	×2	291	2,662.4G	677K	37.78	0.9597	33.28	0.9142	32.08	0.8978	31.31	0.9195	-	-
CARN	×2	D2K+291	222.8G	1,592K	37.76	0.9590	33.52	0.9166	32.09	0.8978	31.92	0.9256	38.36	0.9765
IMDN	×2	D2K	158.8G	694K	38.00	0.9605	33.63	0.9177	32.19	0.8996	32.17	0.9283	38.88	0.9774
LatticeNet	×2	D2K	169.5G	756K	38.15	0.9610	33.78	0.9193	32.25	0.9005	32.43	0.9302	-	-
RFDN-L	×2	D2K	145.8G	626K	38.08	0.9606	33.67	0.9190	32.18	0.8996	32.24	0.9290	38.95	0.9773
SRPN-Lite	×2	DF2K	139.9G	609K	38.10	0.9608	33.70	0.9189	32.25	0.9005	32.26	0.9294	-	-
FMEN	×2	DF2K	172.0G	748K	38.10	0.9609	33.75	0.9192	32.26	0.9007	32.41	0.9311	38.95	0.9778
HNCT	×2	D2K	82.4G	356K	38.08	0.9608	33.65	0.9182	32.22	0.9001	32.22	0.9294	38.87	0.9774
NGswin	×2	D2K	140.4G	998K	38.05	0.9609	33.78	0.9200	32.26	0.9007	32.44	0.9312	38.87	0.9777
NGswin-*me*	×2	D2K	122.3G	919K	38.00	0.9606	33.72	0.9189	32.21	0.9000	32.12	0.9285	38.77	0.9773
EDSR-baseline	×3	D2K	160.2G	1,555K	34.37	0.9270	30.28	0.8417	29.09	0.8052	28.15	0.8527	33.45	0.9439
MemNet	×3	219	2,662.4G	677K	34.09	0.9248	30.00	0.8350	28.96	0.8001	27.56	0.8376	-	-
CARN	×3	D2K+291	118.8G	1,592K	34.29	0.9255	30.29	0.8407	29.06	0.8034	28.06	0.8493	33.50	0.9440
IMDN	×3	D2K	71.5G	703K	34.36	0.9270	30.32	0.8417	29.09	0.8046	28.17	0.8519	33.61	0.9445
LatticeNet	×3	D2K	76.3G	765K	34.53	0.9281	30.39	0.8424	29.15	0.8059	28.33	0.8538	-	-
RFDN-L	×3	D2K	65.6G	633K	34.47	0.9280	30.35	0.8421	29.11	0.8053	28.32	0.8547	33.78	0.9458
SRPN-Lite	×3	DF2K	62.7G	615K	34.47	0.9276	30.38	0.8425	29.16	0.8061	28.22	0.8534	-	-
FMEN	×3	DF2K	77.2G	757K	34.45	0.9275	30.40	0.8435	29.17	0.8063	28.33	0.8562	33.86	0.9462
ESRT	×3	D2K	96.4G	770K	34.42	0.9268	30.43	0.8433	29.15	0.8063	28.46	0.8574	33.95	0.9455
HNCT	×3	D2K	37.2G	363K	34.47	0.9275	30.44	0.8439	29.15	0.8067	28.28	0.8557	33.81	0.9459
NGswin	×3	D2K	66.5G	1,007K	34.54	0.9284	30.52	0.8457	29.18	0.8077	28.47	0.8590	33.84	0.9468
NGswin-*me*	×3	D2K	54.7G	927K	34.40	0.9273	30.44	0.8439	29.13	0.8062	28.19	0.8536	33.67	0.9454
EDSR-baseline	×4	D2K	114.0G	1,518K	32.09	0.8938	28.58	0.7813	27.57	0.7357	26.04	0.7849	30.35	0.9067
MemNet	×4	291	2,662.4G	677K	31.74	0.8893	28.26	0.7723	27.40	0.7281	25.50	0.7630	-	-
CARN	×4	D2K+291	90.9G	1,592K	32.13	0.8937	28.60	0.7806	27.58	0.7349	26.07	0.7837	30.47	0.9084
IMDN	×4	D2K	40.9G	715K	32.21	0.8948	28.58	0.7811	27.56	0.7353	26.04	0.7838	30.45	0.9075
LatticeNet	×4	D2K	43.6G	777K	32.30	0.8962	28.68	0.7830	27.62	0.7367	26.25	0.7873	-	-
RFDN-L	×4	D2K	37.4G	643K	32.28	0.8957	28.61	0.7818	27.58	0.7363	26.20	0.7883	30.61	0.9096
SRPN-Lite	×4	DF2K	35.8G	623K	32.24	0.8958	28.69	0.7836	27.63	0.7373	26.16	0.7875	-	-
FMEN	×4	DF2K	44.2G	769K	32.24	0.8955	28.70	0.7839	27.63	0.7379	26.28	0.7908	30.70	0.9107
ESRT	×4	D2K	67.7G	751K	32.19	0.8947	28.69	0.7833	27.69	0.7379	26.39	0.7962	30.75	0.9100
HNCT	×4	D2K	21.5G	372K	32.31	0.8957	28.71	0.7834	27.63	0.7381	26.20	0.7896	30.70	0.9112
NGswin	×4	D2K	36.4G	1,019K	32.34	0.8966	28.72	0.7849	27.65	0.7394	26.35	0.7942	30.77	0.9127
NGswin-*me*	×4	D2K	33.2G	939K	32.18	0.8949	28.65	0.7831	27.60	0.7375	26.12	0.7874	30.57	0.9097

