iWave: CNN-Based Wavelet-Like Transform for Image Compression

Haichuan Ma, Dong Liu, *Senior Member, IEEE*, Ruiqin Xiong, *Senior Member, IEEE*, and Feng Wu, *Fellow, IEEE*

***Abstract*—Wavelet transform is a powerful tool for multiresolution time-frequency analysis. It has been widely adopted in many image processing tasks, such as denoising, enhancement, fusion, and especially compression. Wavelets lead to the successful image coding standard JPEG-2000. Traditionally, wavelets were designed from the signal processing theory with certain assumption on the signal, but natural images are not as ideal as assumed by the theory. How to design content-adaptive wavelets for natural images remains a difficulty. Inspired by the recent progress of convolutional neural network (CNN), we propose iWave as a framework for deriving wavelet-like transform that is more suitable for natural image compression. iWave adopts an update-first lifting scheme, where the prediction filter is a trained CNN, to achieve wavelet-like transform. The CNN can be embedded into a deep network that is analogous to an auto-encoder, which is trained end-to-end. The trained wavelet-like transform still possesses the lifting structure, which ensures perfect reconstruction, supports multiresolution analysis, and is more interpretable than the deep networks trained as "black boxes." We perform experiments to verify the generality as well as the speciality of iWave in comparison with JPEG-2000. When trained with a generic set of natural images and tested on the Kodak dataset, iWave achieves on average 4.4% and up to 14% BD-rate reductions. When trained and tested with a specific kind of textures, iWave provides as high as 27% BD-rate reduction.**

***Index Terms*—Convolutional neural network (CNN), image compression, lifting scheme, wavelet transform.**

I. Introduction

WAVELET transform, as a powerful tool for multiresolution time-frequency analysis, has been widely adopted in many image processing tasks, such as denoising, enhancement, fusion, and especially compression. Due to the local support of bases, wavelet transform has advantage over Fourier transform series that have global support, when processing signal with point singularities. Wavelet transform leads to the successful image coding standard JPEG-2000 [1], which achieves higher compression efficiency than JPEG [2] that employs discrete cosine transform (DCT).

However, it has been reported that wavelet transform still has limitations when dealing with natural images. First, wavelet transform for two-dimensional images is usually performed by two steps of one-dimensional transform along the horizontal and vertical directions respectively. This causes the inefficiency in handling image features that are neither horizontal nor vertical. In fact, such directional features will result in large magnitude high-frequency wavelet coefficients, leading to deteriorated performance in the compression task. Second, wavelet transform is usually performed uniformly over an entire image, i.e. a same set of transform kernels are applied everywhere. However, natural images are equipped with locally variant features, so there is a conflict between the uniform transform and the non-uniform image. Note that the video coding technology, mostly based on DCT, has introduced variable block size, different intra prediction modes, and adaptive transform kernels to deal with the images with locally variant features. Nonetheless, there is difficulty in directly implanting these techniques into wavelet transform.

Several new kinds of wavelet-like transforms have been proposed as remedies to the above limitations, such as ridgelet [3], [4], curvelet [5], contourlet [6], and so on, which provide good representations for images with arbitrary directional features. However, these transforms adopt over-complete bases, resulting in too many transform coefficients to be efficiently compressed. Another series of methods specifically designed for image compression choose to divide image into blocks, select a direction for each block, and perform transform along that direction, where the direction shall be signaled to decoder. These methods usually leverage the *lifting* scheme to embrace the flexibility of arbitrary directional transform, e.g. adaptive directional lifting (ADL) based wavelet transform [7], weighted adaptive lifting (WAL) based wavelet transform [8], and so on. These methods achieve higher compression efficiency than JPEG-2000, but at the cost of increased complexity and side information (block-wise directions). Moreover, all the aforementioned methods concern directional features, but there are also non-directional features, such as irregular textures, in natural images. How to cope with the complex and diverse nature of image features by hand-crafted wavelets remains a difficulty.

Recently, convolutional neural network (CNN) achieves a series of successes in different image processing tasks, such

Manuscript received March 29, 2019; revised August 13, 2019; accepted November 25, 2019. Date of publication December 12, 2019; date of current version June 23, 2020. This work was supported in part by the National Key Research and Development Plan under Grant 2017YFB1002401, and in part by the Natural Science Foundation of China under Grants 61772483, 61425026, and 61931014. The guest editor coordinating the review of this manuscript and approving it for publication was Dr. Jingdong Wang. *(Corresponding author: Dong Liu.)*

H. Ma, D. Liu, and F. Wu are with the CAS Key Laboratory of Technology in Geo-Spatial Information Processing and Application System, University of Science and Technology of China, Hefei 230027, China (e-mail: hcma@mail.ustc.edu.cn; dongeliu@ustc.edu.cn; fengwu@ustc.edu.cn).

R. Xiong is with the Institute of Digital Media, School of Electronic Engineering and Computer Science, Peking University, Beijing 100871, China (e-mail: rqxiong@pku.edu.cn).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2019.2957990

as recognition, segmentation, super-resolution, and more related, image compression [9]–[16]. CNN has several distinct advantages for dealing with natural images. First, the two-dimensional convolutional kernels in CNN make it suitable to capture the two-dimensional features with arbitrary direction in images. Second, the localized receptive fields and the ensemble of multiple convolutional kernels make it possible to effectively process the different local features simultaneously. Third, and probably the most important, the kernels in CNN are learned from massive images rather than manually designed, which implies that CNN is more specialized for natural images. However, a remaining issue of CNN is its interpretability, especially regarding the network structure: why should the CNN be constructed in this way rather than in that way? There are intuitive interpretations and insightful observations, but there are lacking a systematic and mathematical answer.

In view of the shortcomings of the traditional wavelet/wavelet-like transforms, as well as the successes of CNN, we propose a new CNN-based wavelet-like transform, termed iWave in this paper. Our key idea is originated from the lifting scheme, which decomposes a wavelet transform into a series of prediction and update operations [17]–[19]. While for the traditional wavelet transforms, the prediction/update operations are linear; for the proposed iWave, the prediction/update operations are fulfilled by trained CNNs, being non-linear and signal-adaptive. In other words, iWave adopts the lifting scheme and integrates CNNs as building blocks into the lifting scheme. As long as the CNNs were specifically trained for natural images, iWave is more efficient than the traditional wavelet transforms when processing natural images. This is confirmed by our experimental results, as we adopt iWave for image compression and compare it with JPEG-2000.

Compared with the recent works about CNN-based image compression [9]–[16], our iWave-based compression scheme has a profound mathematical background and much better interpretability. The lifting scheme has been mathematically proved to be equivalent to the normal wavelet transform, thus iWave upon the lifting scheme is indeed a wavelet-like transform. The difference between iWave and the traditional wavelet transforms is only the CNN-based prediction/update operations.[1] iWave also reserves several benefits of the lifting scheme, such as ensuring perfect reconstruction and supporting multiresolution analysis. Owing to the lifting structure, iWave is more interpretable than the deep networks that are trained as "black boxes" without fully understanding the network structure. It is worth noting that several recent works [20], [21] also adopt the lifting scheme to construct interpretable deep networks, but their objectives are to decrease the memory cost during network training and to reserve the information of input image as much as possible. In this paper, our objective is to present a new wavelet-based transform, and to investigate its usefulness for image compression.

We have made the following technical contributions that will be detailed in this paper:

[1]More detailed mathematical analysis of iWave is beyond the scope of this paper. Such analysis shall be interesting and inspiring, but seems not easy due to the lack of mathematical modeling of CNN.

- To the best of our knowledge, we are the first to present a CNN-based wavelet-like transform upon the update-first lifting scheme. iWave is more interpretable than the CNNs trained as "black boxes."
- For the purpose of image compression, we present an efficient algorithm to train iWave in an end-to-end fashion. We also design a method to properly scale the coefficients produced by iWave.
- We conduct an extensive set of experiments to verify the effectiveness of iWave in comparison with the traditional wavelet transforms. Especially, we demonstrate the exceptional benefit of iWave in dealing with specific kinds of irregular textures.

The remainder of this paper is organized as follows. Section II gives a brief review of related works. In Section III, we present the nonlinear wavelet transform iWave, and the framework to adopt iWave for the compression task. In Section IV, we show the performance of iWave transform and iWave-based image compression, and we also discuss different structures to implement the CNN-based wavelet transforms. Section V concludes this paper.

II. Related Work

In this section, we will give an overview of the related works at two aspects. The first is about the wavelet transform that is widely used as a powerful tool for multiresolution time-frequency analysis. The second is about deep learning-based image compression, which is an emerging topic of great interest in recent years.

