Tchebichef Transform Domain-Based Deep Learning Architecture for Image Super-Resolution

Ahlad Kumar, Senior Member, IEEE, Harsh Vardhan Singh, and Vijeta Khare, Senior Member, IEEE

Abstract—Recent advances in the area of artificial intelligence and deep learning have motivated researchers to apply this knowledge to solve multipurpose applications in the area of computer vision and image processing. Super-resolution (SR), in the past few years, has produced remarkable results using deep learning methods. The ability of deep learning methods to learn the nonlinear mapping from low-resolution (LR) images to their corresponding high-resolution (HR) images leads to compelling results for SR in diverse areas of research. In this article, we propose a deep learning-based image SR architecture in the Tchebichef transform domain. This is achieved by integrating a transform layer into the proposed architecture through a customized Tchebichef convolutional layer (TCL). The role of TCL is to convert the LR image from the spatial domain to the orthogonal transform domain using Tchebichef basis functions. The inversion of the transform mentioned earlier is achieved using another layer known as the inverse Tchebichef convolutional layer (ITCL), which converts back the LR images from the transform domain to the spatial domain. It has been observed that using the Tchebichef transform domain for the task of SR takes the advantage of high and low-frequency representation of images that makes the task of SR simplified. Furthermore, a transfer learning-based approach is adopted to enhance the quality of images by considering Covid19 medical images as an additional experiment. It is shown that our architecture enhances the quality of X-ray and CT images of COVID-19, providing a better image quality that may help in clinical diagnosis. Experimental results obtained using the proposed Tchebichef transform domain SR (TTDSR) architecture provides competitive results when compared with most of the deep learning methods employed using a fewer number of trainable parameters.

Index Terms—Convolutional neural network (CNN), COVID-19, deep learning, image super-resolution (SR), Tchebichef moments.

I. INTRODUCTION

The coronavirus disease (COVID-19) is a newly emerging viral disease that caused a worldwide pandemic. The World Health Organization has listed it as the sixth international public health emergency. It has impacted around 170 countries and has almost taken the lives of 2 million people as of January 2021. COVID-19 diagnosis by X-ray and CT images is very popular due to the quick results and that with great accuracy. This article proposes a deep learning-based SR architecture to enhance the quality of COVID-19 medical images for clinical diagnosis.

Image super-resolution (SR) is one of the most famous and significant ill-posed problems since there can be multiple possible solutions existing for one single image. It refers to the process of obtaining high-resolution (HR) images from the corresponding low-resolution (LR) images. SR plays an important role in wide variety of applications such as in security, mobile cameras, and medical imaging, which has attracted researchers over the past few years [1]. SR problems are categorized into single image SR (SISR) and multiimage SR (MISR) [2]–[5]. SISR aims to recover the HR image from a single LR image and the SISR-based methods have the edge over MISR-based methods, as they produce the better perceptual quality of the images. The conventional SR-based methods use dictionary-based approaches, consisting of two dictionaries of HR and LR images or patches [6]–[9]. These dictionaries are often learned with sparse-coding methods to get high-quality SR images. Some methods also use the dictionary features that need to be handcrafted [10]. Recent advances in deep learning methods have shown the state-of-the-art results in SR over the different datasets of images [11]. The earliest deep learning method was SRCNN [12] and later its improved version was reported in [13]. Recursive networks [14] and residual learning [15] is also incorporated in the deep learning architectures to boost the training process. It can be applied in other promising scenarios, such as blockchain, bigdata, or IoT systems [16]–[23].

We propose a new orthogonal domain-based deep learning architecture for image SR. It uses Tchebichef moment to transform spatial domain to orthogonal domain and then finds the difference between the Tchebichef coefficients of HR and LR image pairs. It has been observed that HR–LR image pairs have huge differences in the coefficient values at higher frequencies and small to negligible differences in the lower frequencies. This key observation plays a central role in developing the proposed architecture. The key aspects of this article are summarized as follows.

1) First time in this article, the application of Tchebichef moments, a member of the orthogonal moment family, is extended by using it with the deep learning architecture for the task of image SR. Earlier work
involving Tchebichef moments was restricted to pattern recognition and image processing tasks.

2) A recent article [24] published by one of the authors has shown that Tchebichef coefficients exhibit sparse behavior for images. This sparse behavior helps in accelerating the proposed architecture as it requires fewer coefficients to work with.

3) A deep neural network to solve the SR problem in an orthogonal transform domain is introduced. The architecture includes both the forward and inverse mapping, hence provides a complete pipeline for image SR.

4) The proposed architecture exploits the Tchebichef kernels to generate the representation of images in the transform domain. Two custom convolutional layers are designed; one for converting the image to transform domain [Tchebichef convolutional layer (TCL)] and the other, for doing inverse transformation [inverse TCL (ITCL)]. TCL layer is kept fixed and nontrainable, while the ITCL is trainable to get the optimized reconstruction kernels, which are used in ITCL layer for converting from transform domain to spatial domain.

5) The architecture consists of high and low-frequency paths. The high-frequency path employs Inception-Resnet-based structure with local residual connection to boost the training process. The low-frequency path employs a simple convolutional neural network (CNN) architecture.

6) To handle the artifacts that occur during the reconstruction phase after ITCL, additional convolutional layers are employed to process the spatial domain images, resulting in enhanced SR images.

