Abstract—Convolutional neural networks have made great breakthrough in recent remote sensing image super-resolution tasks. Most of these methods adopt upsampling layers at the end of the models to perform enlargement, which ignores feature extraction in the high-dimension space and thus limits super-resolution performance. To address this problem, we propose a new super-resolution framework for remote sensing image to enhance the high-dimensional feature representation after the upsampling layers. We name the proposed method as Transformer-based Enhancement Network (TransENet), where transformers are introduced to exploit features at different levels. The core of the TransENet is a transformer-based multi-stage enhancement structure which can be combined with traditional super-resolution frameworks to fuse multi-scale high/low-dimension features. Specifically, in this structure, the encoders aim to embed the multi-level features in the feature extraction part and the decoders are used to fuse these encoded embeddings. Experimental results demonstrate that our proposed TransENet can improve super-resolved results and obtain superior performance over several state-of-the-art methods.

Index Terms—Super-resolution, remote sensing images, deep convolutional neural networks, transformer

I. INTRODUCTION

Image super-resolution (SR) is one kind of image processing technology, which aims to recover high-resolution (HR) images from low-resolution (LR) ones. It has been widely used in medical imaging [1], video monitoring [2] and remote sensing analysis [3, 4]. In the field of remote sensing, the ground targets in HR images own more clear edges and contours than the ones in LR images, and the HR images thus often play an important role in many high-level remote sensing tasks such as object detection [5], change detection [6] and semantic labeling [7]. Instead of developing physical imaging technologies, SR is an alternative way to effectively produce HR remote sensing images and has drawn much attention in recent years.

SR from one image is a typical ill-posed problem. Nowadays, most researchers leverage deep learning to obtain strong feature representations from a large amount of HR/LR image pairs [8]. Compared with the traditional learning-based algorithms such as neighborhood embedding-based [9], sparse representation-based [10][11] and local linear regression-based [12, 13] methods, the deep learning-based methods can automatically learn powerful feature representations and produce HR images with clearer edges and contours. Many specific structures are further proposed to enhance the performance, such as the residual block [14], recurrent structure [15][16], attention mechanism [17, 18].

For deep learning-based SR methods, an up-sampling operation is utilized to enlarge the LR input. According to the position of the upsampling operation, these existing methods can be divided into two categories: pre-upsampling framework [19, 20] and post-upsampling framework [14, 21–24]. In this paper, we proposed a new SR framework for remote sensing images. All these frameworks are illustrated in Fig. 1.

The Pre-upsampling Framework. This framework is adopted widely at the early stages of deep learning-based SR algorithm. It first performs an interpolation operation (such as bicubic interpolation) on LR input and enlarges it to the same size as the HR reference. Then a SR model is used to recover the HR image from the interpolated input. The SR model learns a nonlinear mapping between the interpolated LR input and the HR reference, without involving up-sampling operations, which reduces the learning difficulty to some extent. However, the computational cost significantly increases for a very deep network, since the feature extractions are all performed in the enlarged high-dimensional feature space.

The Post-upsampling Framework. In order to alleviate the problem of high computational cost, some researchers introduce the post-upsampling framework to construct an end-to-end SR architecture, in which the whole feature extractions are implemented in a low-dimensional space. For this purpose, the traditional up-sampling method is replaced with learnable upsampling layers, such as deconvolution [25] and sub-pixel convolution [26], which are inserted at the backend of the network and become one part of the SR model. Compared with the pre-upsampling framework, the computational cost reduction of this framework is proportional to the square of the preset magnification, where a feed-forward pass of a trained model can be significantly accelerated. This architecture design has become the mainstream in the image SR community [14, 21, 24]. However, for this framework, the HR image will be directly recovered after up-sampling layers without further perform enhancement of feature expression. It increases the difficulty of training and restricts the improvement of recon-
We propose a new SR framework named TransENet for remote sensing images, named Transformer-based Enhancement Network (TransENet), which aims at making full use of high-dimensional and low-dimensional features to further enhance the feature representation ability after upsampling layers. Moreover, we introduce transformer model [27] to leverage the features of different stages. Compared with the traditional convolution, the transformer can capture long-distance dependencies and effectively mine the correlation between high/low-dimensional features. Meanwhile, in order to utilize multi-level information in remote sensing images, we further design a transformer-based multi-stage enhancement structure which consists of multiple encoders and decoders. Specifically, the encoders are used to encode features of different stages in the feature extraction part, and the decoders perform multi-stage fusion with high/low-dimensional features to strengthen the expressive ability of high-dimensional features. It should be noted that this structure can be combined with most SR methods based on the post-upsampling framework.

