Abstract

We study on image super-resolution (SR), which aims to recover realistic textures from a low-resolution (LR) image. Recent progress has been made by taking high-resolution images as references (Ref), so that relevant textures can be transferred to LR images. However, existing SR approaches neglect to use attention mechanisms to transfer high-resolution (HR) textures from Ref images, which limits these approaches in challenging cases. In this paper, we propose a novel Texture Transformer Network for Image Super-Resolution (TTSR), in which the LR and Ref images are formulated as queries and keys in a transformer, respectively. TTSR consists of four closely-related modules optimized for image generation tasks, including a learnable texture extractor by DNN, a relevance embedding module, a hard-attention module for texture transfer, and a soft-attention module for texture synthesis. Such a design encourages joint feature learning across LR and Ref images, in which deep feature correspondences can be discovered by attention, and thus accurate texture features can be transferred. The proposed texture transformer can be further stacked in a cross-scale way, which enables texture recovery from different levels (e.g., from 1× to 4× magnification). Extensive experiments show that TTSR achieves significant improvements over state-of-the-art approaches on both quantitative and qualitative evaluations.

1. Introduction

Image super-resolution aims to recover natural and realistic textures for a high-resolution image from its degraded low-resolution counterpart [12]. The recent success of image SR can greatly enhance the quality of media content for a better user experiences. For example, the digital zoom algorithm for mobile cameras and image enhancement technology for digital televisions. Besides, this fundamental technology can benefit a broad range of computer vision tasks, like medical imaging [21] and satellite imaging [35].

The research on image SR is usually conducted on two paradigms, including single image super-resolution (SISR), and reference-based image super-resolution (RefSR). Traditional SISR often results in blurry effects, because the high-resolution (HR) textures have been excessively destructed in the degrading process which are unrecoverable. Although generative adversarial networks (GANs) [7] based image SR approaches are proposed to relieve the above problems, the resultant hallucinations and artifacts caused by GANs further pose grand challenges to image SR tasks.

*This work was performed when the first author was visiting Microsoft Research as a research intern.
Recent progress has been made by reference-based image super-resolution (RefSR), which transfers HR textures from a given Ref image to produce visually pleasing results [5, 6, 26, 29, 36]. However, state-of-the-art (SOTA) approaches usually adopt a straightforward way to transfer textures which may result in unsatisfied SR images (as shown in Figure 1). For example, Zheng et al. [43] adopts a flow-based approach which usually searches and transfers inaccurate textures (indicate by red) when facing large viewpoint changes between the LR and Ref image. Zhang et al. [41] adopts a feature space defined by a pre-trained classification model to search and transfer textures between the LR and Ref image. Nevertheless, such high-level semantic features can not effectively represent HR textures which remain to generate implausible results.

To address these problems, we propose a novel Texture Transformer Network for Image Super-Resolution (TTSR). Specifically, four closely-related modules optimized for image generation tasks are proposed. First, we propose a learnable texture extractor, in which parameters will be updated during end-to-end training. Such a design enables a joint feature embedding of LR and Ref images which creates a solid foundation for applying attention mechanism [19, 34, 31] in SR tasks. Second, we propose a relevance embedding module to compute the relevance between the LR and Ref image. More specifically, we formulate the extracted features from the LR and Ref image as the query and key in a transformer [31] to obtain a hard-attention map and a soft-attention map. Finally, we propose a hard-attention module and a soft-attention module to transfer and fuse HR features from the Ref image into LR features extracted from Ref image through the attention maps. The design of TTTSR encourages a more accurate way to search and transfer relevant textures from Ref to LR images.

Furthermore, we propose a cross-scale feature integration module to stack the texture transformer, in which the features are learnt across different scales (e.g. from 1× to 4×) to achieve a more powerful feature representation. As shown in Figure 1, the overall design enables our TTTSR to search and transfer relevant textures from the Ref image (indicated by green) which achieves a better visual result compared with SOTA approaches. The main contributions of this paper are:

- To the best of our knowledge, we are one of the first to introduce the transformer architecture into image generation tasks. More specifically, we propose a texture transformer with four closely-related modules for image SR which achieves significant improvements over SOTA approaches.
- We propose a novel cross-scale feature integration module for image generation tasks which enables our approach to learn a more powerful feature representation by stacking multiple texture transformers.

2. Related Work

In this section, we review previous works of single image super-resolution (SISR) and reference-based image super-resolution (RefSR) which are the most relevant to our work.