7) We also explored the feasibility of applying transfer learning to the problem. Here, the proposed architecture pretrained on natural images will perform SR task on COVID-19-based medical images.

II. RELATED WORK

A. Single Image Super Resolution

The SISR is an ill-posed problem leading to multiple solutions, which makes it highly challenging to solve, and hence, SISR involves multiple solutions for one LR image or patch. SISR can mainly be divided into three categories: learning, interpolation, and reconstruction-based methods.

Learning-based SISR methods [7], [25] are computationally fast and have good performance. Specifically, sparse coding has shown compelling results. The method involves two dictionaries of images, separately for LR and HR patches. LR images or patches are represented in terms of their corresponding sparse code from the LR dictionary. Similarly, a HR image or patch is generated by the same sparse code, but it is applied to the HR dictionary. Interpolation SISR methods involve techniques, like bicubic interpolation [26] and Lanczos resampling [27], although they are speedy and straightforward, but lack satisfactory accuracy. Reconstruction-based methods [28]–[31] are time-consuming. SISR relies on advantages from prior knowledge but these methods highly degrade the SR image quality when the scaling factor is increased.

B. Deep Learning Advancements in Image SR

The conventional-based SR methods result in suboptimal performance due to their limited capabilities in formulating the exact model of the underlying problem. Recent advancements in computational architectures and effective research using deep neural networks have produced state-of-the-art results in the SR domain. Dong et al. [12] introduced a CNN network for the SR problem, known as SRCNN. It involved three layers of architecture that outperformed the previous methods based on sparse coding. SRCNN introduced the nonlinear mapping function between the LR and HR image. Image patches are fed to the CNN to generate the feature representation, followed by more convolutional layers to produce the higher representation of images. Although SRCNN had produced significant results, continuous advancements in the SR domain led to newer methods and architectures.

Since then, fully connected neural networks have also shown considerable improvements in SR by combining several ideas from different architectures, such as sparse coding and residual learning. Prior information-based deep neural networks has shown promising results. FSRNet [32] was used to generate the human face SR images. Wang et al. [33] focused on using the structural feature priors for effective recovery of detailed texture features in an image.

A huge improvement in SR image quality was observed using generative adversarial networks (GANs) [34]. Johnson et al. [35] focused on creating the photo-realistic image with high perceptual quality and focused less on maintaining the pixelwise difference between the images. GANs generate visually better images, but training a GAN is complex and time-consuming, making it unsuitable for many practical applications.

Li et al. [36] introduced the combined architecture using image transform domain as well as CNN and converted the input image/patch into the Fourier transform domain. The discrete Fourier transform (DFT) coefficients, thus obtained, are passed to a CNN architecture. As the convolution of the kernels and images in the spatial domain is equivalent to the multiplication of kernels and images in the Fourier domain, the claim is that the CNN architecture performs the elementwise multiplication to speed up the training process. Experimentally, it was observed that the performance was similar to previous works but not at par with state-of-the-art methods. Similar work is reported in [37] which uses a combination of transform domain and CNN. Here, the images are transformed using discrete cosine transform (DCT) using a custom convolutional layer.

This article exploits the image transform domain using Tchebichef basis functions integrated as kernels with the CNN network. A custom convolutional layer (TCL) is proposed for transforming the image into the orthogonal domain and the reverse using inverse transform (ITCL). The transform domain representations of an image are utilized to learn the high-frequency details lost during the degradation using
bicubic interpolation. The inverse transformation layer (ITCL) is kept trainable and optimizable. Hence, the kernels used in ITCL optimize with the training process. The optimized kernels obtained after the training process provides the best possible reconstruction of the images from their corresponding transform domain. Moreover, the reconstruction artifacts like softness and haziness are removed with the help of additional layers incorporated in the proposed architecture. The main objective of this article is to perform SR effectively by preserving the visual attributes of an image. For easy reference, some of the important symbols that have been used in the manuscript are listed in Table I.

### III. Tchebichef Moments

#### A. Computation of Tchebichef Moments

In this section, a brief review of the definition of Tchebichef moment [38] is discussed. They have been used recently in many pattern recognition [39]–[41] and denoising applications [24], [42]. Its robust feature representation capabilities allows to reconstruct images with promising results. The Tchebichef moment of order \((m + n)\) for an image with intensity function \(g(x, y)\) is given as [38]

\[
T_{n,m}(u) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \tilde{t}_n(x; N)\tilde{t}_m(y; N)g(x, y)
\]

(1)

with \(n, m = 0, 1, 2, \ldots, N - 1\). The image \(g(x, y)\), being of size \(N \times N\), \(\tilde{t}_n(x; N)\) and \(\tilde{t}_m(y; N)\) are the normalized Tchebichef polynomials, given by

\[
\tilde{t}_m(x; N) = \frac{t_m(x; N)}{\rho(m, N)}
\]

(2)

and

\[
\tilde{t}_n(y; N) = \frac{t_n(y; N)}{\rho(n, N)}
\]

(3)

with \(\rho(m, N) = (2m)!/(2m+1)\), \(\rho(n, N) = (2n)!/(2n+1)\) and \(t_n(x; N)\) is the \(N\)th order \(N\)-point Tchebichef polynomial defined as

\[
t_n(x; N) = n! \sum_{k=0}^{n} (-1)^{n-k} \binom{N-1-k}{n-k} \binom{n+k}{n} \binom{k}{n}. \]