The main contributions of this paper are summarized as follows:

- We propose a new SR framework named TransENet for remote sensing images to enhance the high-dimensional feature representation after upsampling layers. Transformers are introduced to leverage features at different stages. Our design can further improve super-resolved results and obtain state-of-the-art SR performance on two public remote sensing datasets.
- We design a transformer-based multi-stage enhancement structure. This structure can be combined with traditional SR framework to fuse multi-scale high/low-dimension features, where encoders aim to embed the multi-level features in the feature extraction and decoders are used to fuse these encoded features. Comprehensive ablation experiments verify the effectiveness of this design.

The rest parts of this paper are organized as follows. In Section [II] we provide detailed related works of image SR and transformer for image processing. The overview of the proposed TransENet and the Transformer-based multi-stage enhancement structure are carefully discussed in Section [III]. In Section [IV] ablation studies and quantitative and qualitative results are presented. Finally, the conclusions are drawn in Section [V].

II. RELATED WORK

A. CNN-based Natural Image SR

In recent years, convolutional neural networks (CNN) have greatly boosted the development of the natural image SR community. Different from the traditional methods [10–13], CNN-based methods often attempt to build an end-to-end network to directly learn a linear mapping from the given LR input to the HR reference. The up-sampling operation is usually utilized to complete the enlargement of the input image. Based on the position of the upsampling operation in CNN models, these methods can be divided into two categories of the pre-upsampling framework based and the post-upsampling framework based. Early methods are most based on the pre-upsampling framework. SRCNN [8] is the first shallow convolutional neural network to recover high-frequency information from an upsampled LR image. Kim et al. [20] introduced a very deep convolutional network (VDSR) with 20 layers to learn the image residual between the HR reference and the upsampled LR one. Recent post-upsampling framework based methods often incorporate de-convolution layers or sub-pixel convolution layers into the SR network. FSRCNN [28] directly adopts the original LR image as input and uses a deconvolution layer at the end of the model to perform upsampling. Lim et al. [14] improved residual blocks by getting rid of batch normalization, and several residual blocks are stacked to construct feature extraction part followed by a upsample block with sub-pixel convolution layers. From then, many researchers denote to developing the feature extraction part to learn better representations on low-dimension feature space. Zhang et al. [17] introduced residual channel attention to exploit interdependencies among feature channels. Mei et al. [18] proposed a cross-scale non-local attention module to leverage the long-range feature-wise similarities.

B. SR for Remote Sensing Images

Nowadays remote sensing image SR has attracted much attention. In early time, sparse representation-based methods led the researches. Pan et al. [29] first introduced the sparse representation and combined structure self-similarity prior to perform remote sensing image SR. Hou et al. [30] proposed a global joint dictionary model to recover remote sensing HR images. Shao et al. [31] developed a coupled sparse autoencoder to better learn the mapping between LR images and HR ones with sparse representation coefficients. In recent years, the deep learning-based methods [22, 24, 31, 32] have achieved much better performance than these early sparse representation-based methods. LGCNet [32] is the first CNN-based model for remote sensing image SR, where local and
global representations are both exploited to learn the image residual between HR images and the upscaled LR ones. Same with the trend in natural image SR field, most SR methods for remote sensing images adopt the post-sampling framework. Haut et al. [22] combined residual units, skip connections and network-in-network structure to extract more informative features. Qin et al. [23] introduced gradient maps to guide the proposed model to focus more on the edges of ground targets. Dong et al. [21] proposed a second-order learning strategy to capture multi-scale feature information. Meanwhile, some works introduce attention mechanism to further improve reconstructed results. MSAN [33] extracts multi-level features via a multi-scale attention design, and a scene-adaptive SR strategy is adopted to make the MSAN to better handle different scenes. HSENet [24] exploits the hybrid-scale self-similarity information in the remote sensing images using non-local attentions. Moreover, many researchers introduced generative adversarial networks (GAN) to improve the visual quality of the super-resolved. Jiang et al. [34] designed an edge-enhancement strategy to weaken the artifacts and noise caused by adversarial training. Lei et al. [21] introduced coupled adversarial training to learn better discriminative ability and achieved better visual quality.