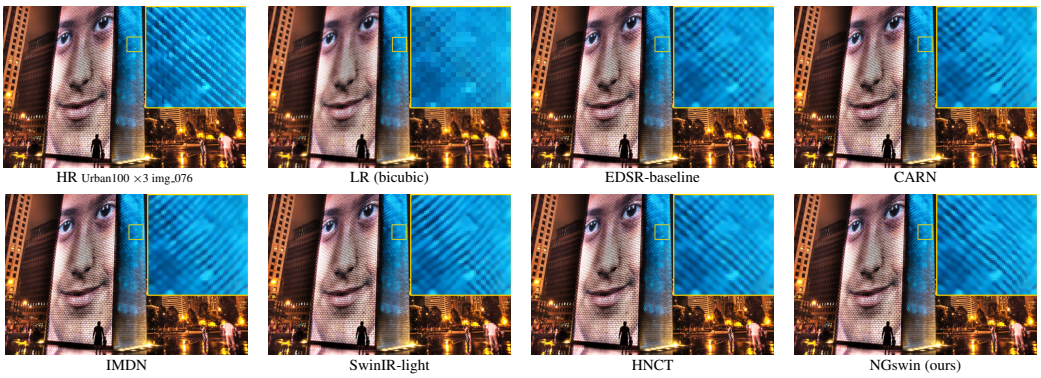


Figure 6: Visual Comparisons. More results are illustrated in A.5

are concatenated in channel (network dimension) space. **Third**, it passes through a depth-wise convolutional layer for learning spatial representations in each channel space. **Finally**, a point-wise linear projection layer is applied to adjust the dimension matching D.

4 Experiments

4.1 Experimental Setup

Training. We use 800 training HR-LR image pairs from DIV2K (Agustsson & Timofte, 2017) dataset. We randomly crop LR images into 64×64 size of patches, following the recent works (Liang et al., 2021; Fang et al., 2022). Our model minimizes L_1 pixel-loss between I_{SR} and the

Table 3: Comparison with SwinIR-light. *hier.* denotes if hierarchical network is used. Swin *dep.* and ver. denote the total number and version of Swin Transformer layers, respectively.

Method	Scale	Mult-Adds	#Params	Feature Enhancement	*hier.*	Swin *dep.*	Swin ver.	window size	Set5		Set14		BSD100	
									PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
SwinIR-light	×2	243.7G	910K	appended convolution	No	24	V1	8×8	38.14	0.9611	33.86	0.9206	32.31	0.9012
NGswin		140.4G	998K	N-Gram interaction	Yes	20	V2		38.05	0.9609	33.78	0.9200	32.26	0.9007
SwinIR-light	×3	109.5G	918K	appended convolution	No	24	V1	8×8	34.62	0.9289	30.54	0.8463	29.20	0.8082
NGswin		66.5G	1,007K	N-Gram interaction	Yes	20	V2		34.54	0.9284	30.52	0.8457	29.18	0.8077
SwinIR-light	×4	61.7G	930K	appended convolution	No	24	V1	8×8	32.44	0.8976	28.77	0.7858	27.69	0.7406
NGswin		36.4G	1,019K	N-Gram interaction	Yes	20	V2		32.34	0.8966	28.72	0.7849	27.65	0.7394

Table 4: Ablation study on N-Gram context use.

scale	N-Gram	Mult-Adds	#Params	Set14		Urban100		manga100	
				PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
×2	without / **with**	138.20G / **140.41G**	750K / **998K**	33.70 / **33.78**	0.9194 / **0.9200**	32.39 / **32.44**	0.9304 / **0.9312**	38.86 / **38.87**	0.9775 / **0.9777**
×3	without / **with**	65.48G / **66.52G**	759K / **1,007K**	30.48 / **30.52**	0.8452 / **0.8457**	28.37 / **28.47**	0.8573 / **0.8590**	33.81 / **33.84**	0.9464 / **0.9468**
×4	without / **with**	35.83G / **36.38G**	771K / **1,019K**	28.70 / **28.72**	0.7844 / **0.7849**	26.25 / **26.35**	0.7918 / **0.7942**	30.70 / **30.77**	0.9123 / **0.9127**

ground truth I_{HR}: $\mathcal{L} = \|I_{HR} - I_{SR}\|_1$, with Adam optimizer. Other experimental details and discussions are in A.4.

Evaluation. We test NGswin on the five popular benchmark datasets, composed of Set5 (Bevilacqua et al., 2012), Set14 (Zeyde et al., 2010), BSD100 (Martin et al., 2001), Urban100 (Huang et al., 2015), and Manga109 (Matsui et al., 2017). LR images are acquired by the MATLAB bicubic kernel from corresponding HR images matching each upscaling task. We use PSNR (dB) and SSIM scores on the Y channel of the YCbCr space as the metrics to evaluate reconstruction performance.