2.1. Single Image Super-Resolution

In recent years, deep learning based SISR methods have achieved significant improvements over traditional non-learning based methods. Deep learning based methods in SISR treat this problem as a dense image regression task which learns an end-to-end image mapping function represented by a CNN between LR and HR images.


The above methods use mean square error (MSE) or mean absolute error (MAE) as their objective function which ignores human perceptions. In recent years, more and more works aim to improve perceptual quality. Johnson et al. [13] introduced perceptual loss into SR tasks, while SRGAN [16] adopted generative adversarial networks (GANs) [7] and showed visually satisfying results. Sajjadi et al. [22] used Gram matrix based texture matching loss to enforce local similar textures, while ESRGAN [32] enhanced SRGAN by introducing RRDB with relativistic adversarial loss. Recent proposed RSRGAN [38] trained a ranker and used rank-content loss to optimize the perceptual quality, which achieved state-of-the-art visual results.

2.2. Reference-based Image Super-Resolution

Different from SISR, RefSR can harvest more accurate details from the Ref image. This could be done by several approaches like image aligning or patch matching. Some existing RefSR approaches [33, 36, 43] choose to align the LR and Ref image. Landmark [36] aligned the Ref image to the LR image through a global registration to solve an
energy minimization problem. Wang et al. [33] enhanced the Ref image by recurrently applying non-uniform warping before feature synthesis. CrossNet [43] adopted optical flow to align the LR and Ref image at different scales and concatenated them into the corresponding layers of the decoder. However, the performance of these methods depends largely on the aligning quality between the LR and Ref image. Besides, the aligning approaches such as optical flow are time-consuming, which is adverse to real applications.

Other RefSR approaches [1, 41, 42] adopt “patch match” method to search proper reference information. Boominathan et al. [1] matched the patches between gradient features of the LR and down-sampled Ref image. Zheng et al. [42] replaced the simple gradient features with features in convolution neural networks to apply semantic matching and used a SISR method for feature synthesis. Recent work SRNTT [41] applied patch matching between VGG [24] features of the LR and Ref image to swap similar texture features. However, SRNTT ignores the relevance between original and swapped features and feeds all the swapped features equally into the main network.

To address these problems, we propose a texture transformer network which enables our approach to search and transfer relevant textures from Ref to LR images. Moreover, the performance of our approach can be further improved by stacking multiple texture transformers with a proposed cross-scale feature integration module.

3. Approach

In this section, we introduce the proposed Texture Transformer Network for Image Super-Resolution (TTSR). On top of the texture transformer, we propose a cross-scale feature integration module (CSFI) to further enhance model performances. The texture transformer and CSFI will be discussed in Section 3.1 and Section 3.2, respectively. A group of loss functions for optimizing the proposed network will be explained in Section 3.3.

3.1. Texture Transformer

The structure of the texture transformer is shown in Figure 2. LR, LR↑ and Ref represent the input image, the 4× bicubic-upsampled input image and the reference image, respectively. We sequentially apply bicubic down-sampling and up-sampling with the same factor 4× on Ref to obtain Ref↓↑ which is domain-consistent with LR↑. The texture transformer takes Ref, Ref↓↑, LR↑ and the LR features produced by the backbone as input, and outputs a synthesized feature map, which will be further used to generate the HR prediction. There are four parts in the texture transformer: the learnable texture extractor (LTE), the relevance embedding module (RE), the hard-attention module for feature transfer (HA) and the soft-attention module for feature synthesis (SA). Details will be discussed below.

Learnable Texture Extractor. In RefSR tasks, texture extraction for reference images is essential because accurate and proper texture information will assist the generation of SR images. Instead of using semantic features extracted by a pre-trained classification model like VGG [24], we design a learnable texture extractor whose parameters will be updated during end-to-end training. Such a design encourages a joint feature learning across the LR and Ref image, in which more accurate texture features can be captured. The process of texture extraction can be expressed as:

\[ Q = LTE(LR↑), \]
\[ K = LTE(Ref↓↑), \]
\[ V = LTE(Ref), \]

where \( LTE(\cdot) \) denotes the output of our learnable texture extractor. The extracted texture features, \( Q \) (query), \( K \) (key), and \( V \) (value) indicate three basic elements of the attention mechanism inside a transformer and will be further used in our relevance embedding module.

