Tree-Structured Data Clustering-Driven Neural Network for Intra Prediction in Video Coding

Hengyu Man, Xiaopeng Fan, Senior Member, IEEE, Ruiqin Xiong, Senior Member, IEEE, and Debin Zhao, Member, IEEE

Abstract—Intra prediction is a crucial part of video compression, which utilizes local information in images to eliminate spatial redundancy. As the state-of-the-art video coding standard, Versatile Video Coding (H.266/VVC) employs multiple directional prediction modes in intra prediction to find the texture trend of local areas. Then the prediction is made based on reference samples in the selected direction. Recently, neural network-based intra prediction has achieved great success. Deep network models are trained and applied to assist the HEVC and VVC intra modes. In this paper, we propose a novel tree-structured data clustering-driven neural network (dubbed TreeNet) for intra prediction, which builds the networks and clusters the training data in a tree-structured manner. Specifically, in each network split and training process of TreeNet, every parent network on a leaf node is split into two child networks by adding or subtracting Gaussian random noise. Then data clustering-driven training is applied to train the two derived child networks using the clustered training data of their parent. On the one hand, the networks at the same level in TreeNet are trained with non-overlapping clustered datasets, and thus they can learn different prediction abilities. On the other hand, the networks at different levels are trained with hierarchically clustered datasets, and thus they will have different generalization abilities. TreeNet is integrated into VVC to assist or replace intra prediction modes to test its performance. In addition, a fast termination strategy is proposed to accelerate the search of TreeNet. The experimental results demonstrate that when TreeNet is used to assist the VVC Intra modes, TreeNet with depth = 3 can bring an average of 3.78% bitrate saving (up to 8.12%) over VTM-17.0. If TreeNet with the same depth replaces all VVC intra modes, an average of 1.59% bitrate saving can be reached.

Index Terms—Intra prediction, neural networks, network split, data clustering, tree structure.

I. INTRODUCTION

As the state-of-the-art video coding standard, H.266/VVC [1] developed by the Joint Video Expert Team (JVET) could achieve approximately 50% bitrate reduction for equivalent perceptual quality compared with its predecessor H.265/HEVC [2], [3] with the increased computational complexity as a trade-off [4]. Significantly, the bitrate saving provided by intra coding of VVC is 25.1% on average, up to 29.3% over HEVC [5]. The improvement is mainly achieved by the more flexible block partition, increased angular prediction modes, additional advanced prediction techniques, and the newly adopted adaptive loop filter (ALF) [6].

As a crucial part of intra prediction, the granularity of angular prediction modes can significantly influence the intra prediction performance. In HEVC, the number of angular prediction modes is 33. By adding a new mode between every two neighbouring angular prediction modes of HEVC, the angular prediction modes in VVC are increased to 65 for square blocks [5]. The expanded angular prediction modes enable VVC to describe the directional patterns in local areas more precisely, thus giving better prediction results.

Although more angular prediction modes can make intra prediction more accurate, predicting blocks with complex patterns, especially blocks without apparent directional features, is still challenging. This is because the angular prediction modes in VVC are all designed manually and almost equidistantly distributed within a particular range. Moreover, each mode can only access a few reference pixels in a specific direction, and only simple linear interpolation is performed for prediction.

Recently, neural network-based video coding has been rapidly developed. For the hybrid video coding framework, deep tools were integrated into every single component [7], for example, intra prediction [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], inter prediction [21], [22], in-loop filtering [23], [24], and entropy coding [25], [26]. For inter prediction, a fractional-pixel reference generation convolutional neural network (CNN) was proposed in [21] for uni-directional and bi-directional motion compensation. To refine the inter prediction result, a novel residue estimation network was proposed in [22] to estimate the residue between the current block and its inter-predicted block using their available spatial neighbours. The estimated residue and the prediction were fed into a deep refinement network to generate the refined prediction. For in-loop filtering, neural networks either acted as additional tools to enhance the filtering performance [23] or replaced the deblocking filter and the sample adaptive offset [24]. Neural networks could also be adopted as entropy coding tools to estimate the probabilities of predefined samples in the selected direction. Recently, neural network-based intra prediction has achieved great success. Deep network models are trained and applied to assist the HEVC and VVC intra modes. In this paper, we propose a novel tree-structured data clustering-driven neural network (dubbed TreeNet) for intra prediction, which builds the networks and clusters the training data in a tree-structured manner. Specifically, in each network split and training process of TreeNet, every parent network on a leaf node is split into two child networks by adding or subtracting Gaussian random noise. Then data clustering-driven training is applied to train the two derived child networks using the clustered training data of their parent. On the one hand, the networks at the same level in TreeNet are trained with non-overlapping clustered datasets, and thus they can learn different prediction abilities. On the other hand, the networks at different levels are trained with hierarchically clustered datasets, and thus they will have different generalization abilities. TreeNet is integrated into VVC to assist or replace intra prediction modes to test its performance. In addition, a fast termination strategy is proposed to accelerate the search of TreeNet. The experimental results demonstrate that when TreeNet is used to assist the VVC Intra modes, TreeNet with depth = 3 can bring an average of 3.78% bitrate saving (up to 8.12%) over VTM-17.0. If TreeNet with the same depth replaces all VVC intra modes, an average of 1.59% bitrate saving can be reached.

Index Terms—Intra prediction, neural networks, network split, data clustering, tree structure.