C. Transformer for Image Processing

Transformer [27] has been widely used in the filed of natural language processing [35,37] and more recently, many attempts have been made to get rid of convolutions and adopt transformer models into computer vision tasks. ViT [38] is a pure transformer-based image classification model and achieves the state-of-the-art. There are also some CNN-transformer hybrid works. DETR [39] combines CNN backbone and the encoder-decoder transformer to build a fully end-to-end detector without anchor generation and non-maximum suppression post-processing. SETR [40] treats semantic segmentation as a sequence-to-sequence prediction task where the transformer is leveraged to accomplish global context model. Meanwhile, some researchers also try to generalize the transformer to low-level visual tasks. Parmar et al. [41] proposed Image Transformer to perform conditional image generation that can sequentially predict each pixel given its previous generated pixels. Jang et al. [42] built the first GAN using purely transformers (TransGAN), free of any convolution operation, and it can achieve high quality image synthesis. Moreover, Chen et al. [43] introduced a new pre-training model, namely, image processing transformer (IPT), to simultaneously handle many low-level computer vision tasks such as denoising, SR and deraining. IPT uses the encoder-decoder transformer as the main body of feature extraction part and is pre-trained on a large-scale dataset via contrastive learning. Different from IPT, our model aims to leverage the transformer to capture long range dependency between high-dimension and low-dimension features to enhance the final feature representation for remote sensing image SR.

III. METHODOLOGY

In this section, we introduce the Transformer-based Enhancement Network (TransENet) for remote sensing SR. The overall framework of TransENet is presented in Section III-A and the transformer-based multi-stage enhancement structure is carefully discussed in Section III-B. Besides, we will give a brief introduction to the implementation details in Section III-C.

A. Overview of TransENet

Fig. 2 illustrates the overall framework of our TransENet. Given a LR image $I_{LR}$, one convolutional layer is utilized to transform the input from RGB pixel space to feature space:

$$f_0 = \text{Conv}(I_{LR})$$

where the $\text{Conv}$ denotes a convolutional operation and the $f_0$ represents initial feature which will be the input of the following low-dimensional feature extraction part.

As shown in Fig. 2 in the low-dimensional feature extraction part, we use several feature extraction modules (FEM) to extract high-frequency details of the ground targets in remote sensing images from different scales. Specifically, we consider two basic components including basic blocks and residual blocks. The structure of the FEM constructed by some basic blocks is shown in Fig. 3 (a). The basic block consists of a convolutional layer and a non-linear function ReLU and uses a local skip connection to ease the training special for a deep model. Moreover, Fig. 3 (b) shows the structure of the FEM constructed by some residual blocks. The residual block is borrowed from ResNet [44] and is widely used in the field of image SR reconstruction [14, 21]. In the experimental part, we will use these two kinds of structure to verify the effectiveness of the transformer-based multi-stage enhancement. The entire low-dimensional feature extraction part is defined as:

$$f_n = FEM_n(f_{n-1}) = FEM_n(FEM_{n-1}(...FEM_1(f_0)....))$$

where $FEM_n$ represents the $n^{th}$ feature extraction module, and we use three FEMs in this paper considering of both speed and performance. Under this condition, the number of encoder modules is decided as 4 (3 for low-dimension feature embedding and 1 for high-dimension feature embedding) and the number of decoder modules is decided as 3.

After the feature extraction in the low-dimension feature space, we employ sub-pixel layer [26] to achieve the feature transformation from the low-dimension space to the high-dimension space.

$$f_{up} = \text{Subpixel}(f_n)$$

The low-dimension feature $f_1, ..., f_n$ and the high-dimension feature $f_{up}$ will be the input of the proposed transformer-based multi-stage enhancement structure, where several encoders and decoders are applied to perform feature enhancement. It should be noted that we reduce the feature dimension via $1 \times 1$ convolution considering of the efficiency of the TransENet. Finally, one convolutional layer is applied to obtained the final super-resolved HR image $I_{SR}$ based on the enhanced features.
We train the proposed model with L1 loss function. Given LR images $I_{LR}$ and the corresponding HR reference $I_{HR}$, the loss function can be obtained as

$$L(\theta) = \frac{1}{N} \sum_{i=1}^{N} ||I_{HR}^{(i)} - G_{\theta}(I_{LR}^{(i)})||_1.$$  \hspace{1cm} (4)

where $G_{\theta}$ is the proposed model with parameters $\theta$ and $G_{\theta}(I_{LR}^{(i)})$ is exactly the aforementioned $I_{SR}^{(i)}$, and $N$ is the number of training images.