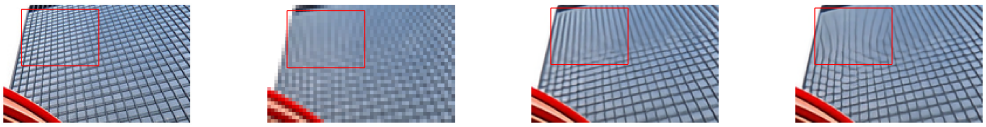
4.2 Comparisons with Leading Models

In Tab. 2, we compare NGswin with other efficient SISR models, including EDSR-baseline (Lim et al., 2017, CVPR2017 workshop), MemNet (Tai et al., 2017, ICCV2017), CARN (Ahn et al., 2018, ECCV2018), IMDN (Hui et al., 2019, ACMMM2019), LatticeNet (Luo et al., 2020, ECCV2020), RFDN-L (Liu et al., 2020, ECCV2020), SRPN-Lite (Zhang et al., 2021, ICLR2021), FMEN (Du et al., 2022, CVPR2022 workshop), ESRT (Lu et al., 2021, CVPR2022 workshop), and HNCT (Fang et al., 2022, CVPR2022 workshop). PSNR and SSIM scores on the three upscaling tasks with the five benchmark test sets are evaluated. Training dataset, Mult-Adds (evaluated on a 1280×720 HR image), and the number of parameters are reported for comparing the network efficiency. NGswin outperforms or matches previous leading models on all of the benchmarks, with relatively efficient structure. Note that although HNCT adapts Swin Transformer more efficiently than ours, it fails to optimize the trade-off between performance and efficiency. Additionally, our **m**ore **e**fficient model, NGswin-*me*, reduces window size (by default 8×8) to 4×4 and shows competitive results. The visual comparisons of SR results are supplied in Fig. 6 (more in A.5).

We conduct a comparative analysis of our model and SwinIR-light (Liang et al., 2021) (SwinIR) in various aspects, as shown in Tab. 3. NGswin requires about 1.65∼1.74 times fewer operations than SwinIR, and maintains around 1M parameters, which is tolerable for contemporary semiconductor. With fewer operations and modest number of parameters, our model achieved a close performance to SwinIR. Our efficiency results from the hierarchical encoder and the shallower depths, compared with SwinIR. Notice that SwinIR paper omitted relative position bias tables, a key component for WSA stated in Liu et al. (2021), from the number of parameters. Thus we correct the omission and include the positional bias tables of both methods in the total number of parameters.

4.3 Ablation Studies and Discussions

Tab. 4 demonstrates that our proposed N-Gram context is effective on SISR. Considering the margins of the metrics between NGswin and other models, it is impossible for NGswin to outperform other methods unless N-Gram context expands the regions seen for reconstruction. As proven, our introduction of N-Gram from text to image domain helps the network to utilize contextual information of each degraded pixel and to reconstruct high-frequency information. Especially, N-Gram context is useful in recovering highly distorted textures such as Urban100 dataset. Fig. 7 visualizes



HR Urban100 ×3 img_062 | LR (bicubic) | *w/* N-Gram | *w/o* N-Gram

Figure 7: Visual comparisons of *w/* and *w/o* N-Gram context. With N-Gram context, NGswin can consider broader regions to recover highly distorted textures.

Table 5: Ablation study on the number of N-Gram directions to be seen from. inf means divergence.

Model	N-Gram direction	uni-Gram dimension	directional weight sharing	Mult-Adds	#Params	Set14		Urban100		Manga109	
						PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
NGswin-*me*	1	D	yes	168.28G	1,158,312	33.72	0.9189	32.15	0.9289	38.78	0.9774
NGswin-*me*	4	$D/4$	yes	119.00G	855,528	inf	inf	inf	inf	inf	inf
NGswin-*me*	2 (default)	$D/2$	yes	122.27G	918,640	33.72	0.9189	32.12	0.9285	38.77	0.9773

Table 6: Ablation study on SCDP bottleneck. *dep.*: # of NSTBs / *resol.*: input resolution.

stages	SCDP	Mult-Adds	#Params	BSD100		encoder1		encoder2		encoder3		encoder4		decoder1		decoder2	
				PSNR	SSIM	dep.	resol.	dep.	resol.	dep.	resol.	dep.	resol.	dep.	resol.	dep.	resol.
extra	*w/o*	87.98G	997,064	32.20	0.8999	4	64×64	4	32×32	4	16×16	4	8×8	2	32×32	2	64×64
default	*w/o*	138.88G	992,464	32.23	0.9002	6		4		4		-	-	6	64×64	-	-
default	*w/*	140.41G	998,384	32.26	0.9007	6	64×64	4	32×32	4	16×16	-	-	6	64×64	-	-

this strength by contrasting the networks with and without N-Gram context. With N-Gram context, NGswin can expand sight and recover more accurate textures than an approach without N-Gram.

N-Gram direction in Tab. 5 counts how many directions are considered to see N-Gram neighbors. If it is set to 2, the algorithm is the same as the bi-direction explained in sec. 3.3. When set to 1, the network sees the neighbors from a lower-right direction only (uni-direction). For uni-directional N-Gram, uni-Gram embedding never reduces channels, which shows the reasonable performance but loses an efficient structure. In the case of 4 directions, neighbors from quad-direction including lower-right, lower-left, upper-right, and upper-left, are considered as N-Gram. For implementation of quad-directional N-Gram, uni-Gram embedding layer outputs $\frac{1}{4}$ times D and then all of $\frac{1}{4}D$ N-Gram contexts are concatenated into D dimension. But this setting ends up with network divergence (i.e., infinite loss) at around the $120\text{-}th$ epoch. We hypothesize that too many operations share the same weights of *sliding-WSA*, which makes back-propagation more challenging and unstable.

In Tab. 6, we examin negative effect of extra stages appended to the encoder and the decoder. While default and extra stages network have the same number of NSTBs, the extra stages handle lower resolution (8×8 in encoder and 32×32 in decoder). Those additionally reduced features drop the performance of SISR neural network, which is different from neural networks of high-level vision tasks such as ResNet50 (He et al., 2016) for classification. That is, because it is easier to reconstruct the high-frequency information from the space preserving HR information richly than from the space preserving HR information insufficiently. Our SCDP bottleneck takes all outputs (i.e., various resolutions) of the encoder stages. Thanks to this property of SCDP bottleneck, we can prevent the network from greatly reducing the performance.

