EXTRACTER: EFFICIENT TEXTURE MATCHING WITH ATTENTION AND GRADIENT ENHANCING FOR LARGE SCALE IMAGE SUPER RESOLUTION

Esteban Reyes-Saldaña, Mariano Rivera

Centro de Investigacion en Matematicas A.C.
Guanajuato, Gto., 36023 Mexico
{esteban.reyes, mrivera}@cimat.mx

ABSTRACT

Recent Reference-Based image super-resolution (RefSR) has improved SOTA deep methods by introducing attention mechanisms to enhance low-resolution(LR) images by transferring high-resolution textures from a high-resolution reference image. The main idea is to search for matches between patches using LR and Reference image pairs in a feature space and merge them using deep architectures. However, existing methods lack the accurate and efficient search of textures. They divide images into as many patches as possible, resulting in inefficient memory usage, and cannot manage inference using large images. Herein, we propose a deep search that can dynamically change the window and stride size in the inference mode resulting in a more efficient memory usage that reduces significantly the number of image patches and finds the k most relevant texture match for each low-resolution patch over the high-resolution reference patches, resulting in an accurate texture match. Our main contribution is that, using larger kernels, we reduce the multiplication cost between LR and Reference features maintaining the SoTA performance and allowing us to generate large-dimension images (1024 px) in a single GPU. We enhance the Super Resolution result by adding gradient density information using a simple residual architecture showing competitive metrics results: PSNR and SSIM.

Index Terms— Reference based super-resolution, Texture transfer, Transformer, Cross-attention, Gradient density features

1. INTRODUCTION

The paradigm Image Reference-Super Resolution aims to recover high-resolution Images by transferring accurate textures from a reference image (with a given similarity degree) reducing burred and artifacts. In recent years, vision transformers have improved super-resolution results. For example, TTSR[1] introduces attention to Ref-Super Resolution by successfully transferring textures from a reference image. They use a learnable feature extractor to obtain attention matrices Q, K, V and perform a cross-attention mechanism to find the best features for the SR reconstruction.

Lin et al. [2] proposed a novel low-resolution backbone capable of extracting a best feature representation and adding a branch to refine the low-resolution and reference features. Some other works [3, 4] claim that a better texture search is required in order to obtain less blurred images and use multiple reference images for a more accurate pattern search. [5, 6] use a two-stage search to filter textures: first, a block search (non–overlapping windows) is performed and then a dense patch search is done using the filtering matches found in the first stage with memory efficiency by using low-resolution dimensions to find correlations. Finally, some corrections are made using contrast correction or gradient information for the final result.

To address the above problems, we propose a search strategy to efficiently split the images into patches. Different from [1] and most of the recent methods, we split images into patches using a 6×6 window and a stride $s = 2$, instead of 3×3 and $s = 1$ for the deepest feature level, resulting in less patches per image and a more memory efficient usage that can allow us to use large-scale images in the inference phase. Then, we find the top_k HR matches for each LR patch and add structural information for enhancing the Super-Resolution result. Specifically, we first extract deep features from a VGG19-based architecture and unfold the texture features into patches using a 6×6 kernel and a stride of 2. Second, we propose an efficient search strategy but different from [5, 1], we use top_k matches between the low-resolution and ref patches instead of the max feature for each low-resolution patch. Finally, we merge textures at different scales and add gradient density information to form a better spatial reconstruction using a simple residual network.

The primary contributions of this paper are. First. we introduce a more efficient Search and Transfer module to identify correlations between low-resolution and reference patches; we use larger windows compared with state-of-the-art (SOTA) methods. This significantly reduces the correla-

Code available at `https://github.com/esteban-rs/EXTRACTER`.

tion matrix dimension and allows to use the top-k matches to enhance texture transfer. Our main contribution is that, using larger kernels with larger strides, we reduce the multiplication cost between low-resolution and reference features maintaining the SoTA performance and allowing us to generate large-dimension images (1024 px) in a single GPU. Second, we introduce a Gradient Density-Enhancing Module (GDE) to improve the merging of textures from different deep levels while considering gradient density information. This module is implemented by a straightforward recurrent network. And third, we conduct extensive experiments on benchmark datasets that provide us strong evidence that the proposal overcomes SOTA methods.