A. *Wavelet Transform and Its Improved Variants*

Wavelet transform was highly praised for its comprehensive mathematical background and excellent multiresolution decomposition capabilities. It inherits and develops the idea of short-time Fourier transform and overcomes the disadvantage that the window size cannot be changed with frequency. Wavelet transform achieves both local and global representations by means of local support of bases and pyramid decomposition, and is more suitable for processing signal with point singularities than Fourier transform. However, traditional wavelet transform cannot capture the two-dimensional singularities in natural images efficiently due to two reasons: arbitrary directional features and locally variant features in natural images. Several variants have been proposed to deal with this problem, including non-adaptive and adaptive methods.

In the non-adaptive category, the ridgelet [3], [4] is first proposed to describe straight lines with arbitrary directions in images. The ridgelet transform has the abilities of direction recognition and selection by firstly performing Randon transform to convert lines into points in the Randon domain, and then employing one-dimensional wavelet transform. Curvelet [5] was proposed to deal with more general curve singularities, which is actually block-based ridgelet transform. However, due to the overlapping blocks in the partition stage, curvelet generates redundant coefficients, which is harmful for image compression. Contourlet [6] is a two-dimensional transform, which employs a structure that performs subband decomposition and then directional transform. Due to the used Laplacian pyramid in the decomposition stage, the contourlet transform also produces redundant coefficients.

In the adaptive category, Claypoole *et al.* proposed to use adaptive wavelet transform for the local properties of images [22]. They firstly proposed the update-first lifting scheme for more stable transform without using side information. Ding *et al.* proposed a framework for adaptive directional lifting (ADL) based wavelet transform for image compression [7]. The ADL combines one-dimensional wavelet transform with directional prediction, to utilize the directional information in images. By means of block partitioning and rate-distortion optimization and sending the side information (block-wise directions), ADL achieves better compression performance than JPEG-2000. Following ADL, Liu *et al.* proposed weighted adaptive lifting (WAL) based wavelet transform [8]. The WAL is designed for addressing several limitations of ADL, including the mismatch between prediction and update steps, the interpolation favoring only horizontal or vertical direction, and invariant interpolation filters for all images. WAL has further improved the performance upon ADL.

All the traditional wavelet transforms, including the improved variants, use hand-crafted wavelets. These wavelets were designed from the signal processing theory with certain assumption on the signal, but natural images are not as ideal as assumed by the theory. In this paper, we propose iWave as a wavelet-like transform that is trained by massive data of natural images.

B. Image Compression Via Deep Learning

Image compression based on deep learning methods has received great attention in the recent years. Currently, when evaluated by visual quality metrics such as SSIM, the compression performance of deep learning-based methods are better than even the state-of-the-art traditional method, like BPG.[2] There are two categories of networks used for image compression, i.e. recurrent neural network (RNN) and CNN.

Toderici *et al.* proposed the first image compression scheme based on RNN [9]. The network directly converts image to binary stream by using binary quantization at the last layer of the network. At the same time, variable rate can be achieved by iterate the RNN network. Lately this work was improved by training a RNN-based arithmetic encoder to reduce the redundancy between the binary codes [10]. More recently, Johnston *et al.* further improved the work by introducing SSIM weighted loss, hidden-state priming, and bit allocation algorithm [11]. One advantage of the above works is that the network can provide compression results at multiple bit rates.

The other series of works choose to optimize the network separately at different bit rates. Balle *et al.* proposed a fully convolutional deep network structure [12]. They introduced optimized generalized divisive normalization (GDN) into the network to exploit the correlation among pixels. Instead of binary quantization, they use multi-ary quantization and simulate quantization by adding noise during training process. This work was improved by introducing a locally variable spatial distribution in the coefficient domain in both encoder and decoder, which yields rate-distortion performance surpassing the published deep learning-based methods when evaluated by PSNR [13]. More recently, Minnen *et al.* introduced joint autoregressive and hierarchical priors, and further improved the compression performance: their scheme surpasses BPG when evaluated with RGB-PSNR [23]. There are similar works at the same time. Theis *et al.* proposed a fully convolutional neural network, but with different rate estimate and integer quantization strategy [14]. Rippel and Bourdev embedded the pyramid decomposition structure into the network [15]. Agustsson *et al.* proposed a soft-to-hard vector quantization method to train the network [16].

All of the above deep learning-based image compression works trained the neural networks as "black boxes." Contrast to them, we embed the CNNs into the lifting scheme, which makes our network more interpretable.

III. Proposed Method

We present iWave for image compression in this section. This consists of three parts. First, we introduce the structure of iWave as a new kind of wavelet-like transform. Secondly, the training method and loss function to obtain iWave are introduced. Third, we adopt iWave for image compression task.

A. iWave Transform

iWave implements wavelet-like transform by introducing CNN into the lifting scheme. The lifting scheme gives a faster and in-place implementation of wavelet compared to the first generation of wavelet which are defined as translates and dilates of one function, and moreover, ensures perfect reconstruction. We design the network structure from the requirements of the compression task, and use the modified update-first lifting scheme. We will start with a comparison of these two kinds of lifting structures, prediction-first and update-first schemes, followed by the introduction of iWave transform structure for two-dimensional images.

The prediction-first lifting scheme consists of three steps: split, prediction and update. Taking one-dimensional case as example, as Fig. 1(a) shows, at first we split a vector x into two parts, often even and odd parts, x_e and x_o. Then we predict x_o given x_e and compute the difference x_d as follows:

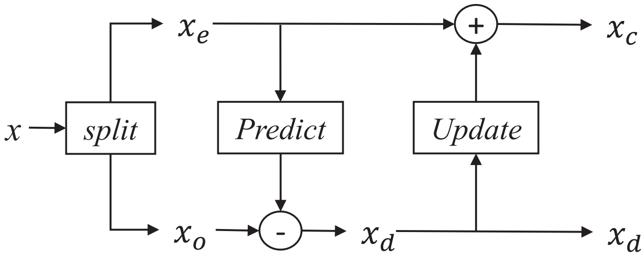
$$x_d = x_o - P(x_e). \tag{1}$$

Then we update x_e into x_c with x_d as follows:

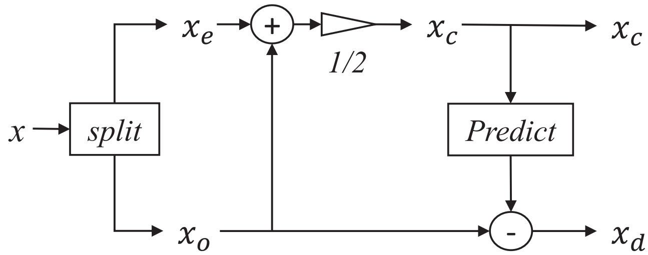
$$x_c = x_e + U(x_d) \tag{2}$$

where $P()$ and $U()$ denote prediction filter and update filter, respectively. In traditional wavelet transforms, the filters $P()$ and $U()$ are of simple linear forms. For example, in the CDF 5/3 wavelet (which is used in the JPEG-2000 standard), (1) can be written as $x_d[m] = x_o[m] - (p_a x_e[m-1] + p_b x_e[m])$, where $p_a = p_b = 0.5$. Other wavelet transforms have different forms of $P()$ and $U()$ [17].

[2][Online]. Available: https://bellard.org/bpg/



(a) Prediction-first lifting scheme



(b) Update-first lifting scheme

Fig. 1. Comparison of two lifting schemes for wavelet transform.

The decomposition results consist of two parts of coefficients, x_c and x_d. The first part x_c is called coarse coefficients, which contains most of the information of original data, and the second part x_d is called detail coefficients, which contains high frequency information.

The update-first lifting scheme is illustrated in Fig. 1(b). There are two main differences compared to the above prediction-first lifting scheme. First, the prediction and update blocks are exchanging their positions, and second, the update block is achieved by a simple mean filter. We formulate the process of update-first lifting given the two split parts x_e and x_o. Update:

$$x_c = (x_e + x_o)/2 \tag{3}$$

Predict:

$$x_d = x_o - P(x_c) \tag{4}$$

The update-first lifting scheme is first proposed as response to the problem of instability of the adaptive wavelet transform in the compression task [22]. In fact, the update-first scheme is worse than the prediction-first one in terms of energy compaction ability, since the mean filter is too simple to reduce the redundancy in the input signal. However, the update-first scheme is more robust to the quantization noise while being applied to image compression. Our iWave follows the update-first lifting scheme, as we empirically observed it is better (see results in Section IV-E). Essentially, we replace the prediction block in Fig. 1(b) with a trained CNN.

iWave can be easily modified from one-dimensional to two-dimensional by performing separate decompositions in the row and column directions, respectively, just like the traditional wavelet transforms. We give the entire iWave transform structure for two-dimensional images in Fig. 2. After the row-wise transform, we transpose the signal and then perform column-wise transform. We use the same set of parameters for row- and column-wise transforms. In traditional wavelet transforms, the filters are also the same for horizontal and vertical directions. It has the benefit of reducing the number of parameters. In addition, it can provide a regularization for the CNN parameters. The decomposition result of iWave of an image is four subbands, one coarse subband LL, and there detail subbands HL, LH, and HH. We emphasize that the separate two-dimensional iWave can still capture the features in images efficiency because of the two-dimensional filters used in CNN.