5 Conclusion

In this paper, we introduce N-Gram context to deep learning in images for the first time in the history. The proposed approaches, including N-Gram interaction with *sliding-WSA*, SCDP bottleneck, and pooling-cascading mechanism, compensate for the limitation of Swin Transformer backbone in which the broad regions that are beneficial to recovering degraded pixels are not considered due to window size. To enhance the network efficiency, the hierarchical encoder with *patch-merging*, asymmetric decoder, and uni-Gram embedding are utilized, all of which significantly decrease the computational operations. With all the methods above, NGswin achieves better or close reconstruction performance, compared with other leading SISR methods. We demonstrate that the concept from other domain (text analysis) can help to solve the problem in vision domain. For the further works, we hope that our proposed N-Gram context succeeds on other low-level vision tasks, such as denosing, deblurring, and deraining.

REFERENCES

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

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

Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Efficient deep neural network for photo-realistic image super-resolution. *Pattern Recognition*, 127:108649, 2022.

Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):898–916, 2010.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie Line Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. 2012.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12299–12310, 2021.

Xiaokang Chen, Mingyu Ding, Xiaodi Wang, Ying Xin, Shentong Mo, Yunhao Wang, Shumin Han, Ping Luo, Gang Zeng, and Jingdong Wang. Context autoencoder for self-supervised representation learning. *arXiv preprint arXiv:2202.03026*, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Shizhe Diao, Jiaxin Bai, Yan Song, Tong Zhang, and Yonggang Wang. Zen: Pre-training chinese text encoder enhanced by n-gram representations. *arXiv preprint arXiv:1911.00720*, 2019.

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

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

Jinsheng Fang, Hanjiang Lin, Xinyu Chen, and Kun Zeng. A hybrid network of cnn and transformer for lightweight image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1103–1112, 2022.

Runze Han, Yachen Xiang, Peng Huang, Yihao Shan, Xiaoyan Liu, and Jinfeng Kang. Flash memory array for efficient implementation of deep neural networks. *Advanced Intelligent Systems*, 3 (5):2000161, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009, 2022.

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

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

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.

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

Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 136–144, 2017.

Zudi Lin, Prateek Garg, Atmadeep Banerjee, Salma Abdel Magid, Deqing Sun, Yulun Zhang, Luc Van Gool, Donglai Wei, and Hanspeter Pfister. Revisiting rcan: Improved training for image super-resolution. *arXiv preprint arXiv:2201.11279*, 2022.

Jie Liu, Jie Tang, and Gangshan Wu. Residual feature distillation network for lightweight image super-resolution. In *European Conference on Computer Vision*, pp. 41–55. Springer, 2020.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, 2021.

Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12009–12019, 2022.

Inigo Lopez-Gazpio, Montse Maritxalar, Mirella Lapata, and Eneko Agirre. Word n-gram attention models for sentence similarity and inference. *Expert Systems with Applications*, 132:1–11, 2019.

Zhisheng Lu, Hong Liu, Juncheng Li, and Linlin Zhang. Efficient transformer for single image super-resolution. *arXiv preprint arXiv:2108.11084*, 2021.

Xiaotong Luo, Yuan Xie, Yulun Zhang, Yanyun Qu, Cuihua Li, and Yun Fu. Latticenet: Towards lightweight image super-resolution with lattice block. In *European Conference on Computer Vision*, pp. 272–289. Springer, 2020.

P Majumder, M Mitra, and BB Chaudhuri. N-gram: a language independent approach to ir and nlp. In *International conference on universal knowledge and language*, 2002.

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

Yusuke Matsui, Kota Ito, Yuji Aramaki, Azuma Fujimoto, Toru Ogawa, Toshihiko Yamasaki, and Kiyoharu Aizawa. Sketch-based manga retrieval using manga109 dataset. *Multimedia Tools and Applications*, 76(20):21811–21838, 2017.

Yiqun Mei, Yuchen Fan, and Yuqian Zhou. Image super-resolution with non-local sparse attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3517–3526, 2021.

Ben Niu, Weilei Wen, Wenqi Ren, Xiangde Zhang, Lianping Yang, Shuzhen Wang, Kaihao Zhang, Xiaochun Cao, and Haifeng Shen. Single image super-resolution via a holistic attention network. In *European conference on computer vision*, pp. 191–207. Springer, 2020.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. Unsupervised learning of sentence embeddings using compositional n-gram features. *arXiv preprint arXiv:1703.02507*, 2017.

Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. *arXiv preprint arXiv:2203.06604*, 2022.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Russell Reed. Pruning algorithms-a survey. *IEEE transactions on Neural Networks*, 4(5):740–747, 1993.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

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

Yan Song, Tong Zhang, Yonggang Wang, and Kai-Fu Lee. Zen 2.0: Continue training and adaption for n-gram enhanced text encoders. *arXiv preprint arXiv:2105.01279*, 2021.

Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. Memnet: A persistent memory network for image restoration. In *Proceedings of the IEEE international conference on computer vision*, pp. 4539–4547, 2017.

Radu Timofte, Eirikur Agustsson, Luc Van Gool, Ming-Hsuan Yang, and Lei Zhang. Ntire 2017 challenge on single image super-resolution: Methods and results. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 114–125, 2017.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357. PMLR, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Zhendong Wang, Xiaodong Cun, Jianmin Bao, Wengang Zhou, Jianzhuang Liu, and Houqiang Li. Uformer: A general u-shaped transformer for image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17683–17693, 2022.

Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9653–9663, 2022.

Jianchao Yang, John Wright, Thomas S Huang, and Yi Ma. Image super-resolution via sparse representation. *IEEE transactions on image processing*, 19(11):2861–2873, 2010.

Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *International conference on curves and surfaces*, pp. 711–730. Springer, 2010.

Xindong Zhang, Hui Zeng, Shi Guo, and Lei Zhang. Efficient long-range attention network for image super-resolution. *arXiv preprint arXiv:2203.06697*, 2022.

Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 286–301, 2018.

Yulun Zhang, Huan Wang, Can Qin, and Yun Fu. Learning efficient image super-resolution networks via structure-regularized pruning. In *International Conference on Learning Representations*, 2021.

A Appendix

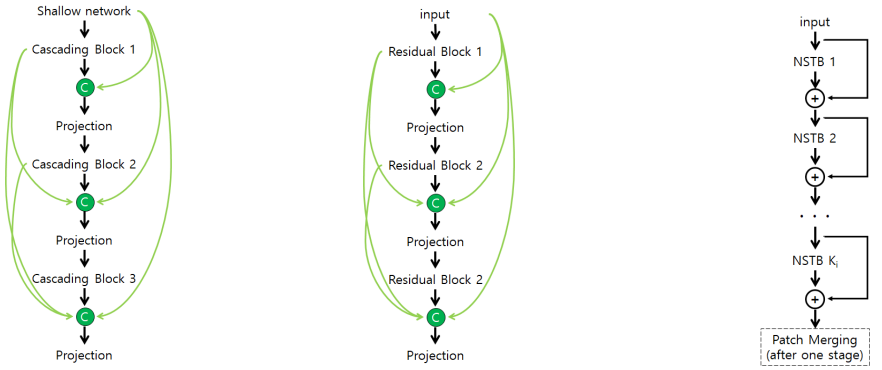
A.1 Motivation.

After Vision Transformer (ViT) (Dosovitskiy et al., 2020) was proposed, ViT family models such as DeiT (Touvron et al., 2021), T2T-ViT, and Swin Transformer (Liu et al., 2021; 2022) have showed outstanding performance in high-level vision tasks. In fact, these models originate from Transformer (Vaswani et al., 2017) in NLP domain, such as BERT (Devlin et al., 2018) and GPT-3 (Brown et al., 2020). More recently, encouraged by masked language modeling (MLM), the masked image models (MIM) are proposed, and have become a mainstream of self-supervised learning (Bao et al., 2021; He et al., 2022; Xie et al., 2022; Chen et al., 2022). We are hugely inspired by this domain-integration and investigate some approaches from other domain (NLP or audio) that can enhance understanding of low-level vision tasks. We find that although the N-Gram language model (LM) seems to hand over the throne to deep learning LM, some recent studies (Pagliardini et al., 2017; Lopez-Gazpio et al., 2019; Diao et al., 2019; Song et al., 2021) explored N-Gram methods in deep learning field and still showed comparable performance to the word-embedding based approaches. Motivated by these attempts, we assume that considering N-Gram can help to handle problems in SISR. As stated in He et al. (2022), heavy spatial redundancy property of images boosts reconstruction of some degraded (or even lost) pixels by use of contextual information of the neighboring pixels or patches. Therefore, we explore the impact of N-Gram context to enhance image restoration performance and achieve competitive performance. As further works, we will carry on examining domain-integration approaches to learn meaningful representation for low-level vision tasks.

A.2 Cascading Mechanism for SISR and Difference from NGswin

The cascading residual network (CARN) for SISR is proposed by Ahn et al. (2018). This approach gives the effect to reflect the flow of the information and gradient in the previous layers, which helps to get more meaningful representations and easy optimization, at not many extra computational costs. As shown in 8, all intermediary features from the encoder cascading blocks are accumulatively concatenated, and this aggregation is taken as input by the next cascading block, starting with a linear projection layer for dimensional reduction. Different from our *across-stage-pooling-cascading*, there is no pooling layer before concatenation. It is because CARN did not use hierarchical network, which makes NGswin have efficiently fewer operations. In addition, cascading connection between residual blocks in each cascading block is made following the same way above. While the former and the latter are named as the *global* and *local* cascading in the original paper (Fig. 8 (a) and (b)), we call them as *across-stage* and *within-stage* cascading (sec. 3.2 and 3.3), as to prevent confusion with the local window self-attention. As mentioned in main contents, our *within-stage-cascading* avoids concatenation of intermediary features to make overall network efficient. Instead, the residual connections between NSTBs are made as in Fig. 8(c), which specifically depicts the small and rounded light green arrows of Fig. 1.

With this mechanism, the extended work is done in PCARN (Ahn et al., 2022). In this version, PCARN explored how to reconstruct more visually natural SR images, by use of substituted objective functions, such as adversarial loss and VGG loss. By doing so, they result in much more photo-realistic and detailed textures. Considering CARN is proposed in 2018 but the cascading mechanism is still valid to the reconstruction task, it can be explored to further increase the effectiveness and efficiency of deep learning models for other low-level vision tasks.



(a) Global Cascading (b) Local Cascading in a Cascading Block (c) *within-stage-cascading* (ours)

Figure 8: The cascading mechanisms in CARN and NGswin. **(a)** Cascading blocks in CARN encoder are globally cascaded into next layers. **(b)** Each cascading block is composed of locally cascaded residual blocks. Each residual block consists of conv-ReLU-conv-skip connection-ReLU. **(c)** *within-stage-cascading* between our proposed NSTBs (sec. 3.3).