2. RELATED WORK

In recent years, Single Image Super Resolution (SISR) improved super-resolution methods by using residual blocks[7] and designing deeper networks. These methods use $\mathcal{L}_1$ and $\mathcal{L}_2$ losses as the training objective functions that have demonstrated non accuracy for human perception [1]. To solve this, novel methods use a GAN strategy[8] resulting in better satisfying results or adopt classic computer vision transformation such as gradient mapping [9].

Since the appearance of vision transformers, vision tasks has been improved. For example, TTSR[1] introduces cross-attention to Ref-Super Resolution for transferring textures: a patch matching based technique robust to miss-alignment problems [10, 11]. Based on TTSR, Lin et al. [2] add channel-wise attention. [3, 4] and use multiple image patches for transferring textures, resulting in better results. In this direction, cross-attention mechanisms are used and better memory usage is required. Gou et al. [6, 5] enhance memory efficiency by using low-resolution dimensions to find correlations and use classical vision transformation for structural reconstruction, such as contrast normalization or gradient density flow.

3. METHOD

In this section, we proposed Efficient Texture Matching with Attention and Gradient Enhancing for Image Super Resolution (Extracter). It consists of four modules: Deep Feature Extractor (DFE), Search and Transfer Module (STM), Cross-Scale Feature Integration (CSFI), and Gradient Density Enhancing Module (GDE). The main scheme is shown in Fig. 1.

The model produces a $4\times$ super-resolution image. It inputs (Lr_u, Ref_{du}, Ref). Lr_u represents the bicubic upsampled low-resolution image and Ref_{du} represents a bicubic downsampling-upsampling concatenation operation over Ref image. Using the FDE, we produce Q, K, V feature maps and find the correlation matrix (R) using Q, K normalized inner product between patches. Then we filtered the best patches based on correlation matrix R and then we take

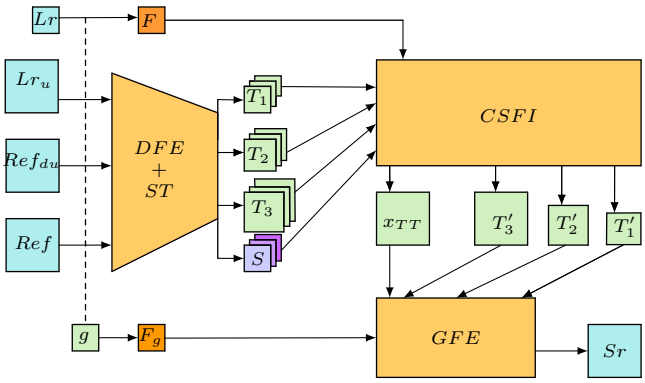

Fig. 1. Efficient Texture Matching with Attention Scheme: We input the low-resolution image, reference down-upsampled, and the reference image, pass it through the Deep Feature Extractor (DFE) to obtain the k relevance textures and score matrices at multiple levels with the SearchAndTransfer (ST). Then we merge the simple features F with the attention textures using a Cross Scale Feature Integration (CSFI). Finally, we refine the partial super-resolution result x_{TT} adding gradient features F_g extracted from the Gradient Density map g to obtain the final Super-Resolution image.

the top_k matches for each patch. We integrate the obtained features at three different scales using a Cross Scale Feature Integration[1] and finally, we add gradient density from the LR image to improve structural information and create the Super-Resolution image.

3.1. Deep Feature Extractor

We transform the data into a new representation with more evident complex characteristics at different resolutions. For this, we use the VGG19 [12] backbone (previously trained with ImageNet[13]). Let (Lr_u, Ref_{du}, Ref) be the input to our Deep Feature Extractor(DFE). The output can be formulated as

$$Q_i = DFE_i(Lr_u) \quad (1)$$
$$K_i = DFE_i(Ref_{du}) \quad (2)$$
$$V_i = DFE_i(Ref) \quad (3)$$

where i denotes the depth of DFE. Similar to [1], we take three depths of features from VGG19 that reduces image dimensions $2\times$ at each level with output channels $[64, 128, 256]$ giving us more complex information.