### B. Transformer-based Multi-stage Enhancement

In this subsection, we introduce a transformer-based multi-stage enhancement structure to enhance the representation ability of the high-dimension feature after upsampling layers. This structure can be combined with traditional SR frameworks to fuse multi-scale high/low-dimension features, which is shown in Fig. 2. We use several transformers consisted of encoders and decoders to capture long-distance dependencies and effectively mine the correlation between high/low-dimensional features. Here, we take Encoder-3 and Decoder-3 in the Fig. 2 as examples to provide a clear description about the process of the feature enhancement which are carefully illustrated in Fig. 4.

#### Transformer Encoder

The standard Transformer takes a set of 1D sequences of token embedding as input. In order to handle 3D features, we split the feature $f \in \mathbb{R}^{H \times W \times C}$ into some patches and reshape them into a sequence of vectors $f_{p_i} \in \mathbb{R}^{P_H P_W C}$, $i = \{1, ..., N\}$, where $H$, $W$, and $C$ denote the height, the width and the number of channels of the feature maps, respectively. $P_H$ and $P_W$ are the height and the width of patches, and $N = \frac{H W}{P_H P_W}$ is the number of these patches and also is the length of the input sequence. Following [27, 38], the input vector size is usually fixed as $D$ dimension, and we need to map $f_{p_i}$ to $D$ with a trainable linear projection. However, different from the setting in [27, 38], the positional embedding is not involved for each feature patches and more detailed discussions will be provided in the next experimental part. Thus the input of the transformer encoder can be represented as

$$y_0 = [f_{p_1} W, f_{p_2} W, ..., f_{p_N} W]$$  \hspace{1cm} (5)

in which $W \in \mathbb{R}^{(P_H P_W C) \times D}$ is the linear projection matrix.

The main architecture of the encoder is following the original design in [27], which contains a multi-headed self-attention (MSA) module and a multi-layer perceptron (MLP) network. Referring to [38], we use the layer normalization (LN) before each module and local residual structures are utilized. The architecture of Encoder-3 is carefully illustrated in Fig. 4(a) and other encoders in our model have the same structure with the Encoder-3. The overall calculations of the encoder can be represented as

$$y_i = MSA(LN(y_{i-1})) + y_{i-1}, i = 1, \ldots, L_e$$

$$y_i = MLP(LN(y_i')) + y_i', i = 1, \ldots, L_e$$

$$[f_{E_1}, f_{E_2}, \ldots, f_{E_N}] = y_{L_e}$$  \hspace{1cm} (6)

where $f_{E_i}$ is the output of the encoder corresponding to $f_{p_i} W$, which own the same dimension with $f_{p_i} W$. Besides, the MLP has two layers in which GELU [46] non-linear function is used.

These encoders can encode the features of different stages of the proposed model, and then some decoders are applied
output of the decoder can be obtained as a function of the input features and the output of the encoder, which is the core part of the decoder. The multi-input MSA module in the Decoder can be formulated as:

\[
Atten = \text{softmax}(QK^T / \sqrt{d_k})
\]

\[
\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O
\]

where \( \text{head}_i = \text{Atten}(QW^Q_i, KW^K_i)VW^V_i \)