Relevance Embedding. Relevance embedding aims to embed the relevance between the LR and Ref image by estimating the similarity between \( Q \) and \( K \). We unfold both \( Q \) and \( K \) into patches, denoted as \( q_i \) (\( i \in [1, H_{LR} \times W_{LR}] \)) and \( k_j \) (\( j \in [1, H_{Ref} \times W_{Ref}] \)). Then for each patch \( q_i \) in \( Q \)
and \( k_j \) in \( K \), we calculate the relevance \( r_{i,j} \) between these two patches by normalized inner product:

\[
 r_{i,j} = \frac{q_i'' \cdot k_j}{\|q_i''\| \cdot \|k_j\|}.
\]  

The relevance is further used to obtain the hard-attention map and the soft-attention map. 

**Hard-Attention.** We propose a hard-attention module to transfer the HR texture features \( V \) from the Ref image. Traditional attention mechanism takes a weighted sum of \( V \) for each query \( q_i \). However, such an operation may cause blur effect which lacks the ability of transferring HR texture features. Therefore, in our hard-attention module, we only transfer features from the most relevant position in \( V \) for each query \( q_i \).

More specifically, we first calculate a hard-attention map \( H \) in which the \( i \)-th element \( h_i := \arg \max_j r_{i,j} \) is calculated from the relevance \( r_{i,j} \):

\[
 h_i = \arg \max_j r_{i,j}.
\]

The value of \( h_i \) can be regarded as a hard index, which represents the most relevant position in the Ref image to the \( i \)-th position in the LR image. To obtain the transferred HR texture features \( T \) from the Ref image, we apply an index selection operation to the unfolded patches of \( V \) using the hard-attention map as the index:

\[
 t_i = v_{h_i},
\]

where \( t_i \) denotes the value of \( T \) in the \( i \)-th position, which is selected from the \( h_i \)-th position of \( V \).

As a result, we obtain a HR feature representation \( T \) for the LR image which will be further used in our soft-attention module.

**Soft-Attention.** We propose a soft-attention module to synthesize features from the transferred HR texture features \( T \) and the LR features \( F \) of the LR image from a DNN backbone. During the synthesis process, relevant texture transfer should be enhanced while the less relevant ones should be relived. To achieve that, a soft-attention map \( S \) is computed from \( r_{i,j} \) to represent the confidence of the transferred texture features for each position in \( T \):

\[
 s_i = \max_j r_{i,j},
\]

where \( s_i \) denotes the \( i \)-th position of the soft-attention map \( S \). Instead of directly applying the attention map \( S \) to \( T \), we first fuse the HR texture features \( T \) with the LR features \( F \) to leverage more information from the LR image. Such fused features are further element-wisely multiplied by the soft-attention map \( S \) and added back to \( F \) to get the final output of the texture transformer. This operation can be represented as:

\[
 F_{out} = F + \text{Conv(Concat}(F,T)) \odot S,\]

where \( F_{out} \) indicates the synthesized output features. \( \text{Conv} \) and \( \text{Concat} \) represent a convolutional layer and Concatenation operation, respectively. The operator \( \odot \) denotes element-wise multiplication between feature maps.

In summary, the texture transformer can effectively transfer relevant HR texture features from the Ref image into the LR features, which boosts a more accurate process of texture generation.

### 3.2. Cross-Scale Feature Integration

Our texture transformer can be further stacked in a cross-scale way with a cross-scale feature integration module. The architecture is shown in Figure 3. Stacked texture transformers output the synthesized features for three resolution scales (1x, 2x and 4x), such that the texture features of different scales can be fused into the LR image. To learn a better representation across different scales, inspired by [25, 37], we propose a cross-scale feature integration module (CSFI) to exchange information among the features of different scales. A CSFI module is applied each time the LR feature is up-sampled to the next scale. For the each scale inside the CSFI module, it receives the exchanged features from other scales by up/down-sampling, followed by a concatenation operation in the channel dimension. Then a convolutional layer will map the features into the original number of channels. In such a design, the texture features transferred from the stacked texture transformers are exchanged across each scale, which achieves a more powerful feature representation. This cross-scale feature integration module further improves the performance of our approach.

### 3.3. Loss Function

There are 3 loss functions in our approach. The overall loss can be interpreted as:

\[
 \mathcal{L}_{\text{overall}} = \lambda_{\text{rec}} \mathcal{L}_{\text{rec}} + \lambda_{\text{adv}} \mathcal{L}_{\text{adv}} + \lambda_{\text{per}} \mathcal{L}_{\text{per}}.
\]
Reconstruction loss. The first loss is the reconstruction loss:

\[ L_{\text{rec}} = \frac{1}{C\cdot H\cdot W} \| I^{HR} - I^{SR} \|_1, \]

where \((C, H, W)\) is the size of the HR. We utilize \(L_1\) loss which has been demonstrated to be sharper for performance and easier for convergence compared to \(L_2\) loss.