(4)

To simplify the notation, \(\tilde{t}_n(x)\) is used to represent \(t_n(x; N)\). Here, \(\tilde{t}_n(x)\) is the orthonormal version of Tchebichef polynomials and it can be calculated using recurrence relation as [38]

\[
\tilde{t}_n(x) = \alpha_1(2x + 1 - N)\tilde{t}_{n-1}(x) + \alpha_2\tilde{t}_{n-2}(x)
\]

where

\[
\alpha_1 = \frac{1}{n} \sqrt{\frac{4n^2 - 1}{N^2 - n^2}}
\]

\[
\alpha_2 = 1 - \frac{n}{N} \sqrt{\frac{2n + 1}{2n - 3} \frac{N^2 - (n - 1)^2}{N^2 - n^2}}.
\]

(5)

The initial conditions for the above recurrence relationship are given as

\[
\tilde{t}_0(x) = 1/\sqrt{N}
\]

\[
\tilde{t}_1(x) = (2x + 1 - N) \sqrt{\frac{3}{N(N^2 - 1)}}.
\]

(6)

Image can be reconstructed back from Tchebichef moments using the inverse Tchebichef transformation as given by

\[
g(x, y) = \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} \tilde{t}_n(x; N)\tilde{t}_m(y; N)T_{n,m}(u).
\]

(7)

#### B. Matrix Form

Tchebichef moment in (1) can also be implemented in matrix format. The set of Tchebichef moments up to order \((m + n)\) in matrix form is given as

\[
T = PGQ^T
\]

(8)

where \(G\) is a square image matrix, \(P\) and \(Q\) are Tchebichef polynomials in matrix form up to orders of \(p\) and \(q\), respectively, given as

\[
P = \begin{bmatrix}
\tilde{t}_0(0) & \cdots & \tilde{t}_0(N-1) \\
\vdots & \ddots & \vdots \\
\tilde{t}_p(0) & \cdots & \tilde{t}_p(N-1)
\end{bmatrix}
\]

(9)

\[
Q = \begin{bmatrix}
\tilde{t}_q(0) & \cdots & \tilde{t}_q(N-1)
\end{bmatrix}
\]

(10)

Similarly, the inverse transformation given in (7) can be represented in a matrix form as

\[
G = P^TQ.
\]

(11)

#### C. Basis Functions of Tchebchief Moment

Tchebichef moments of an image can be interpreted as the projection of the image on the basis (kernel) functions, \(w_{pq}\) which is given as

\[
w_{pq} = [\tilde{t}_p]^T \tilde{t}_q
\]

(12)

where

\[
\tilde{t}_p = [\tilde{t}_p(0) \tilde{t}_p(1) \cdots \tilde{t}_p(N-1)]
\]

\[
\tilde{t}_q = [\tilde{t}_q(0) \tilde{t}_q(1) \cdots \tilde{t}_q(N-1)].
\]

(13)

The complete set of \(w_{pq}\) basis functions is shown in Fig. 1. Tchebichef moments can also be viewed as the correlation between the basis functions and image \(G\). A high value is recorded if there is a strong similarity between the content of the image and the basis function and vice versa.

---

**Table I**

| Symbol | Meaning |
|--------|---------|
| \(\tilde{T}()\) | Tchebichef Moment |
| \(G()\) | Inverse Tchebichef Moment |
| \(I_{\text{low}}\) | Low Frequency Components |
| \(I_{\text{high}}\) | High Frequency Components |
| \(W_{\text{low}}\) | Weight matrix for \(K^{\text{th}}\) layer |
| \(B_{\text{low}}\) | Bias matrix for the \(K^{\text{th}}\) layer |
| \(L()\) | Loss Function |
| \(F()\) | Non-Linear Mapping |
| \(\alpha\) | Regularization Parameter |

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
D. Basis Ordering and Its Significance

In the proposed architecture, Tchebichef basis functions are used as filters and are rearranged in a zig-zag order as shown in Fig. 2. This zig-zag reordering is inspired from the JPEG compression procedure [43].

Zig-Zag reordering of the basis functions allows exploiting the transform domain efficiently. We represent the 64 zig-zag reordered basis functions as \( \mathbf{w}^i \) where \( i = 0 \) to 63. It is observed that this particular reordering of basis functions results in an increased frequency pattern (complexity) in the basis functions, i.e., as the index \( i \) increases, the frequency content increases from low to high.

The average value of the coefficients generated by the convolution of the Tchebichef kernels with HR and LR images, respectively, is shown in Fig. 3. Here, Figs. 3(a) and (b) show the coefficients of an LR and HR version of the medical image, while Fig. 3(c) shows the differences between the HR and LR image coefficients. Please note that the values obtained in Fig. 3(c) are scaled for proper visualization. It can be observed from this figure that with the increase in kernels complexity, there is a substantial loss of the coefficients in the high frequency when compared to the lower frequency region.

In the Tchebichef domain, the problem of SR becomes recovering the high-frequency Tchebichef coefficients of the HR image from their corresponding LR images. This observation is incorporated in the proposed architecture discussed next. A similar analysis is carried out on natural images and the results are shown in Figs. 3(d)–(f).

IV. PROPOSED TCHEBICHEF TRANSFORM DOMAIN SR

In this section, a detailed description of the proposed architecture, shown in Fig. 4 for SR, is discussed. The architecture consists of the following blocks: 1) TCL; 2) frequency cube; 3) nonlinear mapping for low frequency; 4) inception-residual connection for high frequency; and 5) ITCL.