I. INTRODUCTION

As THE state-of-the-art video coding standard, H.266/VVC [1] developed by the Joint Video Expert Team

Manuscript received 22 December 2022; revised 6 May 2023; accepted 8 June 2023. Date of publication 19 June 2023; date of current version 23 June 2023. This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFF0900500 and in part by the National Natural Science Foundation of China (NSFC) under Grant 62272128 and Grant U22B2035. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Marta Mrak. (Corresponding author: Xiaopeng Fan.)

Hengyu Man, Xiaopeng Fan, and Debin Zhao are with the Department of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China, and also with the Peng Cheng Laboratory, Shenzhen 518055, China (e-mail: hengyu.man@hit.edu.cn; dbzhao@hit.edu.cn).

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

Digital Object Identifier 10.1109/TIP.2023.3286256

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
syntax elements, e.g., the quantized coefficient [25] and the transform index [26].

In the existing neural network-based intra predictions, the networks either directly generated the prediction pixels [8], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20] or enhanced the prediction quality [9], [10] using available reference samples. In [8], block-context pairs were separated into two clusters according to a directional context-block relationship to train two neural networks individually. In [11], [12], and [13], more network modes (up to 35) were designed for blocks of different sizes, and the network structures were further simplified. Beyond increasing the number of network modes, intra predictions via more complex network structures, such as CNN [15], the generative adversarial network (GAN) [18], and the conditional autoencoder [19], have also been studied.

Inspired by the progressively refined angular prediction modes in intra prediction, a tree-structured data clustering-driven neural network (TreeNet) is proposed for intra prediction in this paper. TreeNet builds the networks and clusters the training data in a tree-structured manner, which differs from the parallel networks in [8], [11], [12], and [13]. Specifically, in each network split and training process of TreeNet, every parent network on a leaf node is split into two child networks by adding or subtracting Gaussian random noise. Then a K-means-like [27] data clustering-driven training [23] is applied to train the two derived child networks using the clustered training data of their parent. On the one hand, the networks at the same level in TreeNet are trained with non-overlapping clustered datasets, and thus they can learn different prediction abilities. On the other hand, the networks at different levels are trained with hierarchically clustered datasets, and thus they will have different generalization abilities.

The main contributions of this work can be summarized as follows:

1) A tree-structured data clustering-driven neural network (dubbed TreeNet) is proposed for intra prediction, which builds the networks and clusters the training data in a tree-structured manner. TreeNet is integrated into VVC to assist or replace the intra prediction modes, achieving 3.78% and 1.59% bitrate saving over VTM-17.0 on average, respectively. As far as we know, this is the first work replacing all intra modes with neural networks.

2) A network split strategy is designed to split each parent network on a leaf node into two child networks by adding or subtracting Gaussian random noise. Data clustering-driven training is applied to train the two derived child networks using the clustered training data of their parent.

3) A fast termination strategy is proposed to accelerate the search of TreeNet by exploring the correlation between child networks and their parent network.

The rest of the paper is organized as follows: Section II briefly overviews related works, including intra prediction in VVC and neural network-based intra predictions. The proposed TreeNet is detailed in Section III. In Section IV, the experimental results and more analyses are presented. Finally, the conclusion and future works are provided in Section V.

Fig. 1. The directional prediction modes in HEVC and VVC. The solid lines represent the modes that exist in both HEVC and VVC. Modes represented by dotted lines only exist in VVC. The dotted lines in the shadow represent the wide angular modes in VVC. Mode 0 is planar mode, and mode 1 is DC mode.

II. RELATED WORKS

In this section, the intra prediction in VVC is first presented. Then the related neural network-based intra predictions are reviewed.

A. Intra Prediction in VVC

VVC adopts a block-based hybrid video compression framework. In VVC, the basic compression unit is called the coding tree unit (CTU). Firstly, each picture is divided into non-overlapping square CTUs. The CTU size can be set up to 128 × 128. Then, each CTU can be further subdivided into smaller coding units (CUs) along the coding tree structure. Specifically, each CTU is treated as one CU or split into multiple CUs by a recursive quaternary tree partition (QT), followed by a recursive multi-type tree partition (MTT). The MTT partition includes a binary tree partition (BT) and a ternary tree partition (TT). Therefore, the blocks in VVC intra prediction can be non-square. The more flexible block partition in VVC improves the coding efficiency significantly.

In HEVC, there are 35 directional prediction modes, as shown in Fig. 1, including planar mode, DC mode, and 33 angular prediction modes. Planar mode is designed for areas with gradually changing contents. DC mode aims to predict smooth textures, and the predicted sample values are populated with a constant value representing the average of surrounding reference samples. Angular prediction modes are designed to model structures with directional edges. To better adapt to the diverse image contents, the number of angular prediction modes is increased to 65 in VVC by adding a new mode between every two neighbouring angular prediction modes of HEVC, as shown in Fig. 1. Furthermore, aiming at the problem of asymmetric distribution of direction prediction in non-square blocks, VVC adopts wide-angular intra prediction (WAIP) [28], in which 14 angles using prediction from the shorter side are replaced by more extreme angles using prediction from the longer side. After adding directional modes
with wider angles, the number of directions supported in VVC intra prediction is increased to 93.

In addition to the increased directional prediction modes, multiple advanced intra prediction techniques are designed for VVC to make better use of surrounding reference samples, such as position-dependent prediction combination (PDPC) [29], multiple reference lines (MRL) [30], cross-component linear model (CCLM) [31], and matrix-based intra prediction (MIP) [13]. In PDPC, filtering with spatially varying weights is applied to blocks predicted by planar mode, DC mode, and specific angular prediction modes. To benefit the prediction of sharp contents, MRL is designed to use the non-adjacent reference line for prediction rather than only using the nearest line as in traditional directional prediction. The non-adjacent reference line can be two or four lines away from the current block. CCLM is designed for the prediction of chroma components. The chroma components can be predicted from the collocated reconstructed luma samples by linear models whose parameters are derived from adjacent reconstructed samples.