3.2. Search and Transfer Module

Let us omit the i index from (1) for notation simplification. The following calculations are made for a fixed level in DFE and are depicted at Fig. 2. We infer correlations between LR_u and Ref_{du} using attention via Q and K. First we divide Q, K into overlapping patches $q_i : i \in [1, 2, \ldots, H_{LR} \times W_{LR}/s^2]$ and $k_j : j \in [1, 2, \ldots, H_{Ref} \times W_{Ref}/s^2]$, respectively, where s is the stride setup for patch displacement. In experiments, we use a window of 6 and stride $s = 2$ for memory efficiency but we can dynamically change it into one of the following

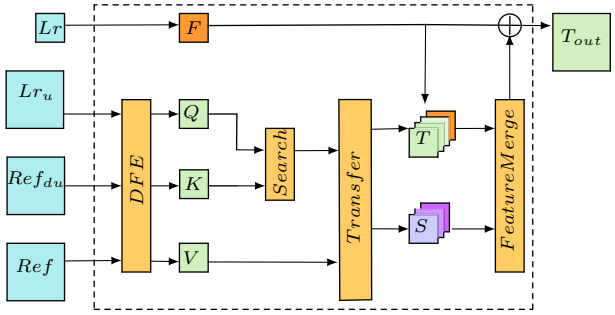

Fig. 2. Feature extraction and texture search: The model inputs Lr_u, Ref_{du}, Ref, pass it through a Deep Feature Extractor (DFE) to perform patch Correlation Search. We use the result as index to select the best k-textures by Transfer. Finally, with textures T and soft-attention matrices S, we merge them with simple features F from Lr to create T_{out}.

configurations $\{[3,1],[6,2],[12,4]\}$ for kernel and stride size, respectively. The correlation matrix is computed as the normalized inner product

$$c_{i,j} = \left\langle \frac{q_i}{||q_i||}, \frac{k_j}{||k_j||} \right\rangle. \tag{4}$$

Next, for each q_i, we keep the u-best score-indices of the key patches $\{k_j\}_{j=1}^{H_{Ref} \times W_{Ref}/s^2}$. Different from other recent methods [1, 5, 6], that uses only the 1-best match, we introduce more feature information by obtaining several index and score mappings $H = \{H_0, H_1, \ldots, H_{u-1}\}$ and $S = \{S_0, S_1, \ldots, S_{u-1}\}$ where

$$H, S = top_u(C) \tag{5}$$

with S, H tensors containing the u-maximum scores and index from C; i.e.,

$$H_0 = \arg\max_j C_{ij}\ ,\ \ S_0 = \max_j C_{ij} \tag{6}$$

and H_1, S_1 be the second maximum indices and scores matches, etc. Using this index mappings, we select the best textures from V as

$$T_i = V_{H_i}$$

So that, we extract the best matches using the hard attention matrices as index. Finally, given a LR image x, the output of the Initial Feature Extractor (IFE) is denoted as

$$F = IFE(x).$$

Hence, we integrate the texture features T_i:

$$F_{TT} = F + \sum_{i=1}^{u} Conv_i(Concat(F, T_i \otimes S_i)) \otimes S_i; \tag{7}$$

where $\otimes$, $Conv_i(\cdot)$ and $Contat(\cdot)$ denotes element-wise multiplication, convolutional 3×3 and concatenation blocks, respectively.

3.3. Cross-Scale Feature Integration

Inspired by SoTA methods for style/texture transferring [14, 15, 1], We integrate the previous attention results at different scales following [1]; this can be modeled as

$$x_{TT}, T_1, T_2, T_3 = CCFI(\{F_{TT}^{(i)}\}_{1=1,2,\ldots});$$

where x_{TT} is the merged super-resolution texture and T_1, T_2, T_3 are the syntetized textures.