The key module of iWave is the CNN-based prediction block which is implemented by dense CNN [24] in this paper. The specific structure of dense CNN used is shown in Fig. 3. Since wavelet has local support of bases, it is related to the order of filters in traditional lifting scheme and receptive fields of CNN in iWave. A larger support leads to smoother coarse coefficients band, however, it is also more likely to cause failure to represent the edges of the image. The dense CNN actually have various receptive fields, which allows the most suitable support for natural images to be obtained after the training process. Another reason that prompts us to use dense CNN is that we could use fewer convolution kernels, reported as a contribution of dense CNN in [24], which contributes to a smaller model size. We use `tanh` function as activation as we found it makes the iWave more robust to quantization noise than other activation functions. Note that ReLU was used extensively, but ReLU has a singularity at the zero point and is observed not suitable in iWave.

The lifting scheme guarantees that we can always reconstruct the input perfectly. Let us consider the update-first lifting, i.e. (3) and (4). Given x_c and x_d, we have

$$x_o = x_d + P(x_c)$$

$$x_e = 2x_c - x_o$$

which is depicted in Fig. 4 for iWave. This fulfills the inverse transform. Similarly, we can derive the two-dimensional inverse transform of iWave, which is omitted herein. We name the forward and inverse transform of iWave as iWave-Fwd and iWave-Inv respectively. It is worth noting that iWave reserves the perfect reconstruction property of the lifting scheme, as long as the same $P()$ is applied in iWave-Fwd and iWave-Inv. The nonlinearity of $P()$ does not hinder the perfect reconstruction. The perfect reconstruction feature distinguishes iWave from all the previous learning-based image compression methods.

B. Training Structure and Loss Function

iWave can be trained with different targets for different tasks. In this paper, we are interested in the image compression task and we want to train iWave for efficient compression. For wavelet-based image compression schemes, e.g. JPEG-2000, there is sophisticated mechanism to compress the transform coefficients because abundant redundancy is within the coefficients. We do not change the coefficient coding mechanism and instead we focus on the transform itself. For this reason, we define an objective to train iWave with the compression task in mind but our

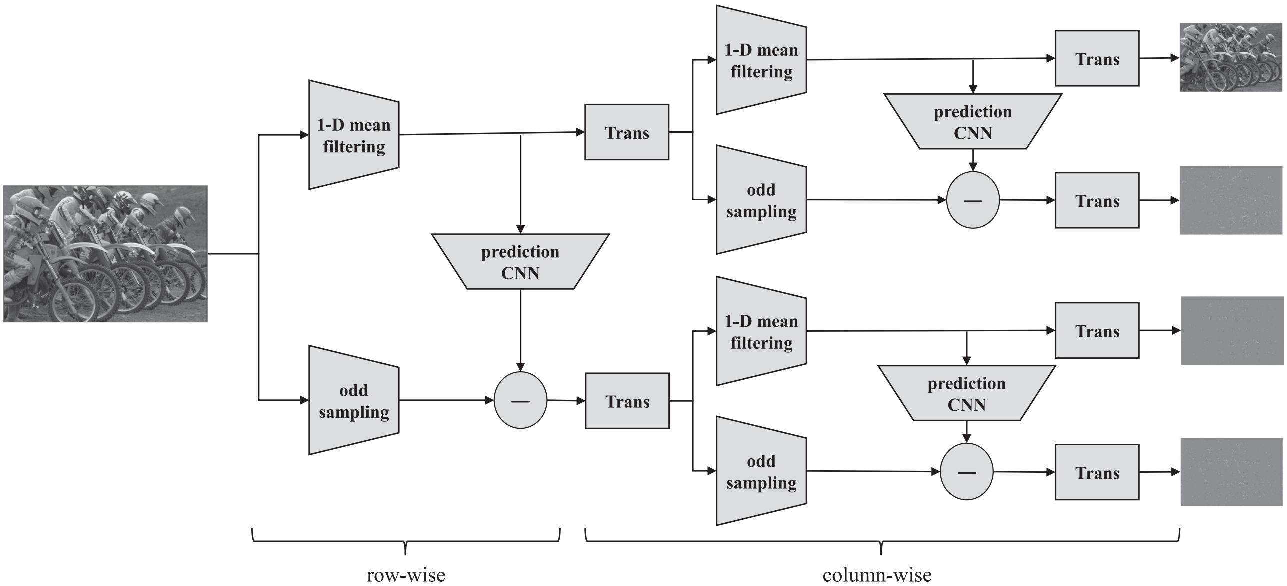


Fig. 2. iWave forward transform for two-dimensional images, where "1-D mean filtering" stands for $(x_e + x_o)/2$, "odd sampling" stands for extracting x_o, and "Trans" stands for transposing the signal. All the three prediction CNN blocks share the same set of parameters.

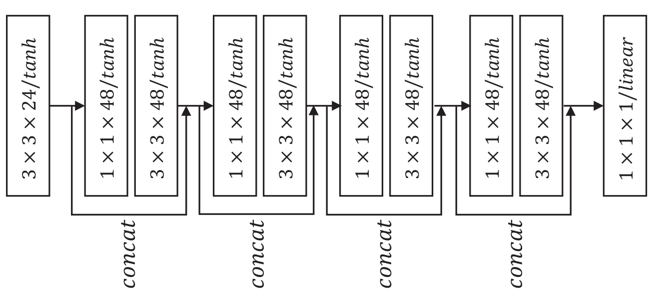


Fig. 3. Network structure of the prediction CNN. Shown numbers like "$3 \times 3 \times 24$" indicate the kernel size (3×3) and the number of kernels (24) in each layer. The tanh activation is used at every layer except for the last one, which has linear activation. The lines marked "concat" between layers represent dense connections.

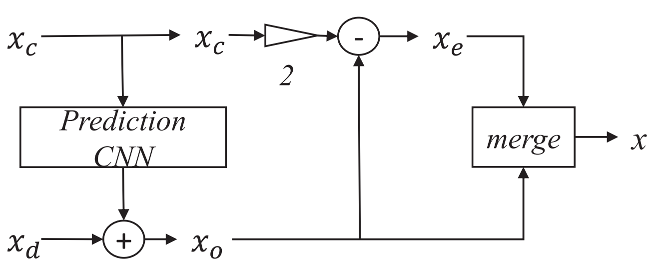


Fig. 4. iWave inverse transform along one dimension.

objective is not exactly equivalent to the rate-distortion cost. In short, our defined objective is the compactness of the transform coefficients. In this paper, a transform is said more compact, if it can represent a signal to the same quality with less coefficients, or it can represent to a higher quality with the same number of coefficients.

The training objective of compactness naturally leads to the distance between the input and output treated as loss function, which means that we should reconstruct the input with partial transform coefficients in the decoding part then minimize the distance. Then it is a question to define which part of coefficients should be preserved. Instead of choosing a fixed part of transform coefficients to preserve, we choose to progressively preserve more and more coefficients, which leads to our loss function having multiple parts, each part measuring the quality of reconstruction under a certain fraction of coefficients. We choose the progressive reconstruction because we hope the trained iWave can be applied to scalable image compression just like traditional wavelet transform does. More specifically, for the training we use a forward transform and multiple rounds of inverse transform, each round using partial coefficients to reconstruct. All of the reconstruction errors are summed up to form the loss function for training. In practice, we treat each subband as a unit, adding one more subband every time. We name this process of adding coefficients as *progressive selection*. Suppose we perform N levels of decomposition, then we have $3N + 1$ subbands, according to the above method, we can get the loss function as follows:

$$L_N = \sum_{i=1}^{3N} d(X, \hat{X}_i) \tag{5}$$

where $d()$ represents the difference between two images, X is the original image, and $\hat{X}_i$ is the reconstructed image when using the first i subbands to reconstruct. Note that it is not necessary to consider $d(X, \hat{X}_{3N+1})$, because the difference is always zero in theory (due to perfect reconstruction of the lifting scheme). In our experiments, we choose to use mean squared error (MSE) to measure the distance, for it is directly related to peak signal-to-noise ratio (PSNR).