B. Neural Network-Based Intra Prediction

Recently, neural network-based intra predictions have been rapidly developed and achieved impressive performance. In [8], a fully connected network-based intra prediction (IPFCN) was proposed to learn an end-to-end mapping from neighbouring reconstructed pixels to the target block. The neural networks were fed by eight reference lines above and on the left side of the target block. Two variants of IPFCN: IPFCN-S and IPFCN-D, were designed. IPFCN-S had a single network that was trained using unclassified training data, while IPFCN-D had two networks trained using the data from the non-angular prediction modes (DC and planar) and the other angular prediction modes separately.

Inspired by [8], more FCN-based intra prediction methods were proposed. In [11], different numbers of networks (up to 35) were designed for blocks of different sizes. All neural networks for a given block size shared all layers except the last one. Compared with IPFCN, the network structures were simplified. To improve the network training in [11], a novel training loss function was proposed in [12], which reflected properties of the residual quantization and coding stages by applying the $L_1$-norm and a sigmoid function to the prediction residual in the DCT domain. In a later version [13], a further simplified matrix-based intra prediction (MIP) was proposed. Each MIP mode had only one layer without non-linear activation, i.e., each network was reduced to a matrix plus a bias vector. Only one row and one column from the reconstructed picture were used for reference. Each MIP mode generated the intra prediction pixels by performing a down-sampling of the reference pixels, a matrix-vector multiplication, and an up-sampling of the result. MIP has been adopted by VVC.

To get higher coding performance, more complex network structures have been studied for intra prediction, for example, [14], [15], [16], [18], [19], and [20]. In [14], a progressive spatial recurrent neural network (PS-RNN) consisting of spatial recurrent units was designed. The prediction was progressively generated by passing information from preceding contents to blocks to be encoded. In [15], it was shown that convolutional neural networks yielded more accurate predictions than fully-connected networks in the case of large block sizes. In the subsequent work [16], a neural network-based transform selection was proposed to combine with neural network-based intra prediction modes. In [18], the intra prediction was modelled as an inpainting task. A generative adversarial network (GAN) was designed to fill in the missing part by conditioning on the available reconstructed pixels. In [19], a conditional autoencoder structure was designed to perform multi-mode prediction by feeding the mode information into a single network. The mode information was a vector generated from the L-shape reference and the original pixels in the target block via an encoder network. To improve the chroma prediction in VVC, a CNN-based chroma prediction (CNNCP) was proposed in [20]. CNNCP took the down-sampled luma block, neighbouring chroma blocks, and distortion level as inputs to generate the refined chroma prediction.

Neural networks could also be utilized for prediction result refinement. In [9], a convolutional neural network-based intra prediction was proposed. The block size was constrained to $8 \times 8$. After HEVC intra prediction, the target block and its three nearest reconstruction blocks formed a $16 \times 16$ block as the input of CNN. The output of the neural network was a residual block of the same size as the input. The prediction quality could be enhanced by subtracting the generated residual block from the original HEVC prediction result. MSCNN proposed in [10] further improved this method by increasing the number of supported block sizes. In addition, multi-scale feature extraction was proposed to take advantage of feature maps at different scales to improve the performance.

In the above methods, one or multiple parallel networks were trained to improve the intra prediction performance, and the training dataset was usually manually partitioned. In this paper, we propose to build networks in a tree-structured manner. The networks are trained with hierarchically clustered datasets.

III. THE FRAMEWORK OF TREE NET

Intra prediction aims to infer a prediction $\hat{Y}$ for a block $Y$ using its surrounding reference pixels $X$ as shown in Fig. 2, which can be expressed as:

$$\hat{Y} = F(X).$$  

(1)

$F$ represents the prediction function. The intra compression efficiency depends on the prediction accuracy and the additional overhead for representing $F$. To better balance the prediction accuracy and the overhead, we propose TreeNet to build networks hierarchically. In TreeNet, the networks at shallow levels cost smaller overhead and have better generalization abilities, while the networks at deep levels cost larger overhead and usually produce more accurate predictions.

In the following, the network structures in TreeNet are first introduced. Then the construction of TreeNet and the data clustering-driven training are presented. After that, the details of integrating TreeNet into the VVC are provided. Finally, the fast termination strategy for the search of TreeNet is described.
A. Network Structure in TreeNet

In [15], it is shown that FCN is more efficient in predicting small blocks, while CNN is more efficient in predicting large blocks. Therefore, both FCN and CNN are applied in TreeNet. The structures of an FCN and a CNN are shown in Fig. 3. The FCN consists of three fully connected layers (FCLs), while the CNN consists of two convolutional stacks followed by two FCLs. Each convolutional stack has four convolutional layers. TreeNet takes multiple reference lines surrounding the target block as the input for more accurate prediction [32], as shown in Fig. 3, where \( w \) and \( h \) represent the width and height of the target block. \( n_a \) and \( n_l \) are the number of reference lines above and on the left side. The input of the FCN: \( X \) is a 1D vector with a size of \( n_a(n_l + 2w) + 2hn_l \), which is reshaped from the 2D context. The inputs of the CNN: \( X_0 \) and \( X_1 \) are 2D matrices with sizes of \( (n_a, n_l + 2w) \) and \( (2h, n_l) \), respectively. The output of the FCN or CNN: \( \hat{Y} \) is a 1D vector of size \( w \times h \), corresponding to the prediction of \( Y \) in raster order.