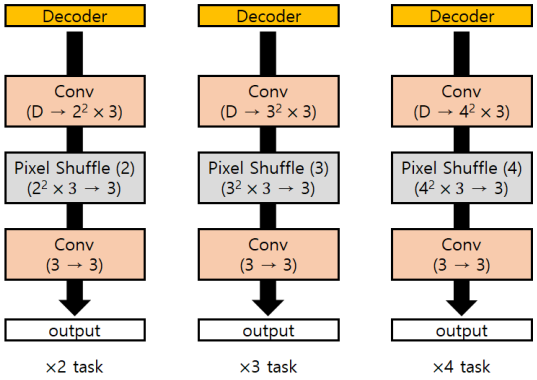


Figure 9: Reconstruction modules for $\times 2$, $\times 3$, and $\times 4$ tasks. What are in parentheses denote change of channel (dimension).

A.3 Reconstruction Module

For final reconstruction module, as following previous methods such as VDSR (Kim et al., 2016), EDSR (Lim et al., 2017), CARN (Ahn et al., 2018), RFDN (Liu et al., 2020), and SwinIR (Liang et al., 2021), we exploit a 3×3 convolutional (conv) layer (channel reducing), pixel-shuffler (sub-pixel operation from Shi et al. (2016)), and a final 3×3 conv layer (equivalent channel). Adding a final conv is difference between ours and previous models. The visualized structures are in Fig. 9. Before reconstruction module, overall architecture including a shallow network, three encoder stages, SCDP bottleneck, and a decoder, equals among all upscaling tasks. In reconstruction phase, the first conv layer of reconstruction module reduces channel from D to $r^2 \times 3$, where r is an upscaling factor (e.g., $\times 2$). After that, pixel-shuffler downsizes channel to 3 (RGB) and upsizes resolution of HR images. Then, the final conv layer processes the output of pixel-shuffler while keeping channel as 3 (RGB channels). At inference and test, with the output of reconstruction module, we clamp the values below 0.0 or over 1.0 into 0.0 or 1.0, respectively.

A.4 Experiment Details and Other Findings About Training Strategies

In this section, we explain experiment settings and our findings from the results of various learning strategies. We hope that future researchers can get insight from our findings. More strategies not mentioned in this section will be investigated in further works.

Model and Hyper-Parameters. The number of both NSTBs in encoder and decoder, $\{K_1, K_2, K_3, K_{dec}\}$, is set to $\{6, 4, 4, 6\}$. The number of WSA heads equals the settings above. By default, we set the network dimension D, hidden dimension of FFN after WSA, window size M, and N-Gram size N to 64, 128, 8, and 2, respectively. The batch size and training epochs are set to 64 and 500, respectively. We train the network from scratch for $\times 2$ task. Whereas, for $\times 3$ and $\times 4$ tasks, warm-start and fine-tuning process (Lin et al., 2022) is employed using pretrained weights on $\times 2$ task. The epochs of the warm-start and fine-tuning are set to 50 and 250, respectively (i.e., total 300 epochs). The process of warm-start is as followed: During the first 50 epochs (warm-start epochs) of training, the network freezes all layers but final reconstruction module and updates reconstruction module only. After warm-start epochs, entire weights are updated by back-propagation. With this strategy, we significantly reduces training times, compared with learning from scratch. For the network optimization, Adam optimizer is used as mentioned in main contents with $\{\beta_1, \beta_2\} = \{0.9, 0.999\}$ and $\epsilon =$ 1e-8. The initial learning rate is set to 0.0004 and decayed by half (half-decay) after $\{200, 300, 400, 425, 450, 475\}$ epochs, with 20 warmup epochs (linearly increasing from 0.0 to 0.0004). At warm-start phase for $\times 3$ and $\times 4$ tasks, learning rate is kept as 0.0004 (initial learning rate). During whole fine-tuning, 10 epochs are used as warmup epochs and the learning rate is reduced in half after $\{50, 100, 150, 175, 200, 225\}$ epochs. We implement the model configuration, training process, and testing process by Pytorch (Paszke et al., 2019) on 4 NVIDIA TITAN Xp GPUs.

Dataset. The whole training dataset is repeatedly used 80 times in each epoch to maximize the merits of random-crop strategy, same as common settings. That is, NGswin is trained for 500K iterations, same as SwinIR. By doing so, the number of total patches (64×64) used in training is 32M (800 images $\times$ 80 repeats $\times$ 500 epochs). In augmentation, random horizontal flip and random 90°, 180°, 270° rotation are applied. Also we normalize the train images with mean and standard deviation (std) of 800 LR images (matching each upscaling task) from DIV2K (Agustsson & Timofte, 2017). When calculating loss function between SR and HR images, the normalized SR images are de-normalized (inverse of normalization). While we train NGswin with different normalization strategies including setting std to 1.0 and not de-normalizing before loss function, those strategies fall behind the aforementioned strategy. The entire images including test data are converted from *".png"* files to *".npy"* (numpy) files, for loading data faster.

Learning Rate. As a result of using various learning rate (*lr*) and batch size (*bs*), we discover the best *lr* for NGswin is when $lr = 0.0004 \times bs/64$. When we can utilize extra computational resources for training, if *bs* goes up, *lr* is correspondingly increased following the formula above. That strategy is equally applied on all of experiments mentioned in this paper. And, it is observed that too high learning rate (matching or over 0.0064) causes the objective function to diverge to the infinity.