The above described process naturally forms an end-to-end training strategy. Fig. 5 gives the illustration of this training

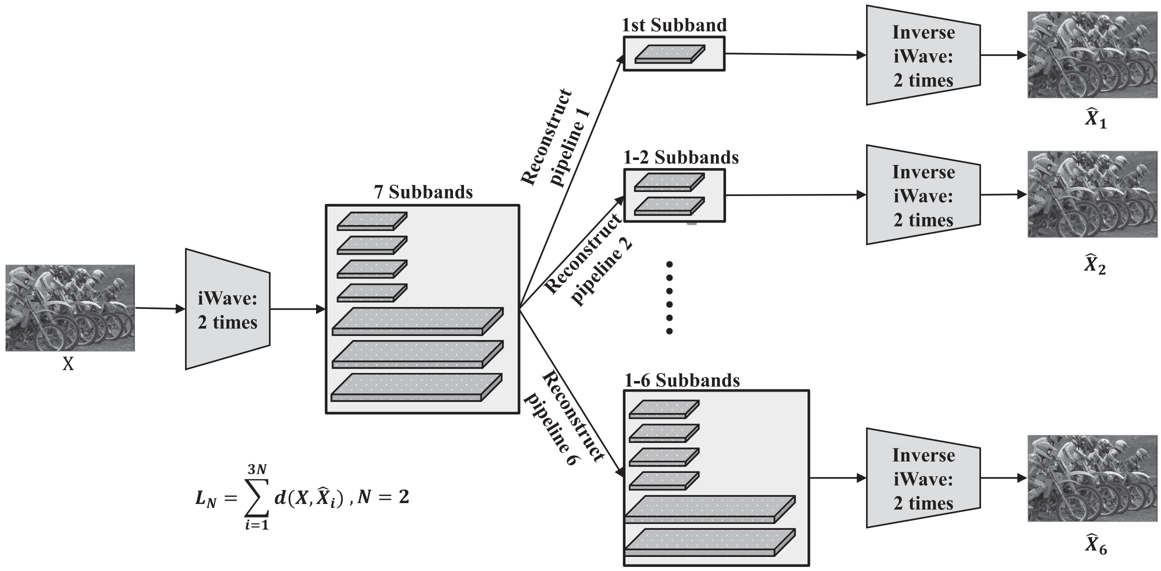


Fig. 5. End-to-end training strategy for iWave. Two-level decomposition ($N = 2$) is shown for example, while extension to multiple levels of decomposition is straightforward. Please refer to Fig. 6 for the indexes of the subbands.

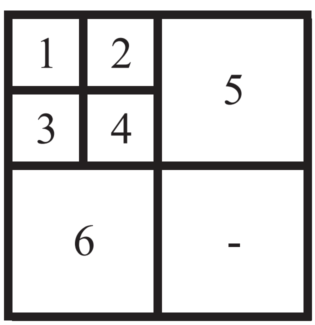


Fig. 6. The order of subbands (see Fig. 5). Two-level decomposition is shown for example, while extension to multiple levels of decomposition is straightforward.

structure, whose encoding part and decoding part correspond to forward and inverse transform of iWave respectively. Note that we depict two times of decomposition in Fig. 5, but one can use iWave iteratively for any times to get multiresolution representations.

We take 2 times of decomposition as example to depict the order of adding subbands in Fig. 6. The coefficients are organized according to the traditional wavelet transform. We regard the subbands obtained by high-level decomposition to be more important.

The training structure consists of multiple times of decomposition, i.e. multiple times of iWave transform, leading to a set of prediction CNN blocks employed insides. We choose to let all prediction CNN blocks share the same set of parameters among multiple times of decomposition during the training process. This makes iWave independent of resolution and once iWave is acquired, it can be used to compress image with any times of decomposition. Sharing parameters also makes the trained model unique, so that the model parameters can be easily stored and transferred.

C. Framework of iWave-Based Image Compression

iWave, the proposed new kind of nonlinear wavelet-like transform, has been acquired as mentioned above. iWave can be applied to image compression as long as the transform coefficients are quantized and entropy coded. One remaining issue for iWave as well as for conventional wavelet transform is to normalize the coefficients. Traditionally, the normalization is to equalize the importance of coefficients, resulting in a set of gain factors associated with every subband that are used to scale the coefficients. All the scaled coefficients can be uniformly quantized, which greatly facilitates the subsequent encoding process. This can be achieved by calculating the L_2-norm of the filters [25]. However for iWave, the transform is achieved by the lifting scheme and there is a nonlinear CNN inside the lifting scheme, the traditional way of calculating gain factors is not applicable. We propose a different method to achieve the gain factors of iWave. In essence, we want to balance the importance of different subbands, so that the quantization noise in one subband has an approximate same influence on the reconstruction quality.

Our method is described as Algorithm 1. We obtain a set of training images x, calculate the iWave coefficients by the forward transform, then add Gaussian noise with a specific variance to one subband, and perform inverse transform to calculate the reconstruction error L. We compare L with a predefined expected error L' to decide how to adjust the gain factor for the subband. If L is larger than L', we need to increase the gain factor, otherwise we need to decrease the gain factor. After several trials we shall have L approximately equal to L'. In this way, we achieve the gain factors for all subbands.

After proper scaling, we simply reuse the quantization and entropy coding modules in JPEG-2000 to build a complete iWave-based image compression scheme. This allows our compression method to provide scalable bitstreams. It is worth

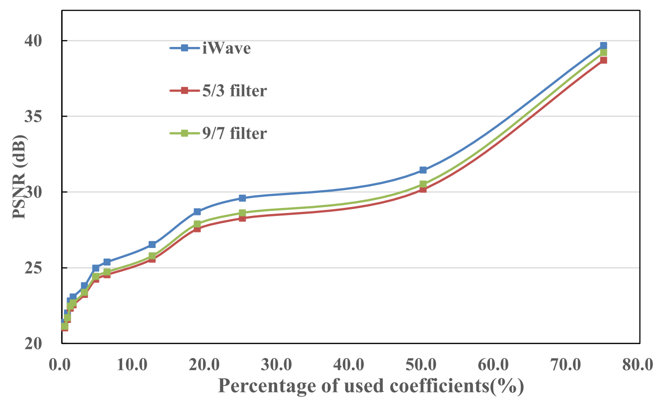


Fig. 7. Reconstruction PSNR with respect to the percentage of coefficients. Shown is the average result of the 24 images in the Kodak dataset (converted to grayscale), using the DIV2K dataset as training data. Four-level decomposition is used, and the coefficients are added progressively in the order (see Fig. 6).

Algorithm 1: $Gain(x, L')$ The Method to Obtain Gain Factors for Different Subbands.

Description of intermediate variables:

c denotes the coefficients after iWave forward transform, which are organized into a set of subbands, i.e. $c = \{c_1, c_2, \ldots, c_N\}$, where $N = 3D + 1$ after D levels of decomposition.

g_t is the corresponding gain factor for c_t.

ϵ and σ^2 are constants.

Input: Image set, x; expected reconstruction error, L';
Output: Gain factors i.e. g_t's;

```
for t = 1, 2, 3, ..., N do
  Initialize g_b = 0, g = 1, L = ∞;
  while (L/L' > 1 + ε) or (L'/L > 1 + ε) do
    Generate Gaussian noise n with zero mean and the
    specific variance σ^2.
    c = iWave-Fwd(x), c_t = (c_t × g + n)/g,
    x̃ = iWave-Inv(c), L = ||x − x̃||^2.
    if L/L' > 1 + ε then
      g_b = g;
      g = g × sqrt(L/L');
    else if L'/L > 1 + ε then
      g = (g_b + g)/2;
    end if
  end while
  g_t = g.
end for
```

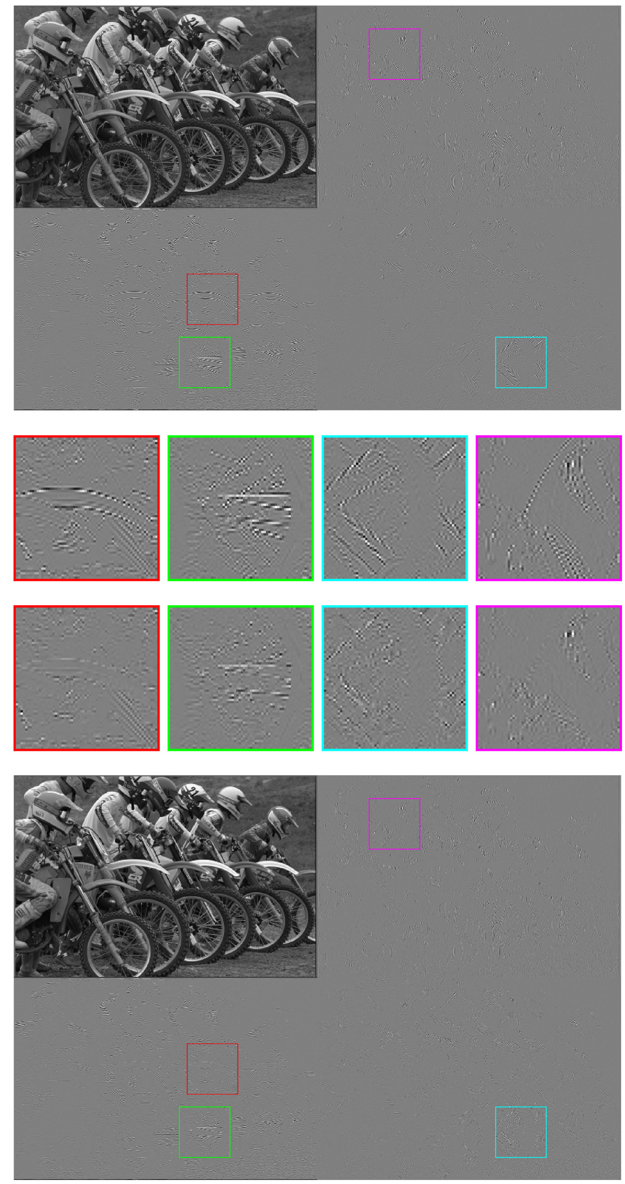


Fig. 8. Coefficients of one-level decomposition of kodim05. Top: CDF 9/7. Bottom: iWave trained with the DIV2K dataset.

noting that advanced quantization and/or entropy coding methods can be designed specifically for iWave, but they are beyond the scope of this paper.