Adversarial loss. Generative adversarial networks [7] are proved effective in generating clear and visually favorable images. Here we adopt WGAN-GP [8], which proposes a penalization of gradient norm to replace weight clipping, resulting in more stable training and better performance. This loss can be interpreted as:

\[
\begin{align*}
\mathcal{L}_D &= \mathbb{E}_{\tilde{x} \sim p_g} [D(\tilde{x})] - \mathbb{E}_{x \sim p_r} [D(x)] + \\
&\quad + \lambda \mathbb{E}_{\tilde{x} \sim p_g} \left( \| \nabla_{\tilde{x}} D(\tilde{x}) \|_2 - 1 \right)^2, \\
\mathcal{L}_G &= -\mathbb{E}_{x \sim p_r} [D(x)].
\end{align*}
\]

Perceptual loss. Perceptual loss has been demonstrated useful to improve visual quality and has already been used in [13, 16, 22, 41]. The key idea of perceptual loss is to enhance the similarity in feature space between the prediction image and the target image. Here our perceptual loss contains two parts:

\[
\begin{align*}
\mathcal{L}_{\text{per}} &= \frac{1}{C_i H_i W_i} \left\| \phi_i^{\text{vgg}} (I^{SR}) - \phi_i^{\text{vgg}} (I^{HR}) \right\|_2^2 + \\
&\quad + \frac{1}{C_j H_j W_j} \left\| \phi_j^{\text{te}} (I^{SR}) - T \right\|_2^2,
\end{align*}
\]

where the first part is a traditional perceptual loss, in which \(\phi_i^{\text{vgg}}(\cdot)\) denotes the \(i\)-th layer’s feature map of VGG19, and \((C_i, H_i, W_i)\) represents the shape of the feature map at that layer. \(I^{SR}\) is the predicted SR image. The second part in our perceptual loss is a transferal perceptual loss, in which \(\phi_j^{\text{te}}(\cdot)\) denotes the texture feature map extracted from the \(j\)-th layer of the proposed LTE, and \((C_j, H_j, W_j)\) represents that layer’s shape. \(T\) is the transferred HR texture features in Figure 2. This transferal perceptual loss constrains the predicted SR image to have similar texture features to the transferred texture features \(T\), which makes our approach to transfer the Ref textures more effectively.

3.4. Implementation Details

The learnable texture extractor contains 5 convolutional layers and 2 pooling layers which outputs texture features in three different scales. To reduce the consumption of both time and GPU memory, the relevance embedding is only applied to the smallest scale and further propagated to other scales. For the discriminator, we adopt the same network used in SRNTT [41] and remove all BN layers. During training, we augment the training images by randomly horizontally and vertically rotating by 90°, 180° and 270°. Each mini-batch contains 9 LR patches with size 40 × 40 along with 9 HR and Ref patches with size 160 × 160. The weight coefficients for \(L_{\text{rec}}, L_{\text{adv}}\) and \(L_{\text{per}}\) are 1, 1e-3 and 1e-2, respectively. Adam optimizer with \(\beta_1 = 0.9, \beta_2 = 0.999\), and \(\epsilon = 1e-8\) is used with learning rate of 1e-4. We first warm up the network for 2 epochs where only \(L_{\text{rec}}\) is applied. After that, all losses are involved to train another 50 epochs.

4. Experiments

4.1. Datasets and Metrics

To evaluate our method, we train and test our model on the recently proposed RefSR dataset, CUFED5 [41]. The training set in CUFED5 contains 11,871 pairs, each pair consisting of an input image and a reference image. There are 126 testing images in CUFED5 testing set, each accompanied by 4 reference images with different similarity levels. In order to evaluate the generalization performance of TTSR trained on CUFED5, we additionally test TTSR on Sun80 [26], Urban100 [11], and Manga109 [20]. Sun80 contains 80 natural images, each paired with several reference images. For Urban100, we use the same setting as [41] to regard its LR images as the reference images. Such a design enables an explicit process of self-similar searching and transferring since Urban100 are all building images with strong self-similarity. For Manga109 which also lacks the reference images, we randomly sample HR images in this dataset as the reference images. Since this dataset is constructed with lines, curves and flat colored regions which are all common patterns. Even with a randomly picked HR Ref image, our method can still utilize these common patterns and achieve good results. The SR results are evaluated on PSNR and SSIM on Y channel of YCbCr space.