Learning Rate Decay. We figure out that cosine learning rate decay (cosine-decay) is not good for SISR. It is because the underfitting (not overfitting) is a crucial issue to SISR (Lin et al., 2022), unlike the high-level vision tasks such as classification, object detection, and semantic segmentation. The cosine-decay tends to reduce the learning rate much faster than the half-decay (keeping a constant learning rate for long phases) and lead to the underfitted model. But as is commonly known, even if half-decay helps the network converge to around the optimal point fast, it is almost impossible to reach the global optimum without the proper decay point. We observed that too early or late decay point, rather, leaded to a wrong converging point.

Regularization. We find that the regularization strategies such as weight decay, gradient clipping, and layer-wise learning rate decay have bad effect on the optimization. At $\times 2$ task (learning from scratch), we apply 0.05 weight decay and 5.0 gradient clipping, but this strategy drops the performance of NGswin. Layer-wise learning rate decay improved the performance of high-level vision tasks when fine-tuning Transformer models (Bao et al., 2021; He et al., 2022). But when we fine-tune NGswin on $\times 3$ and $\times 4$ tasks with 0.9 or 0.75 layer-wise decay hyper-parameter, the model cannot learn representation for reconstructing high-frequency information well. This is also because SISR task suffers from underfitting unlike classification task.

Table 7: Ablation study on NGswin-*me*. Experimentally comparative components are highlighted.

Model	Scale	N-Gram	Swin ver.	training epochs	Mult-Adds	#Params	Set5		Set14		Manga109	
							PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
NGswin-*me* (4×4 window)	×2	*w/o*	V2	400	114.39G	731,944	37.96	0.9605	33.67	0.9189	38.62	0.9768
		w/	V2	400	122.27G	918,640	38.00	0.9606	33.72	0.9187	38.73	0.9772
		w/	V1	200	122.27G	918,432	37.80	0.9599	33.48	0.9171	38.30	0.9763
		w/	V2	200	122.27G	918,640	37.88	0.9602	33.55	0.9175	38.45	0.9767

A.5 More Visual Comparisons

In Fig. 10 and 11, we show additional visual comparisons with other models. As NGswin utilzes longer-span and contextual information, highly degraded pixels by bi-cubic interpolation are restored with more accurate textures.

A.6 Ablation studies on Small Model

We already demonstrate the impact of N-Gram context for NGswin, in Tab. 4. Since it is proven that the window size is a crucial factor to improve our model, we additionally conduct further ablation study on NGswin-*me* that has 4×4 window size, to verify whether our model architecture has power on SISR performances. We evaluate the effects of N-Gram context as shown in Tab. 7 on ×2 task only. N-Gram context consistently shows the meaningful results for reconstruction task with our small model. Moreover, we explore the performance difference between Swin Transformer versions (V1 & V2). When the backbone network is replaced with Swin V1, pre-normalization and standard scaled-dot-product-attention are employed. Because changing the position of layer-norm does not adds any extra multiplications or parameters. In contrast, as scaled-cosine-attention come back to scaled-dot-product-attention, a learnable scalar τ is gone, which very slightly reduces the number of parameters. As a result, Swin V2 (we adopt) is much better than Swin V1 at around middle of training phase, general trend of which tends to be maintained during later epochs (according to our experiences). Note that we report the results of NGswin-*me* at 400-th and 200-th epochs for *w/o vs. w/* N-Gram and Swin V1 *vs.* Swin V2, respectively. It is because this ablation studies is stopped at those epochs, to conduct other ablation studies mentioned in the main contents.