In the FCN, all FCLs except the last one have the same output dimension and are followed by a non-linear activation layer. If the number of FCLs in the FCN is \( d \), the output of the \( i \)th layer \( \hat{Y}_i \) can be calculated as follows:

\[
\hat{Y}_i = f_i(\hat{Y}_{i-1}) = \begin{cases} 
\sigma(W_{i} \times \hat{Y}_{i-1} + B_{i}), & 1 \leq i < d \\
W_{i} \times \hat{Y}_{i-1} + B_{i}, & i = d 
\end{cases}
\]

where \( f_i(.) \) is the \( i \)th FCL, whose weight and bias are represented by \( W_{i} \) and \( B_{i} \), respectively. \( \hat{Y}_0 \) and \( \hat{Y}_d \) are the input and output of the FCN. \( \sigma \) is the LeakyReLU [33] with slope 0.1.

For the CNN, given the inputs: \( X_0 \) and \( X_1 \), the output \( \hat{Y} \) can be obtained by:

\[
\hat{Y} = f_2(f_1(M)),
\]

where

\[
M = \text{concat}(F(M_0), F(M_1)) \quad \text{(4)}
\]

\[
M_0 = C_0(X_0), M_1 = C_1(X_1) \quad \text{(5)}
\]

\( C_0(.) \) and \( C_1(.) \) are two stacks of convolutional layers. \( F(.) \) represents the flattening.

In TreeNet, blocks whose \( \min(w, h) \) is smaller than 16 are predicted by FCNs, in which \( n_a = n_l = \min(w, h) \). The other blocks are predicted by CNNs, in which \( n_a = h/2, n_l = w/2 \). Different networks are trained for \( 4 \times 4, 8 \times 8, 16 \times 4, 16 \times 8, 32 \times 4, 32 \times 8, 16 \times 16, \) and \( 32 \times 32 \) blocks. These networks can also predict blocks of other sizes by transposing or down-sampling the reference samples. For example, \( 8 \times 16 \) blocks share the same FCN as \( 16 \times 8 \) blocks. The CNN for \( 16 \times 16 \) blocks also predicts \( 32 \times 16 \) and \( 16 \times 32 \) blocks, and the CNN for \( 32 \times 32 \) blocks also predicts \( 64 \times 64 \) blocks. If the reference samples are down-sampled, a bi-linear up-sampling should be performed on the prediction. More details on the structures of FCNs and CNNs in TreeNet are provided in Appendix.

B. TreeNet Construction

In the traditional Linde-Buzo-Gray (LBG) data clustering algorithm [34], a perturbation is produced and applied as an addition or subtraction term for splitting one clustering centroid into two. Inspired by [34], we design a network split method for TreeNet. In each network split, every parent network on a leaf node is split into two networks by adding or subtracting noise to/from each fully connected (FC) layer. The two derived child networks have the same structure as their parent network. Fig. 4 is an example of splitting one parent network with FCN structure into two child networks, where \( N_i \) represents the noise introduced to the \( i \)th FC layer.
The output \( \hat{Y}_i \) of the \( i_{th} \) FC layer with noise added can be derived as follows:

\[
\hat{Y}_i = \sigma ((W_i + N(W_i))\hat{Y}_{i-1} + B_i + N(B_i)) = \sigma (W_i \hat{Y}_{i-1} + B_i + N(W_i)\hat{Y}_{i-1} + N(B_i))
\]

where \( \hat{Y}_{i-1}, W_i, B_i \) represent the input, weight, and bias. \( N(W_i) \) and \( N(B_i) \) are the noises added to the weight and the bias, respectively. \( \sigma \) is the activation function which can be formulated as:

\[
\sigma(x) = \begin{cases} 
  x, & \text{if } x \geq 0 \\
  0.1x, & \text{if } x < 0 
\end{cases}
\]

If \( N(W_i) \) and \( N(B_i) \) are small enough, the slope of the activation function will not be changed, then the output \( \hat{Y}_i \) can be represented as:

\[
\hat{Y}_i = \sigma (W_i \hat{Y}_{i-1} + B_i + N(W_i)\hat{Y}_{i-1} + N(B_i)) = \begin{cases} 
  \hat{Y}_{i-1} + N(W_i)\hat{Y}_{i-1} + N(B_i), & \text{if } \hat{Y}_i \geq 0 \\
  \hat{Y}_{i-1} + 0.1(N(W_i)\hat{Y}_{i-1} + N(B_i)), & \text{if } \hat{Y}_i < 0
\end{cases}
\]

where \( \hat{Y}_i \) is the output of the \( i_{th} \) FC layer without noise. The above formulas show that adding a small noise to an FC layer will result in a perturbation term being added to the output. The same conclusion can be obtained when the noise is subtracted. In TreeNet, Gaussian random noise is applied. The means of \( N(W_i) \) and \( N(B_i) \) are set to 0. The variances of \( N(W_i) \) and \( N(B_i) \) are set to 0.15\( W_i \) and 0.15\( B_i \) empirically. It should be noted that the perturbation term cannot improve the prediction ability of the network. However, it can make the two derived networks exhibit competitiveness in different training samples, which is necessary for data clustering-driven training.