4.2. Evaluation

To evaluate the effectiveness of TTSR, we compared our model with other state-of-the-art SISR and RefSR methods. The SISR methods include SRCNN [3], MDSR [17], RDN [40], RCAN [39], SRGAN [16], ENet [22], ESRGAN [32], RSRGAN [38], among which RCAN has achieved state-of-the-art performance on both PSNR and SSIM in recent years. RSRGAN is considered to achieve the state-of-the-art visual quality. As for RefSR methods, CrossNet [43] and SRNTT [41] are two state-of-the-art methods recently, which significantly outperform previous RefSR methods. All experiments are performed with a scaling factor of 4 × between LR and HR images.

Quantitative Evaluation. For fair comparison, we follow the setting in SRNTT [41] to train all the methods
Table 1. PSNR/SSIM comparison among different SR methods on four different datasets. Methods are grouped by SISR methods (top) and RefSR methods (down). Red numbers denote the highest scores while blue numbers denote the second highest scores.

<table>
<thead>
<tr>
<th>Method</th>
<th>CUFED5</th>
<th>Sun80</th>
<th>Urban100</th>
<th>Manga109</th>
</tr>
</thead>
<tbody>
<tr>
<td>RDN [40]</td>
<td>25.95 / .769</td>
<td>29.63 / .806</td>
<td>25.38 / .768</td>
<td>29.24 / .894</td>
</tr>
<tr>
<td>RSRGAN [38]</td>
<td>22.31 / .635</td>
<td>25.60 / .667</td>
<td>21.47 / .642</td>
<td>25.04 / .803</td>
</tr>
<tr>
<td>SRNTT-rec [41]</td>
<td>26.24 / .784</td>
<td>28.54 / .793</td>
<td>25.50 / .783</td>
<td>28.95 / .885</td>
</tr>
<tr>
<td>SRNTT [41]</td>
<td>25.61 / .764</td>
<td>27.59 / .756</td>
<td>25.09 / .774</td>
<td>27.54 / .862</td>
</tr>
<tr>
<td>TTSR-rec</td>
<td>27.09 / .804</td>
<td>30.02 / .814</td>
<td>25.87 / .784</td>
<td>30.09 / .907</td>
</tr>
</tbody>
</table>

Figure 4. User study results. Values on Y-axis indicate the percentage of users that prefer TTSR over other approaches.

Table 2. Ablation study on texture transformer.

<table>
<thead>
<tr>
<th>Method</th>
<th>HA</th>
<th>SA</th>
<th>LTE</th>
<th>PSNR/SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td></td>
<td></td>
<td></td>
<td>26.34 / .780</td>
</tr>
<tr>
<td>Base+HA</td>
<td>✓</td>
<td></td>
<td></td>
<td>26.59 / .786</td>
</tr>
<tr>
<td>Base+HA+SA</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>26.81 / .795</td>
</tr>
<tr>
<td>Base+HA+SA+LTE</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>26.92 / .797</td>
</tr>
</tbody>
</table>

Table 3. Ablation study on CSFI.

<table>
<thead>
<tr>
<th>Method</th>
<th>CSFI</th>
<th>numC</th>
<th>param.</th>
<th>PSNR/SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base+TT</td>
<td></td>
<td>64</td>
<td>4.42M</td>
<td>26.927 / .797</td>
</tr>
<tr>
<td>Base+TT+CSFI</td>
<td>✓</td>
<td>64</td>
<td>6.42M</td>
<td>27.09 / .804</td>
</tr>
<tr>
<td>Base+TT(C80)</td>
<td></td>
<td>80</td>
<td>6.53M</td>
<td>26.93 / .797</td>
</tr>
<tr>
<td>Base+TT(C96)</td>
<td></td>
<td>96</td>
<td>9.10M</td>
<td>26.98 / .799</td>
</tr>
</tbody>
</table>

4.3. Ablation Study

In this section, we verify the effectiveness of different modules in our approach, including the texture transformer, the cross-scale feature integration, the adversarial loss and the transferal perceptual loss. In addition, we also discuss the influence of different reference similarity on TTSR.