Specifically, the feature extraction module \( \text{FEM}_i(i = 1, 2, 3) \) extracts the feature representation after dimensionality reduction, and then enters the corresponding encoding module \( \text{Encoder}-i \) through block and linear mapping. The high-dimensional features after upsampling are encoded by the \( \text{Encoder}-4 \) module. In the subsequent feature enhancement process, high-dimensional features will be mainly used as the \( Q \) component in the Decoder, and the encoded low-dimensional features will be sequentially input as \( K \) and \( V \) into \( \text{Decoder}-1 \) to \( \text{Decoder}-3 \) to be combined with high-dimensional features. The combination process takes place in the multi-input MSA module in the Decoder can be formulated as:

\[
Atten = \text{softmax}(QK^T / \sqrt{d_k})
\]

\[
\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O
\]

where \( d_k \) denotes the dimensions of features in these decoders, \( h \) is the heads of the MSA module and \( W^Q_i, W^K_i, W^V_i \) and \( W^O_i \) are all projection matrices. It should be noted that in Fig. 2 the subscript of \( Q/K/V \) variables is decided according to the index of the related components. Take \( Q_{43} \) for an example, the subscript of this variable is decided by the related \( \text{Encoder}-4 \) and \( \text{Decoder}-3 \).

C. Implementation Details

This paper focuses on remote sensing image SR at three magnifications of \( \times 2, \times 3 \) and \( \times 4 \). In the training phase, \( 48 \times 48 \) patches are randomly extracted from LR remote sensing images as well as the reference patches from their corresponding HR ones. Meanwhile, we use random rotation (\( 90^\circ, 180^\circ \) and \( 270^\circ \)) and horizontal flipping to augment the training samples. In the test phase, the LR test images are cropped into a set of \( 48 \times 48 \) patches. We further use back-projection technology \([47, 48]\) to reduce the blocking effect in the preliminary results, so as to obtain the final HR reconstructed images. The parameter settings of the encoder and the decoder in our model are listed in Table I. The number of layers in the encoder is set to 8, and that in the decoder is set to 1. The detailed analyses and experiments are provided in the next section.

For optimization, we use Adam optimizer \([49]\) to train our model, where \( \beta_1 = 0.9, \beta_2 = 0.99 \) and \( \epsilon = 10^{-8} \). The initial learning rate is set to \( 10^{-4} \), and mini-batch size is set to 16. The total training epochs is 2000 and the learning rate will decrease half at 1500. The proposed method is implemented...
by PyTorch[50], and all experiments are run on a NVIDIA GeForce GTX 1080 Ti graphics card. Our codes will be publicly available at https://github.com/Shaosifan/TransENet.

IV. Experimental Results and Analyses

A. Experimental Data set and Settings

In this paper, we use two public remote sensing data sets including UCMeRed[51] and AID[52] to verify the effectiveness of the proposed method. These data sets have been widely used in the field of remote sensing SR [22, 23, 32].

- **UCMeRed dataset** [51]. This dataset contains 21 classes of remote sensing scenes including airport, baseball-diamond, beach, and etc. There are 100 images for each class with a size of 256 × 256 pixels, and the spatial resolution of these images is 0.3 m/pixel. We split this data set into two halves for train and test, where 20% of the training set are taken as validation.

- **AID dataset** [52]. This dataset consists of 10000 image in 30 classes of remote sensing scenes including airport, bareland, church, dense-residential, and etc. All images are in 600 × 600 pixels, and the spatial resolution is up to 0.5 m/pixel. For AID data set, 80% of the whole dataset are randomly selected to be the training set, and the remaining images are used as the test set. Moreover, we randomly select 5 images per class in total of 150 images to construct the corresponding validation.

In our experiments, the original image in each data set is regarded as a real HR reference, and the corresponding LR image is obtained via the bicubic interpolation, so as to construct HR/LR image pairs for the training and evaluation. All results are measured by peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM) [53].

B. Ablation Studies

In this section, we conduct a series of experiments on the UCMeRed dataset to explore the importance of each component in our method, where all models are trained with the same settings. For simplicity, these experiments are carried out with a magnification of \( \times 4 \).

**Effects of Encoders and Decoders.** The encoders and decoders are key components of the proposed method. We investigate the effect of these components with the aforementioned basic blocks and residual blocks. Table I lists super-resolved results with different settings, where the number of layers of the encoders and decoders in our model is set to 1. Comparing with the baseline model, our method with encoders and decoders can achieve significant improvement both on basic blocks and residual blocks. Specifically, our method obtain 0.19 dB and 0.18 dB higher in term of PSNR than the baseline model with basic blocks and with residual blocks, respectively. It verifies the effectiveness and versatility of the proposed framework on different blocks. According to the results in Table II, residual blocks are finally employed to construct the feature extraction modules, and the encoders and decoders are employed to enhance the features.