3.4. Gradient Enhancing Density Module

To give more information about the structure of the low-resolution image, some work has been done [9, 7]. We incorporate a Gradient Enhancing module for adding structural and edge information to the partial output of the $CSFI(\cdot)$. First, we extract the Gradient Density for each of the RGB image channels we convolve the Image with 3×3 Sobel filters kernels [16] from x and y derivative directions; K_x and K_y, respectively. and calculate Gradient Density as

$$GD(I) = \sqrt{(K_x * I)^2 + (K_y * I)^2}.$$

Now, we pass the image gradient density g through a residual feature extractor: $F_g = GFE(g)$. Finally, using the output from $CSFI(\cdot) : x_{TT}, T_1, T_2, T_3$, the SR image is formulate as

$$\begin{aligned} x_{1g} &= RB_1(Conv(Concat(F_g, T_3))) \\ x_{2g} &= RB_2(Conv(Concat(x_{1g} \uparrow, T_2))) \\ x_{3g} &= RB_3(Conv(Concat(x_{3g} \uparrow, T_3))) \\ SR &= Conv(Concat(x_{3g}, x_{TT})) \end{aligned}$$

where $RB(\cdot)$ represents a residual scheme and $\uparrow$ is $2\times$ bicubic upsampling.

3.5. Loss Function

The overall loss is

$$\mathcal{L}_{total} = \lambda_1 \mathcal{L}_{rec} + \lambda_2 \mathcal{L}_{perc} + \lambda_3 \mathcal{L}_{grad} + \lambda_4 \mathcal{L}_{adv} \tag{8}$$

where

$$\mathcal{L}_{rec} = (chw)^{-1} ||SR - HR||_1,$$

with c, h, w the channel, height, weight of the HR image. In the aim of enhacing the similarity of the feature space representation of the generated image and the SR image using the $vgg19$ feature space [17, 18], we use

$$\mathcal{L}_{perc} = (c_i h_i w_i)^{-1} ||vgg19_i(SR) - vgg19_i(HR)||_1,$$

with c_i, h_i, w_i the channel, height, weight at the correspoinding i level. For structural similarity enhacing, we introduce Gradient Density Loss using (3.4)

$$\mathcal{L}_{grad} = (chw)^{-1} ||GD(SR) - GD(HR)||_1,$$

with c_i, h_i, w_i the channel, height, weight at the correspoinding i level. Similar to [1, 19], we use a WGAN-GP for more stable training. This loss is described as

$$\begin{aligned}\mathcal{L}_D &= \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}\left[D(\tilde{x})\right] - \mathbb{E}_{x\sim\mathbb{P}_r}\left[D(x)\right] \\ &\quad +\lambda\mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}\left[\|\nabla_{\hat{x}}D(\hat{x})\|_2 - 1)^2\right], \\ \mathcal{L}_G &= -\mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}\left[D(\tilde{x})\right].\end{aligned}$$

3.6. Implementation Details

The window size for extracting patches is set as $k = 6$ with padding $p = 2$ and a stride of $s = 2$. In experiments, we explore other configurations. For both LR IFE and Gradient IFE we use 2 residual blocks. The architecture for the CSFI model is $[8, 8, 4]$ and $[9, 9, 9]$ for GDE for each level. To find the correlation matrix, we use only the deepest feature extractor level to perform matrix multiplication and upsample it to higher levels. We use data augmentation for training by randomly flipping up-down and left-right followed by a random rotation of $90°, 180°, 270°$ with a batch fixed to 9 and $p = 0.5$ from a uniform distribution for swapping the LR and Ref image. The weights of the loss coefficients are $1, 1e^{-2}, 1e^{-3}, 1e^{-3}$ in the same order of equation (8). An Adam optimizer with $lr = 1e^{-4}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and default $\epsilon = 1e^{-8}$. All the experiments were performed in a single GPU NVIDIA GeForce RTX 3090 using the pytorch framework. The capability of the purpose model to be tested in a single GPU (24Gb) is an important advantage over the compared methods in Table 1 since such methods can not perform inference in a single GPU for the used image dimensions.