Through our designed network split method, the networks in TreeNet can be built in a tree-structured manner. As TreeNet grows, the encoder complexity and memory usage will increase exponentially, and the overhead to indicate the networks will also increase. If the network split cannot bring enough prediction improvement, the network split will be terminated. An example of constructing TreeNet with depth = 4 is shown in Fig. 5. TreeNet is a perfect binary tree network, and the number of networks at Level \( l \) is \( 2^{l-1} \). The total number of networks in TreeNet with depth = \( L \) is \( 2^L - 1 \). After each network split, data clustering-driven training is applied to train every two derived child networks using the clustered training data of their parent.

### C. Data Clustering-Driven Training of TreeNet

Inspired by the K-means-like training strategy in [23], data clustering-driven training is applied to train every two derived child networks using the clustered training data of their parent. Fig. 6 is an example of data clustering and network training.

1) **Clustering and Training:** During each iteration of clustering and training, the training data of a parent network is first clustered according to the recovery quality of the two derived child networks. We use the squared error between the ground-truth pixels and the prediction to measure the recovery quality. Then the clustered training data is fed into the corresponding derived child networks for training. Given a collection of \( M \) training sample pairs, the loss function is formulated as:

\[
L(\Theta) = \frac{1}{M} \sum_{m=1}^{M} ||F(x_m|\Theta) - y_m||^2 + \gamma \|W\|^2,
\]

where \( F(x_m|\Theta) \) and \( y_m \) represent the network output and the ground truth, respectively. \( \Theta \) is the parameter set. \( \gamma \) represents the weight of the L2 regularization term, and it is set to be \( 5 \times 10^{-4} \) in the experiments. \( W \) is the weight of network layers. ADAM [35] with mini-batches of size 100 is applied as the optimization algorithm to update all parameters.

After several iterations of clustering and training, the training loss gradually converges, and the data clustering stabilizes, which indicates that the clustering centres no longer move. Then the data clustering-driven training will stop.

Taking TreeNet with depth = 4 for \( 8 \times 8 \) blocks as an example, we show the data clustering-driven training. The average training losses of each TreeNet mode during the recursive training are summarized in Table I. As shown in the table, the average training loss of each mode continuously decreases during the recursive training and gradually converges. We also conduct a statistic of the ratio of samples remaining in the same cluster after each round of data clustering to see whether the data clustering could finally converge. As shown in Table II, during the recursive training, less and
Table I

| Iteration | $N_2[1]$ | $N_2[2]$ |
|-----------|----------|----------|
| Level 2   |          |          |
| 1         | 144.83   | 156.27   |
| 2         | 137.53   | 145.79   |
| 3         | 136.45   | 142.70   |
| 4         | 135.07   | 141.24   |
| 5         | 134.97   | 139.49   |

| Iteration | $N_2[1]$ | $N_2[2]$ | $N_2[3]$ | $N_2[4]$ |
|-----------|----------|----------|----------|----------|
| Level 3   |          |          |          |          |
| 1         | 131.78   | 130.12   | 134.71   | 136.72   |
| 2         | 122.69   | 125.63   | 126.88   | 128.96   |
| 3         | 120.38   | 124.95   | 125.20   | 127.65   |
| 4         | 119.23   | 124.13   | 124.65   | 127.21   |
| 5         | 118.06   | 124.00   | 123.91   | 125.98   |

| Iteration | $N_3[1]$ | $N_3[2]$ | $N_3[3]$ | $N_3[4]$ | $N_3[5]$ | $N_3[6]$ | $N_3[7]$ | $N_3[8]$ |
|-----------|----------|----------|----------|----------|----------|----------|----------|----------|
| Level 4   |          |          |          |          |          |          |          |          |
| 1         | 126.16   | 108.67   | 129.75   | 116.02   | 135.02   | 113.27   | 121.82   | 127.59   |
| 2         | 115.72   | 104.90   | 122.12   | 108.72   | 125.20   | 107.02   | 116.55   | 118.04   |
| 3         | 112.95   | 104.51   | 121.13   | 107.55   | 123.21   | 106.24   | 114.23   | 115.94   |
| 4         | 111.37   | 104.33   | 120.72   | 106.51   | 121.94   | 105.18   | 115.09   | 115.22   |
| 5         | 110.58   | 104.23   | 120.89   | 105.68   | 120.69   | 105.46   | 115.45   | 114.50   |

Table II

| Iteration | $N_2[1]$ | $N_2[2]$ |
|-----------|----------|----------|
| Level 2   |          |          |
| 2         | 82.6%    | 82.6%    |
| 3         | 92.2%    | 93.1%    |
| 4         | 95.2%    | 95.1%    |
| 5         | 97.7%    | 97.0%    |

| Iteration | $N_2[1]$ | $N_2[2]$ | $N_2[3]$ | $N_2[4]$ |
|-----------|----------|----------|----------|----------|
| Level 3   |          |          |          |          |
| 2         | 82.8%    | 80.9%    | 82.7%    | 82.8%    |
| 3         | 94.1%    | 92.6%    | 94.1%    | 93.9%    |
| 4         | 96.2%    | 95.2%    | 96.0%    | 95.9%    |
| 5         | 97.0%    | 96.2%    | 97.1%    | 97.0%    |

| Iteration | $N_2[1]$ | $N_2[2]$ | $N_2[3]$ | $N_2[4]$ | $N_2[5]$ | $N_2[6]$ | $N_2[7]$ | $N_2[8]$ |
|-----------|----------|----------|----------|----------|----------|----------|----------|----------|
| Level 4   |          |          |          |          |          |          |          |          |
| 2         | 81.6%    | 81.7%    | 80.6%    | 82.9%    | 80.7%    | 82.9%    | 81.0%    | 80.6%    |
| 3         | 95.1%    | 94.2%    | 94.0%    | 95.4%    | 94.7%    | 95.0%    | 94.4%    | 94.5%    |
| 4         | 97.2%    | 96.5%    | 96.3%    | 97.3%    | 96.9%    | 96.9%    | 96.7%    | 96.8%    |
| 5         | 97.9%    | 97.7%    | 97.3%    | 98.2%    | 97.8%    | 97.8%    | 97.8%    | 97.7%    |