**Effects of Multi-stage Feature Enhancement.** The design of multi-stage feature enhancement aims to leverage the multiscale information in remote sensing images to obtain superior performance, where multiple decoders are involved to fuse high/low-dimensions feature stage-by-stage. Here, we investigate the effect of this design with different decoder configurations. It should be noted that when one certain decoder is added, the corresponding encoder will also be employed to fulfill feature embedding. Table III shows that the more decoders are involved, the better super-resolved performance will be achieved. At this time, more features at bottomed layers will pass to higher layers, and it relieves the difficulty in optimization and it beneficial to convergence of deep models. This phenomenon emphasizes the effectiveness of the multi-stage feature enhancement, and when these three decoders are used at the same time, the highest PSNR and SSIM values will be simultaneously obtained.

**Is Positional Embedding Important for This Task?** Position coding usually plays an important role in some transformer-based models, such as Bert [35], GPT [36] and ViT [38]. However, we find that the positional coding matters little in the proposed SR framework. To verify this point, we retrain the proposed model with or without learned positional encoding. In order to obtain a convincing comparison, we repeat the experiments for three times and report the mean and standard deviation of the results in Table IV where P.E. denotes the position coding, En. and De. denote these encoders and decoders respectively. It can be observed that the position coding does not improve the reconstruction results, and the model with the P.E. trends to have a little lower PSNR and SSIM. We speculate that the reason for this phenomenon lies in the fact that the proposed model can implicitly learn the
The table shows PSNR and SSIM results with or without position embedding. The table is as follows:

<table>
<thead>
<tr>
<th>P.E. in En.</th>
<th>P.E. in De.</th>
<th>PSNR</th>
<th>SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>×</td>
<td>×</td>
<td>27.76 ± 8.05e-3</td>
<td>0.7623 ± 4.12e-4</td>
</tr>
<tr>
<td>✓</td>
<td>×</td>
<td>27.73 ± 1.08e-2</td>
<td>0.7614 ± 1.70e-4</td>
</tr>
<tr>
<td>×</td>
<td>✓</td>
<td>27.72 ± 5.17e-3</td>
<td>0.7603 ± 2.52e-4</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
<td>27.72 ± 8.02e-3</td>
<td>0.7603 ± 7.22e-4</td>
</tr>
</tbody>
</table>

The table lists the results with different EN and DE layers settings. The table is as follows:

<table>
<thead>
<tr>
<th>En. Layers</th>
<th>De. Layers</th>
<th>PSNR</th>
<th>SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>27.76</td>
<td>0.7623</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>27.73</td>
<td>0.7619</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>27.75</td>
<td>0.7626</td>
</tr>
<tr>
<td>1</td>
<td>8</td>
<td>27.72</td>
<td>0.7622</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>27.76</td>
<td>0.7624</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>27.75</td>
<td>0.7627</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>27.77</td>
<td>0.7630</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>27.75</td>
<td>0.7625</td>
</tr>
</tbody>
</table>

The figure shows the training curves of the proposed method and the baseline method on UCmerced dataset. The figure is as follows:

![Training curves of the proposed method and the baseline method on UCmerced dataset.](image)

C. Comparisons with Other Methods

In this subsection, we compare the proposed method with some SR methods, including the classic bicubic interpolation, sparse coding (SC) [54], deep learning-based methods such as SRCNN [8], FSRCNN [28], VDSR [20], LGCNet [32], DCM [22] and DGANet-ISE [23]. Among them, SC, SRCNN, FSRCNN and VDSR are the approaches proposed for natural image SR task, while LGCNet, DCM and DGANet-ISE are recently proposed SR methods specifically designed for remote sensing images.

Quantitative Results on UCmerced Dataset.