A. Network Structure

1) Tchebichef Convolutional Layer: This block transforms images from spatial domain to Tchebichef moment domain and has basis function \( \mathbf{w}_i \) as the kernels. There are 64 such kernels of size \( 8 \times 8 \) arranged in a zig-zag manner for increasing complexity with an increase in index \( i \) of the kernels. The detail about this is discussed later in Section III-D.

The transformation from spatial to Tchebichef moment domain works as follows: TCL layer creates a 64 feature maps \( f_0 \) for the entire image by performing convolution using \( \mathbf{w}_i \) with image \( G \) as given in (14). Here, \( \odot \) represents the convolution operation and is performed using the stride \( S = 1 \) and same padding to preserve the image’s dimension

\[
 f_i = \mathbf{w}_i \odot G \quad \forall i \in \{1, \ldots, 64\}. \tag{14}
\]

Kernels of the TCL layer are kept fixed, and nontrainable during the training phase as the primary role of this layer is to convert images into the transform domain.

2) Frequency Cube: Frequency domain feature maps \( f_{low} = f_{i=0,\ldots,63} \), obtained from (14), are used to form a cube (see label 2 marked in Fig. 4). This cube is reorganized version of Tchebichef coefficients, calculated for the whole image and is ordered in increasing frequency content (complexity). Based on the detailed discussion carried out for Fig. 3 in Section III-D, it has been observed that there is a substantial loss of the coefficients in the high-frequency region compared to the lower frequency region. Due to this reason, the partition of the frequency cube is done into two parts with a particular split point \( T \). The low and high frequency maps are defined as \( f_{low} = f_{i=1,\ldots,T} \) and \( f_{high} = f_{i=T+1,\ldots,63} \), respectively. Fig. 5 shows the details about this partitioning process. The calculation of the split point \( T \) is experimental, and its optimal value is obtained as 5. The discussion about its optimal value is carried out in the experimental section.

The proposed architecture process the partitioned cubes \( f_{low} \) and \( f_{high} \) separately. It can be observed from Fig. 3(c) and (f), that there is a higher amount of coefficient loss in the high-frequency region and thereby the high-frequency block \( f_{high} \) requires more robust and complex mapping to recover the HR image from LR image. On the other hand, the coefficient loss in the low-frequency region is not so significant but does play an important role in image quality. Hence, processing the low-frequency block \( f_{low} \) is done, using simpler nonlinear convolutional mapping to recover the image details. Next, we discuss the simple and complex deep learning architectures for \( f_{low} \) and \( f_{high} \).
Fig. 3. Top row: Tchebichef coefficients for (a) LR, (b) HR, and (c) difference in the coefficient of HR and LR for medical image. Bottom row: Tchebichef coefficients for (d) LR, (e) HR, and (f) difference in the coefficient of HR and LR for natural image.

3) Architecture for \( f_{\text{low}} \): The mapping of low-frequency coefficients of LR image to corresponding low-frequency coefficients of HR image is accomplished via CNN network consisting of two convolutional layers (see green arrow in Fig. 4). The first layer is a \( 5 \times 5 \) followed by \( 1 \times 1 \) convolutional layer. A leaky rectified linear unit (ReLU) is used as the activation function in both layers. The nonlinear mapping is given as

\[
\begin{align*}
\mathbf{z}_{\text{low}}^{[0]} & = f_{\text{low}} \\
\mathbf{z}_{\text{low}}^{[k]} & = \max(\mathbf{z}_{\text{low}}^{[k-1]} \odot \mathbf{W}_{1}^{[k]} + \mathbf{B}_{1}^{[k]}, \alpha f_{\text{low}}) \quad k \in \{1, 2\}
\end{align*}
\]  

(15)

where \( k \) represents the index to the two convolutional layers, \( \mathbf{z}_{\text{low}}^{[k]} \) is the output of \( k \)th layer, \( \mathbf{W}_{1}^{[k]} \) and \( \mathbf{B}_{1}^{[k]} \) are the weights...
and biases of the $k$th layer, $\alpha$ is the leaky ReLU parameter having value of 0.1. The nonlinear mapping of (15) recovers the information loss in the lower frequency spectra of the image.

4) Architecture for $f_{\text{high}}$: To recover the information loss in high-frequency spectra of the image, a nonlinear mapping is implemented using the deep learning architecture inspired from the inception network [44] (see green arrow in Fig. 4). The high-frequency feature maps $f_{\text{high}}$ of LR image is split into three convolutional paths, each of which are using different kernel sizes, namely, $3 \times 3$, $5 \times 5$, and $7 \times 7$. Larger kernel sizes are used to gather the global information, while the smaller kernel size collects information distributed more locally in the feature map. This allows the model to take advantage of the multilevel feature extraction. Finally, concatenating the features obtained from all the levels is done, followed by a $1 \times 1$ convolution, which serves two purposes. First, it creates a linear projection of a stack of feature maps, and secondly, it reduces the depth of the network. The nonlinear mapping for the process described above is given as

$$z_{\text{high}}^{[k]} = \max(f_{\text{high}} \odot W_2^{[k]} + B_2^{[k]}, \alpha f_{\text{high}}), \quad k \in \{1, 2, 3\}$$

(16)

$$z_{\text{high}}^{T} = \sum_{k=1}^{3} z_{\text{high}}^{[k]}$$

(17)

Here, $z_{\text{high}}^{T}$ is the combination of all the feature maps obtained via three parallel paths denoted by $k$.