IV. Experiments

We conduct two sets of experiments to evaluate the performance of the proposed method, in order to test the generality and speciality of iWave-based image compression, respectively. The purpose of generality experiment is to examine the performance of iWave-based compression method for general natural images. And speciality experiment is to identify the benefit of training a special iWave for different image content. An additional experiment to verify the effectiveness of the update-first lifting scheme is also performed.

A. Experimental Settings

In the generality experiment, we use 900 high-resolution images in the DIV2K dataset [26] for training, and test on the well-known Kodak dataset that consists of 24 images.[3] In the speciality experiment, we use four video sequences of the resolution 1920×1080, which will be shown later, corresponding to

[3][Online]. Available: http://www.r0k.us/graphics/kodak/

TABLE I
COMPARISON BETWEEN JPEG-2000 AND iWAVE-BASED IMAGE COMPRESSION (BITRATE IN bpp AND PSNR IN dB)

Kodak image	JPEG-2000 bitrates						iWave bitrates						BD-rate
	0.1	0.2	0.4	0.6	0.8	1.0	0.1	0.2	0.4	0.6	0.8	1.0	
01	23.00	24.66	26.89	28.57	30.16	31.52	23.12	24.75	26.92	28.51	30.12	31.45	−0.94%
02	31.22	32.97	35.32	37.16	38.77	39.98	31.29	33.13	35.43	37.35	38.91	40.09	−3.44%
03	31.53	33.98	37.65	40.46	42.59	44.40	32.24	34.89	38.33	40.89	43.02	44.52	−11.30%
04	30.34	32.26	34.97	36.95	38.57	39.89	30.60	32.53	35.13	37.18	38.74	40.00	−5.16%
05	21.73	23.67	26.36	28.46	30.19	31.93	22.04	24.32	27.22	29.15	31.25	32.87	−14.65%
06	25.20	27.04	29.73	32.09	33.88	35.54	25.22	27.04	29.66	31.97	33.76	35.37	1.22%
07	28.23	31.43	35.64	38.69	41.11	43.15	28.85	32.36	36.55	39.50	41.60	43.36	−11.67%
08	20.37	22.68	25.57	27.85	29.76	31.51	20.67	23.01	25.92	28.06	29.96	31.54	−5.80%
09	29.79	32.92	36.89	39.42	41.07	42.21	30.51	33.45	37.24	39.65	41.12	42.20	−6.90%
10	29.61	32.42	35.97	38.41	40.14	41.60	29.91	32.98	36.48	38.80	40.35	41.66	−8.03%
11	26.54	28.50	31.24	33.44	35.14	36.93	26.64	28.70	31.42	33.56	35.13	36.98	−3.11%
12	31.08	33.47	36.44	38.63	40.31	41.84	31.35	33.70	36.49	38.68	40.22	41.85	−2.54%
13	20.78	22.25	24.18	25.81	27.11	28.28	20.74	22.16	24.07	25.68	26.95	28.07	3.29%
14	25.11	26.97	29.40	31.30	33.01	34.26	25.25	27.24	29.53	31.52	33.12	34.38	−4.60%
15	30.14	32.38	35.50	37.75	39.60	41.07	30.86	32.87	35.73	38.01	39.72	41.19	−7.14%
16	28.91	30.94	33.87	35.96	37.92	39.55	28.98	30.86	33.74	35.84	37.76	39.37	1.91%
17	28.61	31.24	34.55	36.89	38.83	40.35	28.81	31.68	35.12	37.55	39.27	40.62	−8.84%
18	24.41	26.13	28.86	30.78	32.61	34.06	24.45	26.12	28.85	30.88	32.67	34.08	−0.48%
19	26.90	29.38	32.20	34.22	35.91	37.71	27.36	29.59	32.28	34.23	35.84	37.60	−2.76%
20	29.63	32.43	35.85	38.57	41.00	43.09	30.36	33.02	36.34	38.98	41.12	43.37	−8.08%
21	25.58	27.80	30.73	33.18	35.12	37.18	25.53	27.83	30.67	33.18	35.03	37.16	0.50%
22	27.61	29.48	31.98	33.89	35.42	37.08	27.69	29.61	32.08	33.90	35.40	37.11	−1.84%
23	33.36	36.83	40.42	42.57	43.92	44.87	34.00	37.33	40.68	42.70	43.89	44.79	−6.39%
24	23.88	25.78	28.64	30.63	32.60	34.26	23.88	25.68	28.57	30.74	32.77	34.39	0.25%
Average	25.75	27.76	30.42	32.44	34.12	35.63	25.93	27.96	30.58	32.56	34.23	35.68	−4.44%

four kinds of textures: large leaves, small leaves, grass and water ripples. We focus on grayscale image compression in this paper, so all the images in the training set and test set are converted to grayscale, unless otherwise noted.

We split the images into 128×128 patches for training. Three times of decomposition are embedded into the end-to-end training structure. The Adam algorithm [27] with default settings is adopted for training. The learning rate is empirically set to 0.0001, and we found that reducing learning rate does not lead to an increase of performance. The training is conducted using TensorFlow on an NVIDIA GTX1080Ti GPU.

We reuse part of the JPEG-2000 reference software Jasper[4] for experiments. During compression, we use four times of decomposition, which is a default setting in Jasper. Note that this is different from the training setting, and it can reveal that the resolution independence of iWave. All the default settings in Jasper are not changed.

As for Algorithm 1, we empirically predefine $L' = 1$, set ϵ as 0.02 and σ^2 as 1, for 8-bit images.

B. Generality Experiment

We will first show that as a trained wavelet-like transform, iWave is more suitable for natural images to provide more compact representations, which is directly related to our training objective. Then we will demonstrate the performance of the iWave-based image compression method at both objective and subjective aspects.

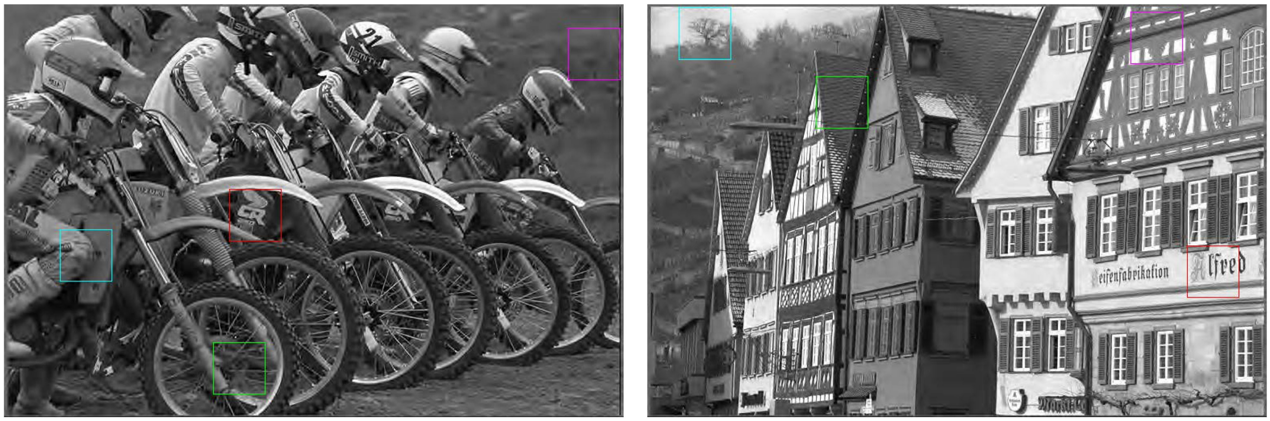
1) Energy Compaction: Consistent with our training goal, we test iWave's energy compaction ability and compare it with that of CDF 9/7 and 5/3, which are built-in JPEG-2000. Specifically, we use the progressive selection method to reconstruct an image with different fractions of transform coefficients, just like what we did in the training process. Fig. 7 shows the results of reconstruction PSNR. The curve gives the average results of 24 images in the Kodak dataset after four times of decomposition. As expected, in the whole range, with the same fraction of coefficients, iWave gives better reconstruction result than CDF 9/7 and 5/3, which means that iWave does provide better energy compaction.