The detailed results of different methods for all 21 scene classes of the UCmed dataset is provided in Table VII at a upscale factor of 3. We can see that TransENet can achieve the best PSNR values in 12 scene classes, while DCM performed better in the other 9 categories. Compared with the DCM model, TransENet is more effective in some scenes with rich edges and contours, such as buildings, dense residential, storage tanks, and tennis court. At the same time, the overall PSNR of the method is

---

1. All these 21 classes of UCmerced dataset: 1—Agricultural, 2—Airplane, 3—Baseballdiamond, 4—Beach, 5—Buildings, 6—Chaparral, 7—Denseresidential, 8—Forest, 9—Freeway, 10—Golfcourse, 11—Harbor, 12—Intersection, 13—Mediumresidential, 14—Mobilehomepark, 15—Overpass, 16—Parkinglot, 17—River, 18—Runway, 19—Sparseresidential, 20—Storage tanks, 21—Tennis court.
quantitative comparisons. In addition, Table IX lists the detailed outcomes on the 30 classes from the UCMerced dataset, this one is larger in amount and contains more scene categories in total of 30. The overall results of various methods on this dataset are shown in Table VIII. It can be seen that, compared with other methods, TransENet has the best results on these three magnifications. In addition, Table IX lists the detailed outcomes on the 30 classes with the magnification of 4 and the average PSNR is measured. It shows that TransENet achieves the best results on all the ground target scenes. From Table VIII and Table IX, it implies that when the size of data set increases, TransENet can obtain better results than DCM.

**Quantitative Comparisons.** In addition to quantitative comparison, we here provide a qualitative comparison of the recovered results with different methods. Fig. 4 shows some super-resolved examples of UCMerced dataset including ‘stadium’ and ’medium-residential’ scenes. Overall, comparing with other methods, the proposed method can obtain better results with clearer edges and contours which are also closer to the HR references.

**V. Conclusion**

In this paper, we propose a new SR framework for remote sensing images, namely, Transformer-based Enhancement Network (TransENet). TransENet aims at making full use of high/low-dimensional features and enhance the high-dimensional feature representation after the upsampling layers. The core part of the TransENet is a transformer-based multi-stage enhancement structure which can be combined with traditional SR frameworks to fuse multi-scale high/low-dimension features. In our TransENet, encoders aim to embed the multi-level features in the feature extraction and decoders are used to fuse these encoded features. Ablation studies have verified the effectiveness of the multi-stage enhancement structure. Meanwhile, experimental results on two public data sets show that compared with some state-of-the-arts, our method can obtain better super-resolved results.

**References**


TABLE VIII
MEAN PSNR (dB) AND SSIM OVER THE AID TEST DATA SET

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>32.39 / 0.8906</td>
<td>34.49 / 0.9286</td>
<td>34.80 / 0.9320</td>
<td>35.05 / 0.9346</td>
<td>35.21 / 0.9366</td>
<td><strong>35.28 / 0.9374</strong></td>
</tr>
<tr>
<td>3</td>
<td>29.08 / 0.7863</td>
<td>30.55 / 0.8372</td>
<td>30.73 / 0.8417</td>
<td>31.15 / 0.8522</td>
<td>31.31 / 0.8561</td>
<td><strong>31.45 / 0.8595</strong></td>
</tr>
<tr>
<td>4</td>
<td>27.30 / 0.7036</td>
<td>28.40 / 0.7561</td>
<td>28.61 / 0.7626</td>
<td>28.99 / 0.7753</td>
<td>29.17 / 0.7824</td>
<td><strong>29.38 / 0.7909</strong></td>
</tr>
</tbody>
</table>

Fig. 6. Result comparisons on UCMerced dataset with different methods.


Fig. 7. Result comparisons on AID dataset with different methods.