5) Inverse Tchebichef Transformation Layer: This layer is required to transform image from Tchebichef moment domain to spatial domain. It takes input which is obtained by combining both, the low and high frequency cubes, $z_{\text{low}}^{[2]}$ and $z_{\text{high}}^{T}$, respectively. The output of this layer reconstruct the image $\hat{y}$ in spatial domain and is given as

$$\hat{y} = \sum_{i=1}^{64} w_i \odot (z_{\text{low}}^{[2]} + z_{\text{high}}^{T})_i,$$

(18)

Here, the weights of the Tchebichef kernel $w_i$ are trainable so that during the training process, the kernels adapt to the data and provides efficient inverse transformation.

6) Fine-Tuning Network: The reconstructed image $\hat{y}$ obtained using (18) is further processed through a small fine-tuning network shown in Fig. 4, which consist of three convolutional layers. The main purpose of introducing this additional network is to get rid of minor artifacts from the image $\hat{y}$.

B. Transfer Learning

Transfer learning is a method wherein a model developed for performing a particular task is reused for another task. It is a widely followed practice in the area of deep learning applications [45], [46]. We implemented the transfer learning approach in the SR of COVID-19 medical images by utilizing the pretrained weights obtained by training the TTDSR architecture on natural images. The architecture is trained in such a way that it is adaptable to both medical as well as natural images. A similar approach is adopted in [45], [47] wherein the authors have performed transfer learning on medical and hyperspectral images by exploiting the knowledge from the pretrained CNN using natural images, and thereby achieved comparable performance. A detailed explanation of the training mechanism involved can be found in the experimental section discussed next.

V. EXPERIMENTAL WORK

A. Training Details

This section discusses the training details of the proposed TTDSR architecture. In order to learn the end to end mapping function $F$ for SR task, optimized values of the network parameters $\theta \in (W_1^{[k]}, B_1^{[k]}, W_2^{[k]}, B_2^{[k]})$ are required. These parameters can be obtained by minimizing the loss between the network generated reconstructed SR image $F(Y_i; \theta)$, and the HR ground truth image $X_i$. Given the batch of HR images $X_i$ and the corresponding LR images $Y_i$, the loss function is given as

$$L(\theta) = \frac{1}{M} \sum_{i=1}^{M} \| F(Y_i; \theta) - X_i \|^2 + \lambda \sum_{j=1}^{l} W_j^2$$

(19)

where $M$ is the total number of training images, $\lambda$ is the regularization parameter, and $l$ is the total number of kernels used in the architectures discussed in Sections IV-A3 and IV-A4. The loss is minimized using the adaptive moment estimation, i.e., Adam [48] optimizer with standard back-propagation [49], that computes adaptive learning rates of each parameter. The training phase of the network involves Adam as optimizer with default parameters as: $\beta_1 = 0.9$ (the first-order moments), $\beta_2 = 0.999$ (the second-order moments), $\epsilon = 1e-7$ (a small constant for numerical stability). Learning rate $\eta$ is initialized as $1e-3$. The filter weights for each layer in the network are initialized with Glorot-uniform that draws the samples from the uniform distribution. It is observed that without using regularization, the network becomes highly unstable. Hence, L2 regularization of $\lambda = 0.01$ is applied to the network weights to penalize the weights. The TCL is kept nontrainable, while the ITCL is trainable to get the best optimized Tchebichef kernels. There are 14 convolutional layers in the
TTDSR architecture leading to a total number of parameters as 94k, out of which 90k are the trainable parameters, and the remaining are fixed parameters used in the TCL layer. The network is trained for 100 epochs with a batch size of 64. We have used high-quality images from T91 [50] and DIV2K [11] for training. Following [51]–[54] five standard benchmark datasets: Set5 [55], Set14 [56], BSDS100 [57], and Urban100 [58] are used for testing. Training and testing phase are conducted on NVIDIA GeForce RTX 2080 Ti GPU, with the Tensorflow as a support package.

B. Datasets

There are several widely used datasets for image SR. The combination of images from T91 [50] and DIV2K [11] dataset have been used to create a combined training dataset. From a robust model, a data augmentation technique is used, where the images are augmented using three methods, i.e.,

1) the images are rotated using $45^\circ$, $90^\circ$, $135^\circ$, $180^\circ$, $225^\circ$;  
2) horizontal and vertical flips of the images are done; and  
3) scaling of images is performed by factor of 0.6, 0.7, 0.8, and 0.9, respectively.

These augmented images are the variations in the HR image, which are then down-sampled by a factor of $\eta$. The down-sampled images are scaled up using the bi-cubic interpolation of the same factor $\eta$ to form the degraded LR images for training. Training images are first converted from RGB to YCbCr format. Inspired from [9], [15], [59], luminance ($Y$) channel is used as the input to the architecture while $Cb$ and $Cr$ channels are directly up-scaled, using the bi-cubic interpolation of LR images. Finally, combining the up-scaled $Cb$ and $Cr$ channels with the predicted luminance ($Y$), the channel from the architecture is used to generate the SR image, which is then converted back into $RGB$ format.