We also visually inspect the result of one-level decomposition of kodim05 as shown in Fig. 8. For visualization purpose we normalize the coefficients of each subband into the range of $[0, 255]$. The difference between the two kinds of wavelet transforms is clearly visible in the high-frequency sub-bands, especially in the areas associated with the edges of the image. This indicates that iWave has advantages in dealing with the two-dimensional singularities in images, which is precisely the part that traditional wavelets have difficulty to process. This gives iWave great potential for image compression especially for edge-rich images.

2) Compression Performance: Table I summarizes the results on the Kodak dataset using iWave-based image compression and JPEG-2000 with CDF 9/7 at different rates including 0.1, 0.2, 0.4, 0.6, 0.8, and 1.0 bit-per-pixel (bpp). On average, iWave achieves 4.44% BD-rate reduction compared to JPEG-2000. The performance is much better for edge-rich images, such as kodim05 where the PSNR gain can be up to 1.06 dB at 0.8 bpp. This result is consistent with the decomposition results shown in the previous subsubsection.

In parallel with the results shown in Table I, we select kodim05 at 0.4 bpp and kodim08 at 0.6 bpp to visualize the reconstructed images, as shown in Fig. 9. It can be observed that iWave helps

[4][Online]. Available: http://www.ece.uvic.ca/~frodo/jasper/



(a) bitrate: 0.4 bpp PSNR: 26.36 dB (b) bitrate: 0.6 bpp PSNR: 27.85 dB

(c) bitrate: 0.4 bpp PSNR: 27.22 dB (d) bitrate: 0.6 bpp PSNR: 28.06 dB

Fig. 9. Visual quality comparison. Top: JPEG-2000. Bottom: iWave-based image compression.

reconstruct the edges better and the iWave-reconstructed images contain less ringing artifacts. Thus, in addition to objective quality improvement, iWave achieves subjective quality improvement.

There are several cases where iWave performs not satisfactorily, such as on kodim01, kodim06, and kodim13. These images contain a large area of water or extremely fine-granularity textures. To understand the reason, we need to go back to the loss function. As described in Section III-B, when training iWave, the loss function evaluates the reconstruction error of using partial coefficients, which indicates the ability of energy compaction. Although energy compaction correlates to compression performance positively, the former cannot fully decide the latter. Here, we calculate the energy compaction metric as well as the BD-rate for several Kodak images. The results are shown in Table II. The energy compaction is evaluated by the defined reconstruction error of using partial coefficients. Specifically, it is calculated by (5) with $N = 4$ because we use four-level decomposition in the experiments. From Table II, it can be observed that iWave always

TABLE II
ENERGY COMPACTION PERFORMANCE AND BD-RATE ON SEVERAL IMAGES IN THE KODAK DATASET

Kodak image	Rec. Error			BD-rate
	CDF 9/7	iWave	Diff	
01	4770.12	4451.14	−6.7%	**−0.94**%
06	3204.38	3071.76	−4.1%	**1.22**%
13	6815.54	6656.41	−2.3%	**3.29**%
05	5725.45	5025.31	−12.2%	**−14.65**%
07	2074.55	1807.70	−12.9%	**−11.67**%

achieves better energy compaction than the CDF 9/7 wavelet. It verifies that the training of iWave is effective. The reconstruction error difference varies across different images, as it depends on image content. Better energy compaction does not necessarily lead to better compression performance, for example comparing kodim05 and kodim07. To solve this problem, we need to use a new loss function that correlates better to the rate-distortion cost for image compression.

We compare the performance of iWave with that of BPG, which is the most efficient non learning-based image compression scheme. We also compare with the methods proposed in [13], [23], which represent the state-of-the-arts of learning-based image compression. Most of the learning-based methods report compression results for RGB images. To make a fair comparison, we also extend iWave for RGB image compression, where we convert RGB to YUV, and perform wavelet transform on the Y, U, V channels separately. Our implementation for RGB images is exactly the same to that in Jasper. The results are shown in Fig. 10. It can be observed that iWave performs consistently better than JPEG-2000, WebP, and JPEG. However, it is less efficient than BPG and the two advanced learning-based methods. Note that iWave reuses the entropy coding module in JPEG-2000, while more efficient entropy coding methods are used in BPG and [13], [23]. BPG also has post-processing filters that improve quality. On the other hand, note that iWave-based image compression has several distinctive advantages, such as providing scalable bitstreams, compared to BPG and the other learning-based compression methods.

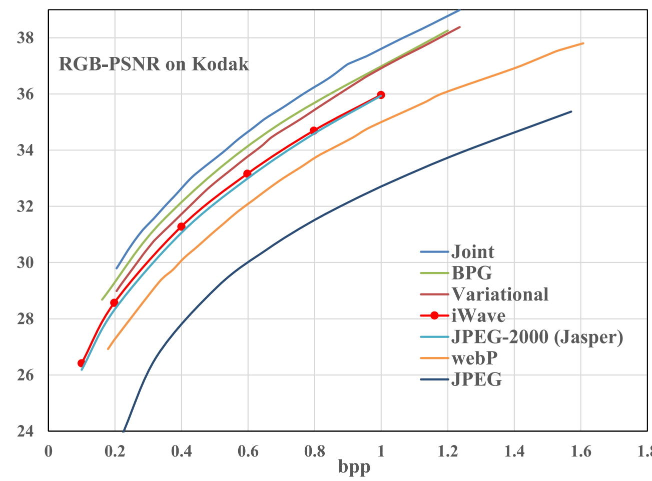


Fig. 10. Comparison of the average rate-distortion performance of different methods on the Kodak dataset. PSNR is calculated on RGB images. The method "Joint" refers to [23] and the method "Variational" refers to [13].

Fig. 11. Four kinds of textures used for the speciality experiment. Top left: large leaves; Top right: small leaves; Bottom left: grass; Bottom right: water ripples.

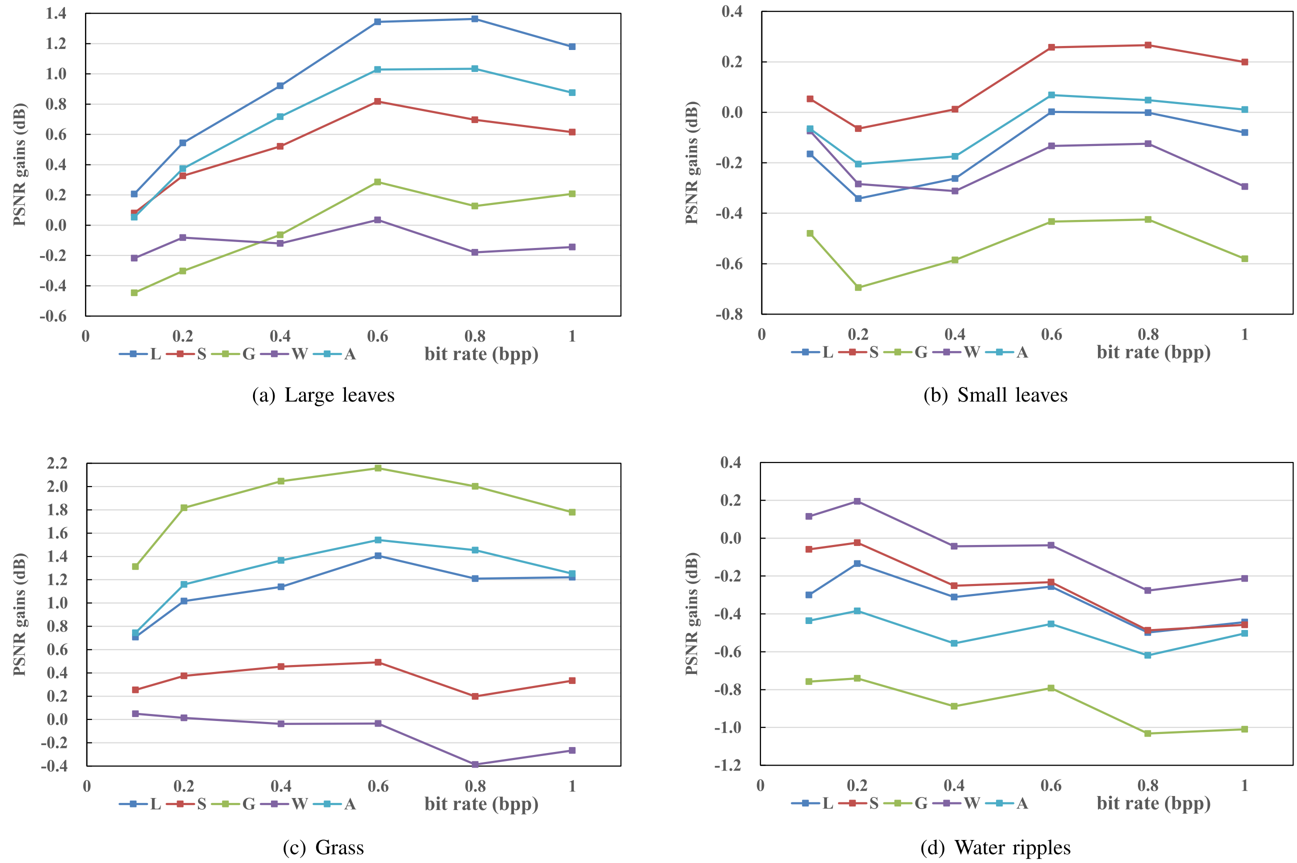
C. Speciality Experiment

One noticeable benefit of trained transform is its capability in identifying the characteristics of the training data and in performing very well for the same kind of data. We have shown this benefit in the previous experiment. We further perform speciality experiment to demonstrate the benefit when the training data and test data are more specialized. This can be the case in the image compression task, if we want to compress a number of images containing similar content. In practice, we can first encode the trained transform itself, and then encode images using the trained transform. It is worth noting that our iWave model uses shared parameters and thus the cost of model coding is small.