<table>
<thead>
<tr>
<th>class</th>
<th>Bicubic</th>
<th>SRCNN</th>
<th>LGCNet</th>
<th>VDSR</th>
<th>DCM</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>27.03</td>
<td>27.70</td>
<td>28.09</td>
<td>28.99</td>
<td>29.23</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>34.88</td>
<td>35.63</td>
<td>35.78</td>
<td>35.98</td>
<td>36.17</td>
<td>36.20</td>
</tr>
<tr>
<td>3</td>
<td>29.06</td>
<td>30.51</td>
<td>30.75</td>
<td>31.18</td>
<td>31.36</td>
<td>31.63</td>
</tr>
<tr>
<td>4</td>
<td>31.07</td>
<td>31.92</td>
<td>32.08</td>
<td>32.29</td>
<td>32.45</td>
<td>32.55</td>
</tr>
<tr>
<td>5</td>
<td>28.98</td>
<td>30.41</td>
<td>30.67</td>
<td>31.19</td>
<td>31.39</td>
<td>31.63</td>
</tr>
<tr>
<td>6</td>
<td>25.26</td>
<td>26.59</td>
<td>26.92</td>
<td>27.48</td>
<td>27.72</td>
<td>28.03</td>
</tr>
<tr>
<td>7</td>
<td>22.15</td>
<td>23.41</td>
<td>23.68</td>
<td>24.12</td>
<td>24.29</td>
<td>24.51</td>
</tr>
<tr>
<td>8</td>
<td>25.83</td>
<td>27.05</td>
<td>27.24</td>
<td>27.62</td>
<td>27.78</td>
<td>27.97</td>
</tr>
<tr>
<td>9</td>
<td>23.05</td>
<td>24.13</td>
<td>24.33</td>
<td>24.70</td>
<td>24.87</td>
<td>25.13</td>
</tr>
<tr>
<td>10</td>
<td>38.49</td>
<td>38.84</td>
<td>39.06</td>
<td>39.13</td>
<td>39.27</td>
<td>39.31</td>
</tr>
<tr>
<td>11</td>
<td>32.30</td>
<td>33.48</td>
<td>33.77</td>
<td>34.20</td>
<td>34.42</td>
<td>34.58</td>
</tr>
<tr>
<td>13</td>
<td>24.75</td>
<td>26.00</td>
<td>26.24</td>
<td>26.72</td>
<td>26.92</td>
<td>27.21</td>
</tr>
<tr>
<td>14</td>
<td>32.06</td>
<td>32.57</td>
<td>32.65</td>
<td>32.77</td>
<td>32.88</td>
<td>32.94</td>
</tr>
<tr>
<td>15</td>
<td>26.09</td>
<td>27.37</td>
<td>27.63</td>
<td>28.06</td>
<td>28.25</td>
<td>28.45</td>
</tr>
<tr>
<td>16</td>
<td>28.04</td>
<td>28.90</td>
<td>28.97</td>
<td>29.11</td>
<td>29.18</td>
<td>29.28</td>
</tr>
<tr>
<td>17</td>
<td>26.23</td>
<td>27.25</td>
<td>27.37</td>
<td>27.69</td>
<td>27.82</td>
<td>28.01</td>
</tr>
<tr>
<td>18</td>
<td>22.33</td>
<td>24.01</td>
<td>24.40</td>
<td>25.21</td>
<td>25.74</td>
<td>26.40</td>
</tr>
<tr>
<td>19</td>
<td>29.27</td>
<td>28.72</td>
<td>29.04</td>
<td>29.62</td>
<td>29.92</td>
<td>30.30</td>
</tr>
<tr>
<td>20</td>
<td>28.98</td>
<td>28.85</td>
<td>30.00</td>
<td>30.26</td>
<td>30.39</td>
<td>30.53</td>
</tr>
<tr>
<td>21</td>
<td>24.69</td>
<td>25.82</td>
<td>26.02</td>
<td>26.43</td>
<td>26.62</td>
<td>26.91</td>
</tr>
<tr>
<td>22</td>
<td>26.31</td>
<td>27.55</td>
<td>27.76</td>
<td>28.19</td>
<td>28.38</td>
<td>28.61</td>
</tr>
<tr>
<td>23</td>
<td>25.98</td>
<td>27.12</td>
<td>27.32</td>
<td>27.71</td>
<td>27.88</td>
<td>28.08</td>
</tr>
<tr>
<td>24</td>
<td>29.61</td>
<td>30.48</td>
<td>30.60</td>
<td>30.82</td>
<td>30.91</td>
<td>31.00</td>
</tr>
<tr>
<td>28</td>
<td>24.81</td>
<td>26.10</td>
<td>26.37</td>
<td>26.88</td>
<td>27.10</td>
<td>27.41</td>
</tr>
<tr>
<td>29</td>
<td>24.18</td>
<td>25.27</td>
<td>25.48</td>
<td>25.86</td>
<td>26.00</td>
<td>26.20</td>
</tr>
<tr>
<td>30</td>
<td>25.86</td>
<td>27.03</td>
<td>27.26</td>
<td>27.74</td>
<td>27.93</td>
<td>28.21</td>
</tr>
</tbody>
</table>

**Average:** 27.30 ± 28.30 ± 28.61 ± 28.99 ± 29.17 ± 29.38