Table III

| Iteration | $N_3[1]$ | $N_3[2]$ |
|-----------|----------|----------|
| Level 2   |          |          |
| 1         | 52.4%    | 47.6%    |
| 5         | 50.3%    | 49.7%    |

| Iteration | $N_3[1]$ | $N_3[2]$ | $N_3[3]$ | $N_3[4]$ |
|-----------|----------|----------|----------|----------|
| Level 3   |          |          |          |          |
| 1         | 23.9%    | 26.4%    | 25.2%    | 24.5%    |
| 5         | 25.7%    | 24.6%    | 25.1%    | 24.6%    |

| Iteration | $N_3[1]$ | $N_3[2]$ | $N_3[3]$ | $N_3[4]$ | $N_3[5]$ | $N_3[6]$ | $N_3[7]$ | $N_3[8]$ |
|-----------|----------|----------|----------|----------|----------|----------|----------|----------|
| Level 4   |          |          |          |          |          |          |          |          |
| 1         | 11.3%    | 14.2%    | 12.0%    | 12.6%    | 10.4%    | 14.7%    | 12.6%    | 12.0%    |
| 5         | 12.4%    | 13.3%    | 11.6%    | 13.0%    | 11.2%    | 13.9%    | 12.4%    | 12.2%    |

are initialized via Xavier’s initialization [36] to accelerate the training. The weights of the first fully connected layer in FCNs (FCL1 in Fig. 3(a)) and the first convolution layers in the CNN (Conv1 and Conv5 in Fig. 3(b)) are randomly generated from a Gaussian distribution with a mean of 0 and a standard deviation of 0.1 to reduce the variance of the output. The biases $B$ of all network layers in FCNs and CNNs are initialized as 0.

After initialization, a root network of TreeNet is pre-trained for 60 epochs. The base learning rate is set to $2 \times 10^{-4}$ as recommended in [16]. The network is first trained with a fixed learning rate for 30 epochs and then uses the exponentially decaying learning rate for another 30 epochs. The learning rate settings are listed in Table IV.

D. Integrating TreeNet in VVC

To test the performance of TreeNet, we integrate TreeNet into VVC reference software VTM-17.0 as shown in Fig. 7. In VVC, the best prediction mode is decided at the CU level. For a CU whose top-left pixel is at position $(x, y)$ in the current frame, TreeNet modes are disabled when $x < n_l$ or $y < n_u$. If TreeNet is enabled, the rate-distortion optimization (RDO) [37] will be used to select the optimal mode between TreeNet and the original VVC intra modes, which includes the directional modes and MIP. Then a new flag $S$ has to be coded and transmitted in the bitstream to indicate whether TreeNet is used. If all VVC Intra modes are replaced by TreeNet, the flag $S$ can be ignored.
TABLE V
THE CODEWORD OF EACH MODE IN TREENET WITH DEPTH = 4

| Level | Mode | Binarization |
|-------|------|--------------|
|       |      | p c p c p c |
| 1     | 1    | 0 0         |
| 2     | 2    | 1 0 1 0     |
| 3     | 3    | 1 1 1 1     |
| 4     | 4    | 1 0 1 0     |
|       | 5    | 1 0 1 1     |
|       | 6    | 1 1 1 0     |
|       | 7    | 1 1 1 0     |
|       | 8    | 1 1 1 0     |
|       | 9    | 1 1 1 0     |
|       | 10   | 1 1 1 0     |
|       | 11   | 1 1 1 1     |
|       | 12   | 1 1 1 1     |
|       | 13   | 1 1 1 1     |
|       | 14   | 1 1 1 1     |
|       | 15   | 1 1 1 1     |

Since multiple advanced tools have been introduced in VTM-17.0, the collaboration between TreeNet and these tools must be carefully designed. Similar to MRL, TreeNet does not work with Intra Sub-Partitions (ISP) coding mode [38] since a single TreeNet mode cannot handle blocks of different sizes. TreeNet works together with transform tools like Multiple Transform Selection (MTS) and Low Frequency Non-Separable Transform (LFNST). It should be noted that when the VVC intra modes are applied, the index of the LFNST matrices for the secondary transform and whether the primary transform coefficients are transposed are implicitly decided by the mode index. When TreeNet is applied, all LFNST modes are searched, and the optimal one in terms of RDO is written to the bitstream via truncated binary code.

The codeword of each mode in TreeNet with depth = 4 is shown in Table V. At each branch, one bit is needed to represent if the parent network is better than the child networks, which is indicated by $p$. If a child network works better, another bit is needed to represent which child network is better, which is indicated by $c$. Therefore, 1, 3, 5, and 6 bits are needed for coding TreeNet modes at Level 1, 2, 3, and 4, respectively. Context-Based Adaptive Binary Arithmetic Coding (CABAC) [39] is used to encode the codewords, and a context model is built for a corresponding bit. The reasonably allocated codewords for modes at different levels make TreeNet more practical under different bitrate conditions.