In this experiment, we use four video sequences of different kinds of textures, as shown in Fig. 11, including large leaves, small leaves, grass, and water ripples. For each video sequence, we select a portion of frames for training and use the remaining frames for testing. We take cross tests, i.e. using the model trained with one sequence to compress for another sequence. We also test the general model, i.e. the one trained with the DIV2K dataset, as mentioned before. Fig. 12 shows the performance of these five models on each kind of specific textures. And the BD-rate reduction results using the matching model are summarized in Table IV. First, it can be observed from Fig. 12 that the specialized model is always better than the general model for compressing special kind of textures. For the grass category, the specially trained iWave achieves up to 27.25% BD-rate reduction than JPEG-2000. Second, the general model performs very well for different kinds of textures: it is the second best for Large leaves, Small leaves, and Grass, which implies that a large-scale dataset with diverse content (e.g. DIV2K) can capture most of the characteristics in natural images. Third, the performance of iWave on Small leaves and Water ripples is still not satisfactory even using the specialized data to train. Such textures seem to be hard cases for iWave, which is also verified in the generality experiment (failure cases of kodim01, kodim06, kodim13).

D. Computational Complexity

One known drawback of CNN-based methods is the high computational complexity. In this paper, we do not optimize our implementation for computational speed. And we compare the computational time of iWave-based image compression method with that of JPEG-2000, as shown in Table III. Since we reuse the quantization and entropy coding part of JPEG-2000, we record the time of transform and entropy coding separately. It can be observed from Table III that the computational time depends on



(a) Large leaves

(b) Small leaves

(c) Grass

(d) Water ripples

Fig. 12. Results of the speciality experiment, using different combinations of training data and test images. Each subfigure shows results (PSNR gain is calculated over JPEG-2000) of a specific kind of *test* images. Lines with different colors indicate different *training* data: L (large leaves), S (small leaves), G (grass), W (water ripples), and A (the DIV2K dataset).

TABLE III
COMPUTATIONAL TIME OF iWAVE-BASED IMAGE COMPRESSION COMPARED TO JPEG-2000 (THE OVERALL TIME OF JPEG-2000 IS USED AS ANCHOR: 100%)

	Encoder			Decoder		
	Transform	Entropy encoding	Overall	Inverse transform	Entropy decoding	Overall
JPEG-2000	25.47 ms	109.63 ms	100%	21.79 ms	28.19 ms	100%
iWave (CPU)	1132.63 ms		919.53%	1203.27 ms		2463.61%
iWave (GPU)	28.45 ms		102.20%	29.65 ms		115.72%

TABLE IV
BD-RATE RESULTS OF THE SPECIALITY EXPERIMENT OF iWAVE VERSUS JPEG-2000

Texture	Large leaves	Small leaves	Grass	Water ripples
BD-rate	−12.97%	−1.58%	−27.25%	−0.77%

the adopted hardware. When using CPU for iWave, the computational time is much longer than that of JPEG-2000; but if we switch to GPU, the time is much shorter.

E. Verification of Different Lifting Schemes

iWave adopts the update-first lifting scheme as inspired by [22], but the prediction-first scheme is much more adopted in most of the previous works. We here compare the two different lifting schemes. We implement a variant of iWave, i.e. prediction-first iWave, by replacing the prediction and update blocks in Fig. 1(a) with trained CNNs. The training strategy to obtain the prediction-first iWave is the same as that for the update-first iWave.

We compare these two CNN-based wavelet-like transforms at two aspects: energy compaction capability and compression performance, and the results are shown in Fig. 13. It can be seen that the prediction-first iWave gives out more compact representations but worse compression performance. This observation is consistent with that reported in [22]. The reasons are twofold. On the one hand, the prediction-first scheme has two adjustable blocks (P and U in Fig. 1(a)), but the update-first scheme has only one (P in Fig. 1(b)). The former has more parameters to learn and probably can achieve better energy compaction. On the other hand, the training of iWave does not consider quantization noise in compression. Since the prediction-first iWave has more nonlinear units, it is more sensitive to noise. To demonstrate this, we conduct a simple experiment. We use prediction- and

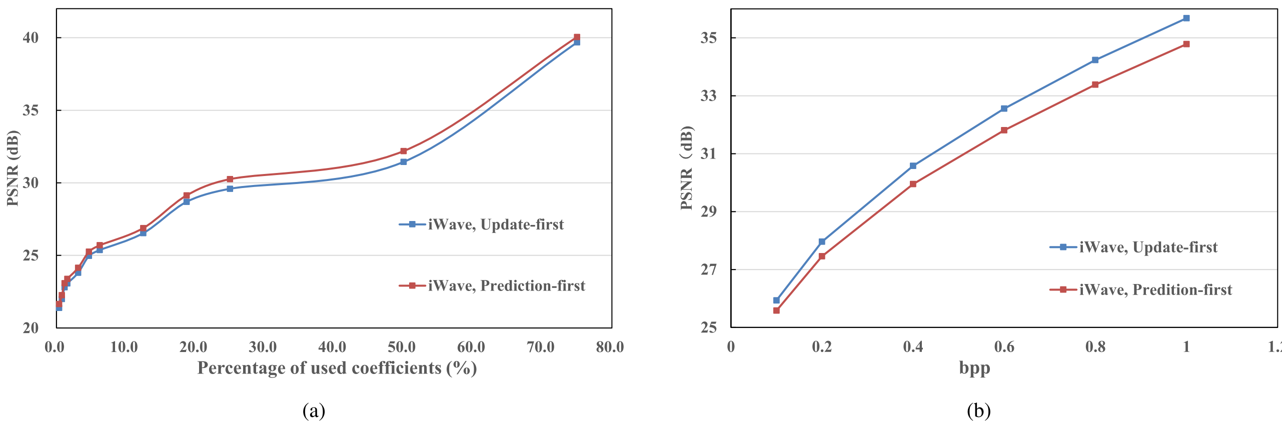


Fig. 13. Comparison between iWave (update-first) and its variant (prediction-first). Shown is the average result of the 24 images of the Kodak dataset, using the DIV2K dataset as training data. (a) Reconstruction PSNR with respect to the percentage of coefficients, where four-level decomposition is used. (b) Reconstruction PSNR with respect to bit-rate.

TABLE V
RECONSTRUCTION QUALITY (PSNR IN dB) OF iWAVE (UPDATE-FIRST) AND ITS VARIANT (PREDICTION-FIRST), WHEN ADDING GAUSSIAN NOISE TO THE TRANSFORM COEFFICIENTS

Kodak image	Update-first	Prediction-first
01	36.80	35.90
02	37.30	37.15
03	37.10	37.03
04	37.17	36.93
05	36.15	35.08
06	36.83	36.33
07	36.75	36.61
08	36.06	35.27
09	36.94	36.89
10	37.01	36.88
11	36.89	36.30
12	37.17	36.94
13	36.77	35.60
14	36.87	36.06
15	36.90	36.71
16	37.35	36.83
17	36.95	36.63
18	36.89	35.96
19	36.88	36.57
20	36.81	37.01
21	36.94	36.36
22	37.22	36.56
23	37.40	37.42
24	36.73	35.87
Average	36.91	36.45

update-first schemes to transform the 24 images in the Kodak dataset, add some noise (Gaussian with zero mean and variance equal to 1.0) into the coefficients, and then inversely transform. As shown in Table V, the reconstruction PSNR values of update- and prediction-first schemes are 36.91 dB and 36.45 dB, respectively. That says, noise at the same level will deteriorate reconstruction quality more severely on the prediction-first scheme. Therefore, from the compression point of view, update-first iWave is better than the prediction-first iWave. It is worth noting that, the phenomenon is fundamentally attributed to the loss function. Because the loss function (5) does not determine the compression performance, the prediction-first scheme has a smaller loss but a worse compression efficiency. If we change to another loss function, the case may be different.