The proposed architecture is trained on a single channel, i.e., $Y$ in YCbCr domain, allowing for the flexibility of performing transfer learning on COVID-19 medical images, which are gray-scale images. SR results for the same can be analyzed in the experimental work.

During the testing phase, several standard datasets like Set5 [55], Set14 [56], BSDS100 [57] and Urban100 [58] have been utilized to evaluate the performance of the proposed architecture. The metrics used for image quality assessment are PSNR, and SSIM [60]. Few published methods work with larger datasets like DIV2K [11], ImageNet [61] and MS-COCO [62]. However, our choice of datasets for comparison has been used to keep it consistent with the majority of the methods that use it. The above datasets used for testing mainly consist of natural images. As pointed out earlier, transfer learning is carried out on the proposed architecture, which is now tested on medical images. For this, the COVID-19 image database contains a set of images collected by Cohen et al. [63] are used. The dataset contains chest X-ray and computed tomography (CT) images. The images are mainly in gray-scale format and is a collection of anterior-posterior view of chest X-rays. The dataset is continuously updated, and it is worth mentioning that the resolution of images varies from image to image. A sample of these images can be found in Fig. 6.

C. Comparative Analysis

In this section, the performance of the proposed architecture with the other existing methods is discussed. For this, several standard datasets (Section V-B) are being used for its evaluation. Here, the following methods are used for comparison with our architecture.

1) ScSR [9]: Sparse coding-based SR method, constructs LR-HR image patch dictionary.
2) A+ [64]: Adjusted anchored neighborhood regression for fast SR is the updated and modified version of [65].
3) SelfEx [58]: Self similarity-based method that measures the similarity within the images.
4) SCN [66]: Sparse prior method implemented with the help of CNN.
5) SRCNN [12]: Earliest deep learning method for image SR based on CNN architecture.
6) FSRCNN [13]: An advanced and modified version of SRCNN with deeper architecture and transpose convolution approach.
7) VBPS [67]: Recent method for image SR that exploits the inherent self-similarities found in images.

Tables II and III report the PSNR and SSIM results of TTDSR and other methods, respectively. Out of all the methods, FSRCNN and VBPS give competitive scores when compared with TTDSR. However, TTDSR, on average, performs well on all kinds of datasets. We now make a subjective comparison of SR results using various methods. Figs. 7 and 8 show the SR results for the TTDSR and other methods in enhancing the quality of the image degraded due to bi-cubic interpolation. Fig. 7 shows the SR results on monarch image. The enlarged version shows the thin black edge at the head of the monarch image. It can be observed that the bi-cubic interpolated image shows a heavy loss of thin edge details with discontinuity. Also, other methods fail to generate the edges gracefully. The proposed TTDSR architecture generates a clear edge, overcomes the discontinuity artifact observed in

![Sample images from COVID-19 dataset which contains both X-ray and CT images.](image-url)
other methods, and gives better PSNR and SSIM. Fig. 8 shows the SR results on bi-cubic interpolated zebra image. It can be observed that the black and white strips present on zebra lacks details and fails to capture the orientation of the edges. Further, the FSRCNN method shows slightly better results than our method in terms of PSNR and SSIM, but the diagonal edges overlap, leading to poor image visualization. On the other hand, though TTDSR gives second-best results, it exploits the frequency domain details to overcome this degradation and thereby generate correctly oriented black and white strips.

Next, SR results on COVID-19 medical images is carried out as an additional experiment. Here, the aim is not to detect the infection of COVID-19 through these images but to enhance the quality of these image to provide a better diagnosis. It is worth mentioning that the architecture enhances the quality of medical images and has the potential for better diagnosis due to the improved quality of images. But it might add some minor artifacts that can make the task more difficult, particularly for the medical use case. However, when we visualize the natural and medical images, we have not observed any significant artifacts that corrupt the quality of an image. The dataset used for this purpose is presented in Cohen et al. [63]. Our model is trained on a single channel, i.e., Y(luminance). Medical images are gray-scale images containing only luminance information of the pixel and no color information. This provides the flexibility to use TTDSR architecture on medical images using transfer learning. The SR results on the image can be seen in Fig. 9. The average
Table V

| Performance Comparison Using Various Methods |
|---------------------------------------------|
| Methods          | Scale | FLOPs | Trainable Parameters | Memory   | Running Time (sec) |
|------------------|-------|-------|----------------------|----------|--------------------|
| SCN [66]         | × 2   | -     | 3kB                  | 0.52MB   | -                  |
| SRCNN [12]       | × 2   | 52.7G | 57kB                 | 0.086MB  | 13.14              |
| FSRCNN [13]      | × 2   | 6G    | 12kB                 | 0.038MB  | 12.025             |
| SRGAN [34]       | × 2   | 166G  | 1.5MB                | 6.08MB   | 16.278             |
| TTD[35]          | × 2   | 94G   | 94kB                 | 1.4MB    | 0.1772             |

PSNR and SSIM comparison on the COVID-19 dataset can be seen in Table IV. It can be observed that the proposed method gives better results compared to other methods.

In addition, Table V compared our method with the existing ones using various parameters such as FLOPs, trainable parameters, memory consumed, and running time. It can be observed that the running time of the proposed method is the shortest. SRGAN takes the longest time. Due to wide range of convolutional layers, SRCNN takes longer time between LR and HR image patch pairs. FSRCNN performs better than SRCNN, however, the visual results obtained are not that good. The number of trainable parameters, memory used and FLOPs are less compared to SRGAN.