E. Fast Termination Strategy

After TreeNet is well-trained and integrated, it can be used for intra prediction. However, a full search of TreeNet is very time-consuming. Since the training of derived child networks only uses the clustered training data from their parent network, the predictions between a parent network and its two child networks are correlated, which leads to the design of the fast termination strategy for TreeNet. The proposed fast termination strategy is shown in Algorithm 1. The rate-distortion (R-D) cost is used to compare different TreeNet modes, and it can be expressed as:

$$J = D + \lambda Q P \ast R$$

(10)

where $J$, $D$, and $R$ represent the R-D cost, the sum of squared error (SSE), and costed bits, respectively. $\lambda Q P$ is the Lagrange multiplier decided by a quantization parameter (QP).

In Algorithm 1, $C$ and $P$ represent the derived child and the parent networks, respectively. $\alpha$ and $\beta$ are two coefficients larger than 1. If both $J_C$ and $J_P$ are larger than $\alpha \ast J_P$, which indicates that the parent network predicts much better than either of the two child networks, the search of TreeNet will be early terminated. Otherwise, TreeNet modes at the deeper level will be searched. In this case, if $\max(J_C, J_P)$ is larger than $\beta \ast \min(J_C, J_P)$, which indicates that one child network predicts much better than another one, then the downward search along the child network with the higher R-D cost will be skipped. Otherwise, the derived networks of both the two child networks will be searched. The fast termination strategy could significantly save the encoding time by reducing the number of TreeNet modes to be searched adaptively. The settings of $\alpha$ and $\beta$ should depend on the QP. When QP is small, $\lambda Q P$ is also small, and the RD-cost is more sensitive to the distortion $D$. Then $\alpha$ and $\beta$ should be set larger to enable a deeper search of TreeNet. After exhaustive experiments, both $\alpha$ and $\beta$ are set to $1 + ((51 - Q P)^2/2500)$.

IV. EXPERIMENTAL RESULTS

In this section, the experimental results of TreeNet are presented and analyzed in detail. We first introduce the experiment setting and the training dataset briefly. Then, the coding performance of TreeNet when assisting and replacing VVC intra modes is presented. After that, the usage ratio of TreeNet is analyzed, followed by a comparison with the state-of-the-art methods. Finally, the complexity of TreeNet is provided.

A. Experiment Setting

TreeNet is implemented into VVC reference software VTM-17.0 to test its performance. VTM-17.0 is also the comparison anchor in our experiments. The training of TreeNet is based on Pytorch [40], and Libtorch is used to perform the feed-forward of networks in the codec. All experiments comply with the common test conditions (CTC) specified in [41] with QP set to 22, 27, 32, 37, and the all-intra main configuration is deployed. 22 video sequences with different resolutions, grouped as Class A, B, C, D, and E, are utilized for experiments, and the first 10 temporally-sampled frames of each sequence are tested. Bjøntegaard Delta rate (BD-rate) [42] is used to evaluate the performance.

B. Training Dataset

The training data is generated from 100,000 RGB images with various resolutions (from $32 \times 32$ to $3552 \times 4680$) randomly picked from ILSVRC 2012 dataset [43] and 800 2K images from DIV2K [44]. Firstly, these images are converted to YUV 4:2:0 colour format. Then all training images are encoded via VTM-17.0 with all-intra configuration. The QPs
TABLE VI
THE BD-RATE RESULTS OF TREE NET ON THE LUMA COMPONENT WHEN ASSISTING VVC INTRA MODES

| Sequence       | Class A (4K) | Class B (1080P) | Class C (WVGA) | Class D (WQVGA) | Class E (720P) | Average of Class A | Average |
|----------------|--------------|-----------------|----------------|------------------|----------------|--------------------|---------|
|                |               |                 |                |                  |                |                    |         |
|                | Depth = 1    | Depth = 2       | Depth = 3      | Depth = 4        | Depth = 3      | Depth = 4          |         |
| TreeNet        | -5.47%       | -5.65%          | -5.88%         | -5.98%           | -5.76%         | -5.93%             |         |
|                | -7.43%       | -7.79%          | -8.01%         | -8.14%           | -8.12%         | -8.02%             |         |
|                | -2.70%       | -3.12%          | -3.30%         | -3.44%           | -3.25%         | -3.27%             |         |
|                | -2.97%       | -3.22%          | -3.25%         | -3.45%           | -3.30%         | -3.31%             |         |
|                | -2.46%       | -2.55%          | -2.55%         | -2.70%           | -2.54%         | -2.69%             |         |
|                | -2.30%       | -2.31%          | -2.40%         | -2.44%           | -2.38%         | -2.42%             |         |
|                | -2.61%       | -2.56%          | -2.76%         | -2.65%           | -2.61%         | -2.66%             |         |
|                | -4.16%       | -4.34%          | -4.65%         | -4.72%           | -4.43%         | -4.49%             |         |
|                | -2.73%       | -2.93%          | -3.08%         | -3.18%           | -3.04%         | -3.14%             |         |
|                | -3.77%       | -4.03%          | -4.19%         | -4.27%           | -4.25%         | -4.24%             |         |
|                | -1.90%       | -2.20%          | -2.36%         | -2.39%           | -2.28%         | -2.33%             |         |
|                | -3.17%       | -3.02%          | -3.46%         | -3.39%           | -3.41%         | -3.44%             |         |
|                | -3.22%       | -3.45%          | -3.66%         | -3.81%           | -3.57%         | -3.55%             |         |
|                | -2.66%       | -2.98%          | -3.23%         | -3.32%           | -3.11%         | -3.09%             |         |
|                | -2.55%       | -2.71%          | -2.81%         | -2.91%           | -2.82%         | -2.83%             |         |
|                | -2.75%       | -2.73%          | -3.23%         | -3.56%           | -3.39%         | -3.19%             |         |
|                | -2.65%       | -3.04%          | -3.24%         | -3.27%           | -3.00%         | -3.05%             |         |
|                | -3.60%       | -3.64%          | -3.59%         | -3.58%           | -3.59%         | -3.74%             |         |
|                | -3.51%       | -3.81%          | -3.97%         | -4.08%           | -4.10%         | -3.94%             |         |
|                | -3.94%       | -4.35%          | -4.50%         | -4.64%           | -4.57%         | -4.50%             |         |
|                | -4.37%       | -4.99%          | -5.09%         | -5.32%           | -5.15%         | -5.08%             |         |
|                | -3.99%       | -4.39%          | -4.49%         | -4.38%           | -4.39%         | -4.67%             |         |