V. CONCLUSION

We have proposed iWave, a new kind of nonlinear wavelet-like transform based on the update-first lifting scheme and trained CNN. Experimental results show that iWave can represent the two-dimensional singularities in images better than traditional wavelets, leading to a more compact representation. When we adopt iWave for image compression, the iWave-based method outperforms the well-known wavelet-based method JPEG-2000. We found that the proposed method performs much better on edge-rich images. In the speciality experiment, we have shown that further improvement could be obtained when training and using iWave for specific content. iWave maintains several advantages of wavelets, such as scalable coding, perfect reconstruction, and is more interpretable than the deep networks trained as “black boxes.” We believe that beyond image compression, iWave may be beneficial for other image processing tasks, such as denoising and enhancement.

Our work has several limitations. First, the adopted loss function for training iWave can be improved. For image compression, it is promising to use rate-distortion cost, or its estimates, to train iWave. For other tasks, advanced loss functions are to be investigated in the future. Second, as a common practice in CNN-based methods, we use floating-point parameters and floating-point operations in the current implementation of iWave. It has the potential defect of platform dependence. We plan to investigate the parameter quantization for iWave in our future work.

REFERENCES

[1] D. Taubman and M. Marcellin, *JPEG2000 Image Compression Fundamentals, Standards and Practice*. Berlin, Germany: Springer, 2012.

[2] W. B. Pennebaker and J. L. Mitchell, *JPEG: Still Image Data Compression Standard*. Berlin, Germany: Springer, 1992.

[3] E. J. Candes, “Monoscale ridgelets for the representation of images with edges,” Dept. Statist., Stanford Univ., Tech. Rep., 1999.

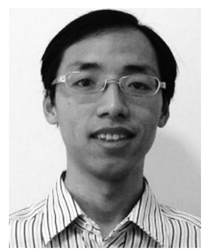
[4] M. N. Do and M. Vetterli, “The finite ridgelet transform for image representation,” *IEEE Trans. Image Process.*, vol. 12, no. 1, pp. 16–28, Jan. 2003.

[5] E. J. Candes and D. L. Donoho, "Curvelets, multiresolution representation, and scaling laws," in *Proc. SPIE*, vol. 4119. Int. Soc. Opt. Photon., 2000, pp. 1–13.
[6] M. N. Do and M. Vetterli, "The contourlet transform: An efficient directional multiresolution image representation," *IEEE Trans. Image Process.*, vol. 14, no. 12, pp. 2091–2106, Dec. 2005.
[7] W. Ding, F. Wu, X. Wu, S. Li, and H. Li, "Adaptive directional lifting-based wavelet transform for image coding," *IEEE Trans. Image Process.*, vol. 16, no. 2, pp. 416–427, Feb. 2007.
[8] Y. Liu and K. N. Ngan, "Weighted adaptive lifting-based wavelet transform for image coding," *IEEE Trans. Image Process.*, vol. 17, no. 4, pp. 500–511, Apr. 2008.
[9] G. Toderici *et al.*, "Variable rate image compression with recurrent neural networks," 2015, *arXiv:1511.06085*.
[10] G. Toderici *et al.*, "Full resolution image compression with recurrent neural networks," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 5435–5443.
[11] N. Johnston *et al.*, "Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 4385–4393.
[12] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," 2016, *arXiv:1611.01704*.
[13] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," 2018, *arXiv:1802.01436*.
[14] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," 2017, *arXiv:1703.00395*.
[15] O. Rippel and L. Bourdev, "Real-time adaptive image compression," in *Proc. 34th Int. Conf. Machine Learning*, 2017, pp. 2922–2930.
[16] E. Agustsson *et al.*, "Soft-to-hard vector quantization for end-to-end learning compressible representations," in *Adv. Neural Inf. Process. Syst.*, 2017, pp. 1141–1151.
[17] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," *J. Fourier Anal. Appl.*, vol. 4, no. 3, pp. 247–269, 1998.
[18] W. Sweldens, "The lifting scheme: A custom-design construction of biorthogonal wavelets," *Appl. Comput. Harmon. Anal.*, vol. 3, no. 2, pp. 186–200, 1996.
[19] W. Sweldens, "The lifting scheme: A construction of second generation wavelets," *SIAM J. Math. Anal.*, vol. 29, no. 2, pp. 511–546, 1998.
[20] A. N. Gomez, M. Ren, R. Urtasun, and R. B. Grosse, "The reversible residual network: Backpropagation without storing activations," in *Adv. Neural Inf. Process. Syst.*, 2017, pp. 2214–2224.
[21] J.-H. Jacobsen, A. Smeulders, and E. Oyallon, "i-RevNet: Deep invertible networks," 2018, *arXiv:1802.07088*.
[22] R. L. Claypoole, G. M. Davis, W. Sweldens, and R. G. Baraniuk, "Nonlinear wavelet transforms for image coding via lifting," *IEEE Trans. Image Process.*, vol. 12, no. 12, pp. 1449–1459, Dec. 2003.
[23] D. Minnen, J. Ballé, and G. D. Toderici, "Joint autoregressive and hierarchical priors for learned image compression," in *Adv. Neural Inf. Process. Syst.*, 2018, pp. 10 771–10 780.
[24] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 4700–4708.
[25] M. Rabbani and R. Joshi, "An overview of the JPEG 2000 still image compression standard," *Signal Process., Image Commun.*, vol. 17, no. 1, pp. 3–48, 2002.
[26] E. Agustsson and R. Timofte, "NTIRE 2017 challenge on single image super-resolution: Dataset and study," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit. Workshops*, 2017, pp. 126–135.
[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

Haichuan Ma received the B.S. degree in electrical engineering from Xidian University, Xi'an, China, in 2017. He is currently working toward the Ph.D. degree with the Department of Electronic Engineering and Information Science with the University of Science and Technology of China, Hefei, China. His research interests include image/video coding, signal processing, and machine learning.

Dong Liu (M'13–SM'19) received the B.S. and Ph.D. degrees in electrical engineering from the University of Science and Technology of China (USTC), Hefei, China, in 2004 and 2009, respectively.

He was a Member of Research Staff with Nokia Research Center, Beijing, China, from 2009 to 2012. He joined USTC as an Associate Professor in 2012. His research interests include image and video coding, multimedia signal processing, and multimedia data mining. He has authored or coauthored more than 100 papers in international journals and conferences. He has 16 granted patents. He has one technical proposal adopted by AVS. He was the recipient of the 2009 IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY Best Paper Award and the VCIP 2016 Best 10% Paper Award. He and his team were winners of several technical challenges held in ICCV 2019, ACM MM 2018, ECCV 2018, CVPR 2018, and ICME 2016. He is a Senior Member of CCF and CSIG, and an elected member of MSA-TC of IEEE CAS Society. He served as a Registration Co-Chair for ICME 2019 and a Symposium Co-Chair for WCSP 2014.



Ruiqin Xiong (M'08–SM'17) received the B.S. degree in computer science from the University of Science and Technology of China, Hefei, China, in 2001, and the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2007.

He was a Research Intern with Microsoft Research Asia from 2002 to 2007, and a Senior Research Associate with the University of New South Wales, Australia, from 2007 to 2009. He joined the Institute of Digital Media, School of Electronic Engineering and Computer Science, Peking University, Beijing, China, in 2010, where he is currently a Professor. He has authored or coauthored more than 120 technical papers in referred international journals and conferences. His research interests include statistical image modeling, deep learning, image and video processing, compression and communications. He is (co-)recipient of Best Student Paper Award at VCIP 2005, Best Paper Award at VCIP 2011, Best Student Paper Award at VCIP 2017, and Best Paper Award at CVPR 2019 Workshop.

Feng Wu (M'99–SM'06–F'13) received the B.S. degree in electrical engineering from Xidian University, Xi'an, China, in 1992, and the M.S. and Ph.D. degrees in computer science from the Harbin Institute of Technology, Harbin, China, in 1996 and 1999, respectively.

He is currently a Professor and the Assistant to the President with the University of Science and Technology of China, Hefei, China. Previously, he was a Principle Researcher and Research Manager with Microsoft Research Asia, Beijing, China. His research interests include various aspects of video technology and artificial intelligence. He has authored or coauthored more than 120 journal papers (including several dozen IEEE TRANSACTIONS papers) and top conference papers on MOBICOM, SIGIR, CVPR, and ACM MM. He has 80 granted U.S. patents. His 15 techniques have been adopted into international video coding standards. He serves or had served as the Deputy Editor-in-Chief for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY and as an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING and IEEE TRANSACTIONS ON MULTIMEDIA. He also serves as General Chair of ICME 2019, TPC Chair of MMSP 2011, VCIP 2010, and PCM 2009. He was the recipient of the best paper awards in IEEE TCSVT 2009, VCIP 2016, PCM 2008, and VCIP 2007, and Best Associate Editor Award of IEEE TRANSACTIONS ON IMAGE PROCESSING in 2018.