4. EXPERIMENTS AND RESULTS

Following the recent work, we use two metrics to evaluate our results: Peak Signal to Noise Ratio (PSNR) and Structure Similarity Index (SSIM) [20]. We conduct the training using CUFED5 Dataset [21]. It contains 11,871 pairs consisting of an input and reference image. There are 126 testing images, each having 4 reference images with different similarity levels. We also evaluate our method using different test sets such as Sun80 [22], Urban100 [23], and Set14[24]. Sun80 contains 80 natural images, each of them paired with several reference images. Urban100 and Set14 do not have reference images so we took it randomly from the same dataset. All the SR results are evaluated of PSRN and SSIM on the Y channel of YCbCr space. Following the SOTA methods, we train our model using the train set from CUFED5 and test it on the CUFED5 test set, Sun80, Urban100, and Set14. Two versions of our model were trained, the first one trained only using reconstruction loss (**Extracter-rec**) and the second one, using all losses. EXTRACTER-rec outperforms recent methods despite using a bigger window size with bigger stride size, resulting in less patches, reducing matrix multiplication, as we can see in Table 1. We observe better visual results when all losses were used, Fig. 1 illustrates some visual results

Method	CUFED5	Sun80	Urban100	Set14
SRNTT	25.61 / .764	27.59 / .756	25.09 / .774	26.73 / .731
SRNTT-rec	26.24 / .784	28.54 / .793	25.50 / .784	27.68 / .766
TTSR	25.63 / .765	28.59 / .774	24.69 / .748	26.88 / .748
TTSR-rec	27.03 / .802	30.02 / .814	25.88 / .784	28.10 / .782
SSEN-rec	26.78 / .791	-	-	-
DPFSR	25.23 / .749	28.59 / .774	24.35 / .734	-
DPFSR-rec	27.25 / .808	30.10 / .815	**26.03 / .787**	-
C^2- Matching	27.16 / .805	29.75 / .799	25.52 / .764	-
MASA	24.92 / 0.729	27.12 / 0.708	23.78 / 0.712	-
Extracter	26.40 / .789	29.02 / .789	24.72 / .752	26.50/.740
Extracter-rec	**27.43 / .813**	**30.14 / .816**	**26.04 / .785**	**28.15 / .782**

Table 1. Quantitative metric results of the generated images using PSNR / SSIM. The highest score is denoted in black. Our model shown better metric results that other SOTA methods while improving efficiency in large-scale datasets inference.

with other novel models. We study different configurations for our model. Table 2 shows the number of parameters and the correlation matrix shape during the training phase for the CUFED5 dataset. We found that our method reduces $4\times$ the shape from the attention mechanism. Table 3 shows the effectiveness of changing the kernel size for the test phase using large image size datasets such as Sun80 and Urban100. Finally, Table 4 show the importance of different modules in our model. Better results are obtain when adding GDE module and more texture in transfering.

Method	Params. (M)	Kernel size	corr. matrix shape
TTSR	6.99	3×3	1600×1600
DPFSR	6.91	3×3	1600×1600
Extracter	8.31	6×6	800×800

Table 2. Model parameters and shape of the training correlation matrix. Our method reduce significantly the matrix multiplication cost by extracting larger patches.

Kernel Size	Sun80	Urban100
3×3	OFM	OFM
6×6	30.14 / .815	26.04 / .785
12×12	29.98 / .814	25.74 / .781

Table 3. Kernel size when obtaining patches for PSNR / SSIM metrics with our model. All comparisons where made on single GPU. Models using 3×3 kernel like TTSR and DPFSR produces Out of Memory (OFM) due the large image dimensions on Sun80 and Uban100 datasets.