D. Network Parameters and Its Impact

1) Split Point for Tchebichef Frequency Cube: Tchebichef polynomials are treated as filters and create a frequency cube in the image transform domain as shown in Fig. 5. In the architecture (see Fig. 4), it can be observed that there are two subnetworks, one for recovering the loss in high-frequency content and the other works for recovering loss in low-frequency content. The frequency cube is split into two halves at a split point $T$. Selecting a smaller split point $T$ generates a fraction of frequency cube containing less number of lower frequency components and vice versa. The splitting of the frequency cube is experimental, and the performance of the network varies with the different values of $T$. However, we observe that $T < 5$ shows the equal contribution to frequency spectra, which reveals that the lower frequency components are shared, and the majority loss can be seen in the higher frequency components. This concludes that we can combine the lower frequency components up to the threshold value $T$, making it the most appropriate and optimal split point. In this article, the value of $T$ is taken as 5 based on the experiment conducted in Fig. 10, where for different datasets, the average PSNR reported by the network is highest when $T$ is taken as 5.

2) Impact of Residual Connection: Initially, the network was structured without the residual connection and only relied on the inception-based module. In this case, the network performed with limited capability due to the vanishing gradient.
To overcome this problem, two major residual connections are added in the network, local residual connection for high-frequency components ($f_{\text{high}}$) and the overall global residual connection for both high and low-frequency component. The local residual connection [68] in the inception module was introduced to boost the gradients in the training phase for the recovery of high-frequency components of the image. The use of local residual connectivity overcomes the vanishing gradient problem and helps the optimizer reach the minima faster. Experimental analysis for same is shown in Fig. 11, where it can be observed that the architecture with residual connections converges to smaller loss $L$ when used without residual connections.

3) Optimized Learned Tchebichef Filters: As discussed, the network architecture consists of two custom layers, namely TCL and ITCL. The kernel functions used in TCL are fixed,
whereas in that of ITCL are kept trainable to adapt to the training phase of the network. Fig. 12 shows the optimized kernels obtained after the training process. These optimized kernels are used to reconstruct the image from the Tchebichef moment domain. Hence, they give part of its contribution in providing better image quality compared to other methods.

VI. CONCLUSION

A deep learning architecture for SR of natural and COVID-19 medical images is presented. It uses the Tchebichef transform domain that helps in exploiting the low and high-frequency details present in the images to enhance its quality. A detailed analysis of various parameters that affect the performance of the architecture is discussed. The objective and visual comparison of the SR results with the existing methods shows that the proposed architecture provides superior results when evaluated in terms of average PSNR and SSIM metrics. The visual comparison of the result shows that our work restores the details present in an image effectively. This work opens up the possibility to explore SR architectures using different transform domains.

REFERENCES

[1] S. C. Park, M. K. Park, and M. G. Kang, “Super-resolution image reconstruction: A technical overview,” IEEE signal Process. Mag., vol. 20, no. 3, pp. 21–36, May 2003.
[2] S. Farsiu, M. D. Robinson, M. Elad, and P. Milanfar, “Fast and robust multiframe super resolution,” IEEE Trans. Image Process., vol. 13, no. 10, pp. 1327–1344, Oct. 2004.
[3] S. Farsiu, D. Robinson, M. Elad, and P. Milanfar, “Advances and challenges in super-resolution,” Int. J. Imag. Syst. Technol., vol. 14, no. 2, pp. 47–57, 2004.
[4] Q. Yuan, L. Zhang, and H. Shen, “Multiframe super-resolution employing a spatially weighted total variation model,” IEEE Trans. Circuits Syst. Video Technol., vol. 22, no. 3, pp. 379–392, Mar. 2012.
[5] X. Li, Y. Hu, X. Gao, D. Tao, and B. Ning, “A multi-frame image super-resolution method,” Signal Process., vol. 90, no. 2, pp. 405–414, Feb. 2010.
[6] S. Ma and G. Yu, “Super-resolution with sparse mixing estimators,” IEEE Trans. Image Process., vol. 19, no. 11, pp. 2889–2900, Nov. 2010.
[7] H. Chang, D.-Y. Yeung, and X. Yiong, “Super-resolution through neighbor embedding,” in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2004, pp. 1–8.
[8] D. Glasner, S. Bagon, and M. Irani, “Super-resolution from a single image,” in Proc. IEEE Int Conf. Comput. Vis., Sep. 2009, pp. 349–356.
[9] J. Yang, J. Wright, T. S. Huang, and Y. Ma, “Image super-resolution via sparse representation,” IEEE Trans. Image Process., vol. 19, no. 11, pp. 2861–2873, Nov. 2010.
[10] L. Zhang and W. Zuo, “Image restoration: From sparse and low-rank priors to deep priors [lecture notes],” IEEE Signal Process. Mag., vol. 34, no. 5, pp. 172–179, Sep. 2017.
[11] R. Timofte et al., “Ntire 2017 challenge on single image super-resolution: Methods and results,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), Jul. 2017, pp. 1110–1121.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
[37] T. Guo, H. S. Mousavi, and V. Monga, “Adaptive transform domain image super-resolution via orthogonally regularized deep networks,” IEEE Trans. Image Process., vol. 28, no. 9, pp. 4685–4700, Sep. 2019.