Texture transformer. Our texture transformer contains mainly four parts: the learnable texture extractor (LTE), the relevance embedding module, the hard-attention module for feature transfer (HA) and the soft-attention module for feature synthesis (SA). Ablation results are shown in Table 2. We re-implement SRNTT [41] as our “Base” model by only removing all BN layers and Ref part. On top of the baseline model, we progressively add HA, SA, and LTE. Models without LTE use the VGG19 features to do relevance embedding. As we can see, when HA is added, the PSNR performance can be improved from 26.34 to 26.59, which verifies the effectiveness of the hard-attention module for feature transfer. When SA is involved, relevant texture features will be enhanced while the less relevant ones will be relieved during the feature synthesizing. This fur-
ther boosts the performance to 26.81. When replacing VGG with the proposed LTE, the PSNR is finally increased to 26.92, which proves the superiority of joint feature embedding in LTE.

To further verify the effectiveness of our LTE, we use the hard attention map to transfer the original image. It is expected that a better feature representation can transfer more accurate textures from the original images. Figure 6 shows the transferred original image by VGG19 in SRNTT and LTE in TTSR. In this figure, TTSR can transfer more accurate reference textures and generate a globally favorable result, which further proves the effectiveness of our LTE.

Cross-scale feature integration. On top of the texture transformer, CSFI can further enable texture recovery from different resolution scales (1×, 2× and 4×). We conduct an ablation study in Table 3. The first row shows the performance of our model with only TT, while the second row proves the effectiveness of CSFI, which brings 0.17
increase on PSNR metric. In order to verify that the performance improvement is not brought by the increase of parameter size, we increase the channel number of “Base+TT” model to 80 and 96. As we can see, there is almost no growth of “Base+TT(C80)” which has almost the same parameter number as “Base+TT+CSFI”. Even if we increase the parameter number to 9.10M to obtain “Base+TT(C96)” model, there is still a performance gap. This demonstrates that CSFI can efficiently utilize the reference texture information with a relatively smaller parameter size.

Adversarial loss. To make sure that the improvement of perceptual quality benefits from model design rather than the adversarial loss. We conduct an ablation among “Base-rec”, “Base”, TTSR-rec and TTSR, where TTSR can be interpreted as “Base+TT+HA+SA+LTE+CSFI”. The transferal perceptual loss constraints the LTE’s features between the predicted SR image and the transferred image $T$ to be similar. As shown in Figure 8, using this loss is able to transfer textures in a more effective way which achieves visually pleasing results. In addition, this loss also improves the quantitative metrics PSNR and SSIM of TTSR from 25.20/$\sigma_{757}$ to 25.53/$\sigma_{765}$. Extensive experiments demonstrate the superior performance of our TTSR over state-of-the-art RefSR approaches.

Table 4. Ablation study on reference images of different similarity.

<table>
<thead>
<tr>
<th>Level</th>
<th>CrossNet</th>
<th>SRNTT-rec</th>
<th>TTSR-rec</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
<td>25.48 / .764</td>
<td>26.15 / .781</td>
<td>26.99 / .800</td>
</tr>
<tr>
<td>L2</td>
<td>25.48 / .764</td>
<td>26.04 / .776</td>
<td>26.74 / .791</td>
</tr>
<tr>
<td>L3</td>
<td>25.47 / .763</td>
<td>25.98 / .775</td>
<td>26.64 / .788</td>
</tr>
<tr>
<td>L4</td>
<td>25.46 / .763</td>
<td>25.95 / .774</td>
<td>26.58 / .787</td>
</tr>
<tr>
<td>LR</td>
<td>25.46 / .763</td>
<td>25.91 / .776</td>
<td>26.43 / .782</td>
</tr>
</tbody>
</table>

5. Conclusion

In this paper, we propose a novel Texture Transformer Network for Image Super-Resolution (TTSR) which transfers HR textures from the Ref to LR image. The proposed texture transformer consists of a learnable texture extractor which learns a jointly feature embedding for further attention computation and two attention based modules which transfer HR textures from the Ref image. Furthermore, the proposed texture transformer can be stacked in a cross-scale way with the proposed CSFI module to learn a more powerful feature representation. Extensive experiments demonstrate the superior performance of our TTSR over state-of-the-art approaches on both quantitative and qualitative evaluations. In the future, we will further extend the proposed texture transformer to general image generation tasks.

Acknowledgement This paper is partially supported by NSFC (No. 61772330, 61533012, 61876109), the pre-research project (No. 61403120201), Shanghai Key Laboratory of Crime Scene Evidence (2017XCWZK01) and the Interdisciplinary Program of Shanghai Jiao Tong University (YG2019QNA09).
References