Config	Metrics	Params(M)
Extracter-rec (no GDE)	27.14/.8085	6.02
Extracter-rec (top-1)	27.19/.8110	7.58
Extracter-rec (top-2)	27.27/..8125	7.94

Table 4. Effectiveness of or model configuration with PSNR / SSIM metrics. All comparisons were made on a single GPU using the CUFED5 Test-Set. Results show the importance of adding the Gradient Density Enhanced module and also using more textures in equation 5

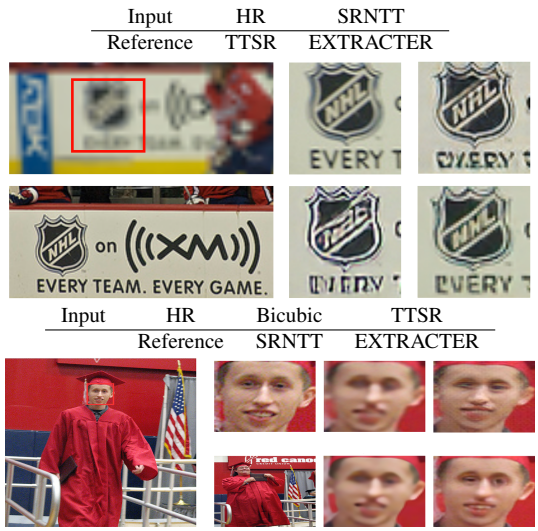

Fig. 3. Experimental results: we compare our model with available testing models online using CUFED5 Test Set. We display two cases: In the fist row, we are able to recover shapes and letters successfully. In the second row: we recover global results but faces are very familiar to the human eye and we can discriminate better. In Section 4.1 we use a specific dataset of human faces to train, and we can appreciate the differences in visual results.

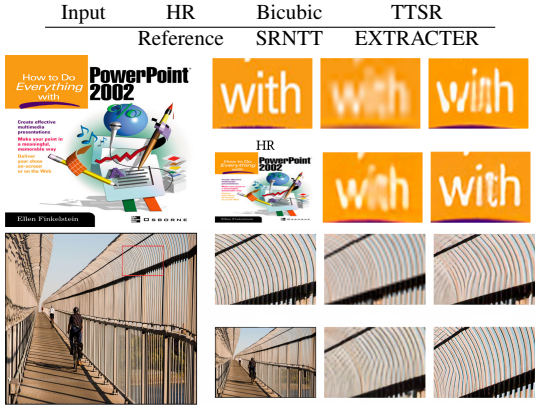

Fig. 4. Experimental results on different TestSets. First row: ppt3.png in Sun80. Second row: Urban100 image_24.png of in Urban100 Dataset. We use the LR Upsampled as the reference for self-similarity.

4.1. Applicatons

To prove the importance of the transferring mechanism in other tasks, we use our model in two different scenarios: generative model result improvement and dehazing. For the SR task, we produce a $4\times$ image but for texture enhazing, we use the same size for all input images). In [19], the transferring is used to enhance the VAE result. Let (x, z, x') be the input face image, the latent representation and the reconstructed face and (y, z', y') a reference face. Using a Extracter-Face, we input x' as the blurred low-resolution face and (y, y') as the (reference, reference$_{du}$). The model produces a tuned texture face approximation of x. Using CelebA-HQ dataset at 256 image size, we train Extracter-Face and performe evalueation on $30k$ samples as other SoTA methods. Table 5 show FID metric results and Figure show visual results sampling.

Model	FID ↓
VAE 1024	87.68
VQ-VAE	40.38
VAE + Extracter	34.75
VQ-VAE + Extracter	25.71

Table 5. Quality metric of the generated images. Smaller is better for FID. The configuration for the network architectures are the same as [19].

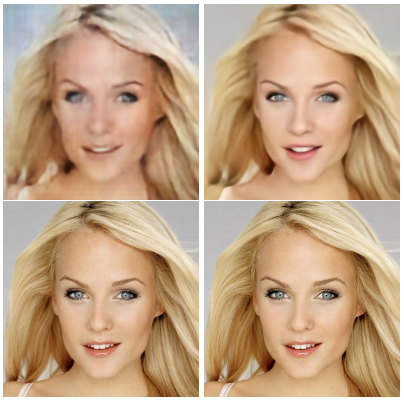

Fig. 5. Results comparation. VAE result in up-left, VQ-VAE result in up-right, VQVAE + Extracter in down-left and Ground True in down-right.