[38] R. Mukundan, S. H. Ong, and P. A. Lee, “Image analysis by Tchebichef moments,” IEEE Trans. Image Process., vol. 10, no. 9, pp. 1357–1364, Sep. 2001.

[39] H. Zhu, H. Shu, T. Xia, L. Luo, and J. L. Coatrieux, “Translation and scale invariants of Tchebichef moments,” Pattern Recognit., vol. 40, no. 9, pp. 2530–2542, 2007.

[40] B. Xiao, J.-F. Ma, and J.-T. Cui, “Radial Tchebichef moment invariants for image recognition,” J. Vis. Commun. Image Represent., vol. 23, no. 2, pp. 381–386, Feb. 2012.

[41] H. Wu and S. Yan, “Computing invariants of Tchebichef moments for shape based image retrieval,” Neurocomputing, vol. 215, pp. 110–117, Nov. 2016.

[42] A. Kumar, M. O. Ahmmed, and M. Swamy, “Image denoising via overlapping group sparsity using orthogonal moments as similarity measure,” ISA Trans., vol. 85, pp. 293–304, Feb. 2019.

[43] G. K. Wallace, “The JPEG still picture compression standard,” IEEE Trans. Consum. Electron., vol. 38, no. 1, pp. 17–34, Feb. 1992.

[44] C. Szegedy et al., “Going deeper with convolutions,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 1–9.

[45] A. Zhong, B. Li, N. Luo, Y. Xu, L. Zhou, and X. Zhen, “Image restoration for low-dose CT via transfer learning and residual network,” IEEE Access, vol. 8, pp. 112078–112091, 2020.

[46] S. J. Pan and Q. Yang, “A survey on transfer learning,” IEEE Trans. Knowl. Data Eng., vol. 22, no. 10, pp. 1345–1359, Oct. 2009.

[47] Y. Yuan, X. Zheng, and X. Lu, “Hyperspectral image superresolution by transfer learning,” IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 10, no. 5, pp. 1963–1974, May 2017.

[48] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. Int. Conf. Learn. Represent., 2014, pp. 1–15.

[49] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proc. IEEE, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[50] S. Schuster, C. Leistner, and H. Bischof, “Fast and accurate image upsampling with super-resolution forests,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 3791–3799.

[51] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, “Enhanced deep residual networks for single image super-resolution,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), Jul. 2017, pp. 136–144.

[52] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, “Residual dense network for image super-resolution,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 2472–2481.

[53] K. Zhang, W. Zuo, and L. Zhang, “Learning a single convolutional super-resolution network for multiple degradations,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 3262–3271.

[54] Z. Li, J. Yang, Z. Liu, X. Yang, G. Jeon, and W. Wu, “Feedback network for image super-resolution,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 3867–3876.

[55] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. A. Morel, “Low-complexity single-image super-resolution based on nonnegative neighbor embedding,” in Proc. Brit. Mach. Vis. Conf., 2012, pp. 1–10.

[56] R. Zeyde, M. Elad, and M. Prott, “On single image scale-up using sparse-representations,” in Proc. Int. Conf. Curves Surfaces, 2012, pp. 711–730.

[57] D. Martin, C. Fowlkes, D. Tal, and J. Malik, “A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics,” in Proc. IEEE Int. Conf. Comput. Vis., vol. 2, Feb. 2001, pp. 416–423.

[58] Y. Zheng, F. Li, and Y. Zhang, “Learning applied to document recognition,” IEEE Trans. Consum. Electron., vol. 85, pp. 293–304, Feb. 2019.

[59] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” IEEE Trans. Image Process., vol. 13, no. 4, pp. 600–612, Apr. 2004.

[60] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2009, pp. 248–255.

[61] T.-Y. Lin et al., “Microsoft COCO: Common objects in context,” in Proc. Eur. Conf. Comput. Vis. (ECCV), D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., 2014, pp. 740–755.

[62] I. Paul Cohen, P. Morrison, and L. Dao, “COVID-19 image data collection,” 2020, arXiv:2003.11597.

[63] R. Timofte, V. De, and L. Gool, “A+: Adjusted anchored neighborhood regression for fast super-resolution,” in Proc. Asian Conf. Comput. Vision (ACCV), 2014, pp. 111–126.

[64] R. Timofte, V. De, and L. V. Gool, “Anchored neighborhood regression for fast example-based super-resolution,” in Proc. IEEE Int. Conf. Comput. Vis., Dec. 2013, pp. 1920–1927.

[65] Z. Wang, D. Liu, J. Yang, W. Han, and T. Huang, “Deep networks for image super-resolution with sparse prior,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 370–378.

[66] G. Chantas, S. N. Nikolopoulos, and I. Kompatsiaris, “Heavy-tailed self-similarity modeling for single image super resolution,” IEEE Trans. Image Process., vol. 30, pp. 838–852, 2021.

[67] Z. Li, J. Yang, Z. Liu, X. Yang, G. Jeon, and W. Wu, “Feedback network for image super-resolution,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 3867–3876.

[68] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. A. Morel, “Low-complexity single-image super-resolution based on nonnegative neighbor embedding,” in Proc. Brit. Mach. Vis. Conf., 2012, pp. 1–10.

[69] R. Zeyde, M. Elad, and M. Prott, “On single image scale-up using sparse-representations,” in Proc. Int. Conf. Curves Surfaces, 2012, pp. 711–730.