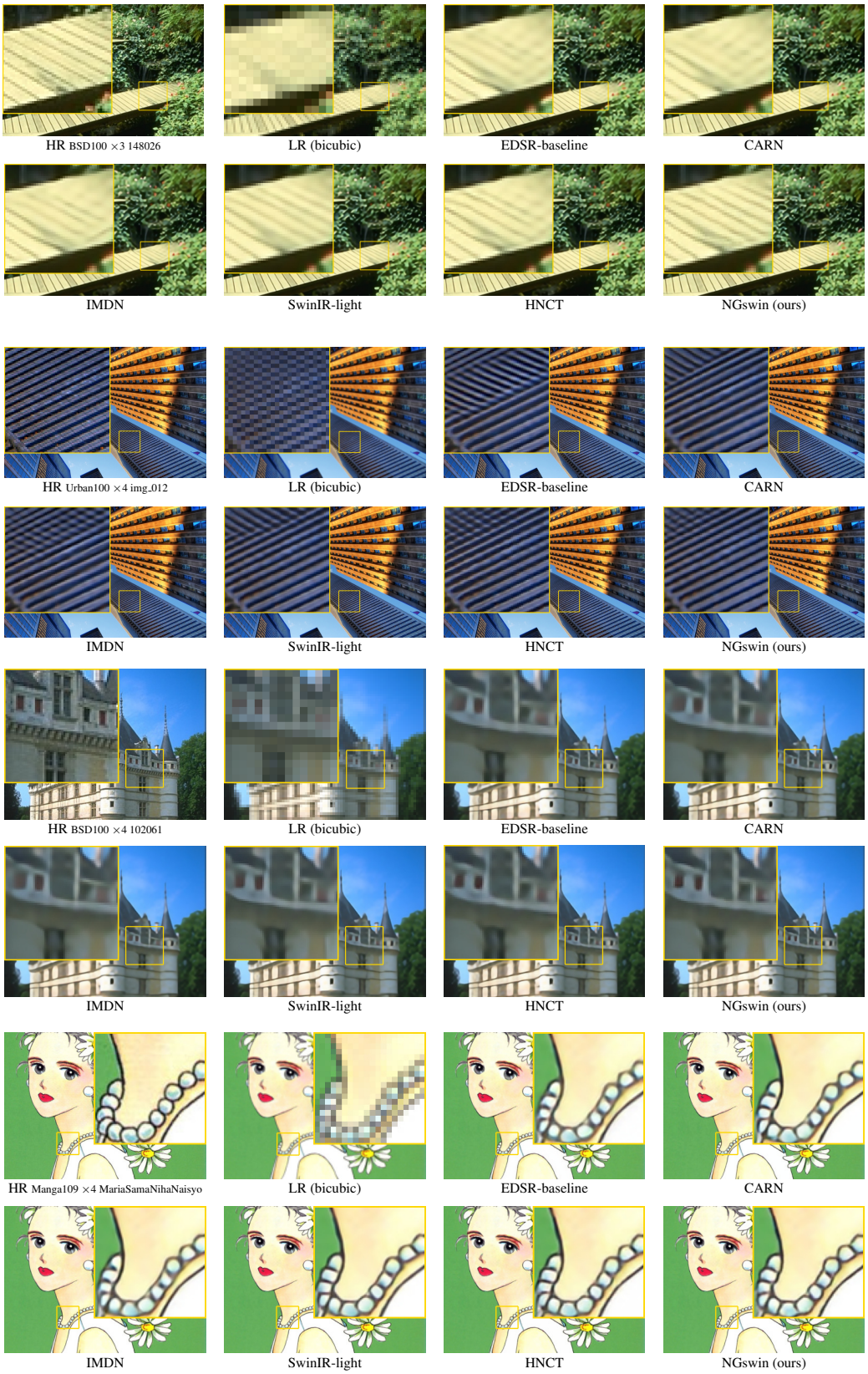


Figure 10: More visual comparisons with other models.

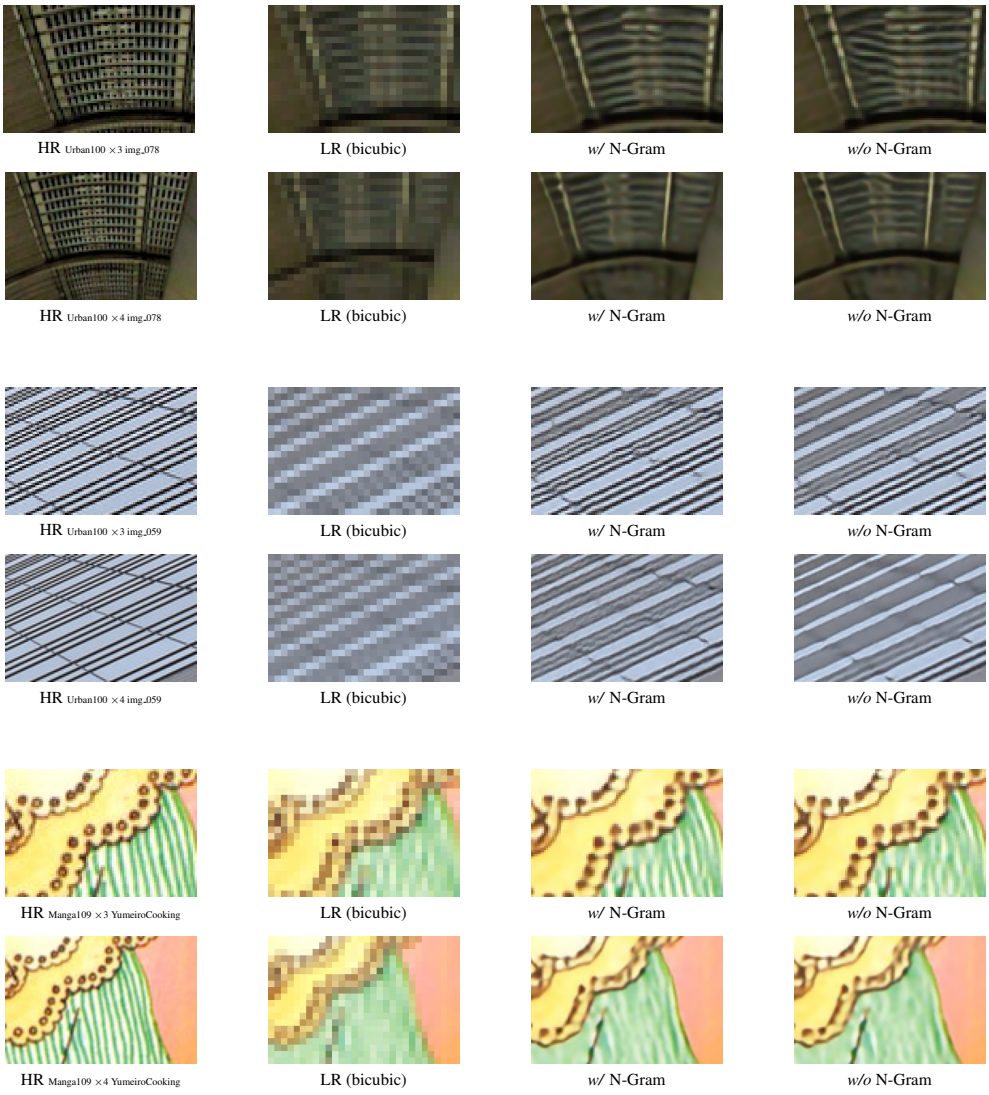


Figure 11: More visual comparisons of *w/* and *w/o* N-Gram context.