The second application is on the Dehazing task. We use RESIDE-INDOOR Dataset [25], when the input is a haze image and we use randomly a clean reference image from the same dataset. We use the TestSet to compare with more complex SoTA models. Table 6 show the PSNR and SSIM metric results. Figure 6 show some visual dehazing examples.

Model	PSNR	SSIM
Extracter	33.74	.984
U2-Former [26]	36.42	0.988
DehazeFormer-L [27]	40.04	0.996

Table 6. Quality metric of the generated images. Smaller is better for FID. The configuration for the network architectures are the same as [19].

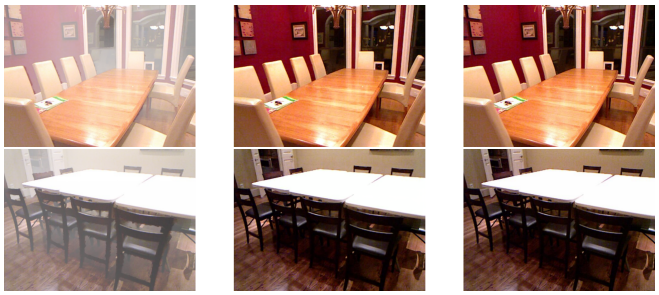

Fig. 6. Visual results comparison. The first column correspond to the haze image, the second is the Extracter output and finally the Ground Truth.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we propose a novel deep texture search with more efficient memory usage for RefSR. The proposed model consists of a learnable Deep Feature Extractor, a Search and Transfer Module that uses the top-k matches between the Lr and Ref patches for transferring textures in a more efficient memory usage way than SOTA methods by using larger windows with larger strides, a Cross Scale Feature Integrator and, finally, a Gradient Enhancing Density module. Our experiments demonstrate the competitive performance of EXTRACTER over the recent attention mechanisms for RefSR using PSNR and SSIM metrics. The ablation studies demonstrate the efficiency of managing larger windows when using large-scale images, resulting in a non-out-of-memory as other recent methods. In the future, we would like to enhance our model by changing the CSFI for a simpler network to reduce training time, using the transferring mechanisms to refine generative models, and exploring RefSR real-world applications, such as satellite super-resolution and movie super-resolution.

Acknowledges. Work supported by Conachyt, Mexico (Grant CB-A1-43858) and E. Reyes Scholarship.

6. REFERENCES

[1] F. Yang, H. Yang, J. Fu, H. Lu, and B. Guo, "Learning texture transformer network for image super-resolution," in *Proc. CVPR*, 2020, pp. 5791–5800.

[2] Ruirong Lin and Nanfeng Xiao, "Dual projection fusion for reference-based image super-resolution," *Sensors*, vol. 22, no. 11, pp. 4119, 2022.

[3] Xu Yan, Weibing Zhao, Kun Yuan, Ruimao Zhang, Zhen Li, and Shuguang Cui, "Towards content-independent multi-reference super-resolution: Adaptive pattern matching and feature aggregation," in *Computer Vision – ECCV*. 2020, pp. 52–68, Springer Int. Pub.

[4] Yanchun Xie, Jimin Xiao, Mingjie Sun, Chao Yao, and Kaizhu Huang, "Feature representation matters: End-to-end learning for reference-based image super-resolution," in *Computer Vision – ECCV*, Cham, 2020, pp. 230–245, Springer Int. Pub.

[5] Liying Lu, Wenbo Li, Xin Tao, Jiangbo Lu, and Jiaya Jia, "Masa-sr: Matching acceleration and spatial adaptation for reference-based image super-resolution," 2021.

[6] Kehua Guo, Liang Chen, Xiangyuan Zhu, Xiaoyan Kui, Jian Zhang, and Heyuan Shi, "Double-layer search and adaptive pooling fusion for reference-based image super-resolution," *ACM Trans. Multimedia Comput. Commun. Appl.*, 2023.

[7] Ziyu Liu, Ruyi Feng, Lizhe Wang, and Tieyong Zeng, "Gradient prior dilated convolution network for remote sensing image super-resolution," *IEEE Jou. Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 16, pp. 3945–3958, 2023.

[8] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. CVPR*, 2017, pp. 4681–4690.

[9] Jian Sun, Zongben Xu, and Heung-Yeung Shum, "Gradient profile prior and its applications in image super-resolution and enhancement," *IEEE Trans. on image process.*, vol. 20, pp. 1529–42, 11 2010.

[10] Huanjing Yue, Xiaoyan Sun, Jingyu Yang, and Feng Wu, "Landmark image super-resolution by retrieving web images," *IEEE Trans. on Image Process.*, vol. 22, no. 12, pp. 4865–4878, 2013.

[11] Haitian Zheng, Mengqi Ji, Haoqian Wang, Yebin Liu, and Lu Fang, "Crossnet: An end-to-end reference-based super resolution network using cross-scale warping," in *Proc. CVPR*, 2018, pp. 88–104.

[12] K Simonyan and A Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations (ICLR 2015)*. Computational and Biological Learning Society, 2015.

[13] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE/CVF CVPR*. Ieee, 2009, pp. 248–255.

[14] L. Gatys, A. Ecker, and M. Bethge, "A neural algorithm of artistic style," *Jou. of Vision*, vol. 16, no. 12, pp. 326–326, 2016.

[15] Yanhong Zeng, Jianlong Fu, Hongyang Chao, and Baining Guo, "Learning pyramid-context encoder network for high-quality image inpainting," in *Proc. CVPR*, 2019, pp. 1486–1494.

[16] N. Kanopoulos, N. Vasanthavada, and R.L. Baker, "Design of an image edge detection filter using the sobel operator," *IEEE Jou. of Solid-State Circuits*, vol. 23, no. 2, pp. 358–367, 1988.

[17] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee, "Enhanced deep residual networks for single image super-resolution," in *Proc. CVPR Workshops*, 2017, pp. 136–144.

[18] Jianbo Wang, Huan Yang, Jianlong Fu, Toshihiko Yamasaki, and Baining Guo, "Fine-grained image style transfer with visual transformers," in *Proc. ACCV*, 2022, pp. 841–857.

[19] Reyes-Saldana Esteban and Rivera Mariano, "Deep variational method with attention fo high-definition face generation," in *Pattern Recognition: 14th Mexican Conference, MCPR 2022, Ciudad Juárez, Mexico, June 22–25, 2022, Proceedings*, Berlin, Heidelberg, 2022, p. 116–126, Springer-Verlag.

[20] Alain Horé and Djemel Ziou, "Image quality metrics: Psnr vs. ssim," in *ICPR*, 2010, pp. 2366–2369.

[21] Zhifei Zhang, Zhaowen Wang, Zhe Lin, and Hairong Qi, "Image super-resolution by neural texture transfer," in *Proc. CVPR*, 2019, pp. 7982–7991.

[22] Libin Sun and James Hays, "Super-resolution from internet-scale scene matching," in *2012 IEEE Int. Conf. Computational Photography*, 2012, pp. 1–12.

[23] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja, "Single image super-resolution from transformed self-exemplars," in *Proc. CVPR*, 2015, pp. 5197–5206.

[24] Roman Zeyde, Michael Elad, and Matan Protter, "On single image scale-up using sparse-representations," in *Curves and Surfaces*, Berlin, Heidelberg, 2012, pp. 711–730, Springer Berlin Heidelberg.

[25] Boyi Li, Wenqi Ren, Dengpan Fu, Dacheng Tao, Dan Feng, Wenjun Zeng, and Zhangyang Wang, "Reside: A benchmark for single image dehazing," *arXiv preprint arXiv:1712.04143*, 2017.

[26] Haobo Ji, Xin Feng, Wenjie Pei, Jinxing Li, and Guangming Lu, "U2-former: A nested u-shaped transformer for image restoration," 2021.

[27] Yuda Song, Zhuqing He, Hui Qian, and Xin Du, "Vision transformers for single image dehazing," *IEEE Transactions on Image Processing*, vol. 32, pp. 1927–1941, 2023.