-5.38%  -4.11%  -4.23%  -4.36%  -4.22%  -4.27%

| TreeNet        | Depth = 1    | Depth = 2       | Depth = 3      | Depth = 4        | Depth = 3      | Depth = 4          |         |
| Fast TreeNet   | Depth = 1    | Depth = 2       | Depth = 3      | Depth = 4        | Depth = 3      | Depth = 4          |         |
|                | -3.41%       | -3.63%          | -3.80%         | -3.89%           | -3.78%         | -3.80%             |         |

-3.89%  -4.11%  -4.23%  -4.36%  -4.22%  -4.27%

are set to 22, 27, 32, and 37. The reconstructed reference samples of a block are extracted from the bitstream on the decoder side as network input. The original pixels of the block are taken as the ground truth. After generating the training data, a TreeNet is trained for each block size using the data from the four QPs. In our experiments, a total of 90 million data samples are collected to generate 9 datasets to train 9 TreeNet models for predicting blocks of different sizes. Each dataset consists of 10 million data samples.

It should be noted that the dataset should be pre-processed for more efficient network training. First, the input and output pixel values are mapped to the 8-bit depth from the internal bit-depth, which is 10-bit in VVC. Then, the mean value of available reference pixels is subtracted from both the input and the output to eliminate the low-frequency component and make the training easier. Finally, if some required reference pixels of the input are not available, the vacancies are filled in by a constant value of 255 to avoid ambiguities with the available reference pixels. After the network inference, the mean value should be added back to the output. Then the output is remapped and clipped to the internal bit-depth.

C. Coding Performance of TreeNet When Assisting VVC Intra Modes

In the experiments, we test the performance of applying TreeNet to assist the VVC intra modes. The results are shown in Table VI. For simplicity, TreeNet applying the fast termination strategy is referred to as fast TreeNet in the following. TreeNet and fast TreeNet with depth = i are shortened as TN_Di and FTN_Di, respectively. As observed, TreeNet with depth = 4, 3, 2, 1 can bring an average BD-rate saving of 3.89%, 3.80%, 3.63%, and 3.41% for the luma component, respectively. The experimental results indicate that the deeper TreeNet can improve the coding efficiency more significantly. After applying the fast termination strategy, the performance of TreeNet will be slightly degraded. Taking FTN_D3 as an example, it could bring 3.78% BD-rate saving on average, which is 0.02% less than that of TN_D3. The BD-rate loss of FTN_D4 over TN_D4 is 0.09% on average. For all test sequences, FTN_D3 brings similar gains as FTN_D4. The complexity of fast TreeNet is given in Section IV-G.

In addition to the BD-rate saving, some visual examples of the prediction of TreeNet are also provided and compared with VVC intra prediction. The examples are taken from the first frame of *FourPeople* encoded with QP = 32. The comparison results are shown in Figs. 8, 9, and 10. It can
Algorithm 1 The Fast Termination Strategy

**Input:** TreeNet with depth \(L_{\text{max}}\)
- Lagrange multiplier: \(\lambda_{\text{QP}}\)
- \(L\)-shape reference: \(X\), Original pixels: \(Y\)

**Output:** Minimum R-D cost: \(R_{\text{D,min}}\)
- Level: \(L\)
- Network Index: \(I\)

\(l = 1, \ i = 1\)

**Function** TreeSearch \((l, i)\):

\[
P = N_{i}[i]
\]
\[
C_1 = N_{i+1}[2 * i - 1], \ C_2 = N_{i+1}[2 * i]
\]
\[
J_P = D(P(X), Y) + \lambda_{\text{QP}} * R_P
\]
\[
J_{C_1} = D(C_1(X), Y) + \lambda_{\text{QP}} * R_{C_1}
\]
\[
J_{C_2} = D(C_2(X), Y) + \lambda_{\text{QP}} * R_{C_2}
\]

if \(J_P \leq \min(J_{C_1}, J_{C_2})\) then

\[
L = l, \ I = i, \ R_{\text{D,min}} = J_P
\]

else if \(J_{C_1} < J_{C_2}\) then

\[
L = l + 1, \ I = 2 * i - 1, \ R_{\text{D,min}} = J_{C_1}
\]

else if \(\max(J_{C_1}, J_{C_2}) > \beta * \min(J_{C_1}, J_{C_2})\) then

\[
\text{if } J_{C_1} < J_{C_2} \text{ then}
\]

\[
\text{RD}_{l, l, l} = \text{TreeSearch}(l + 1, 2 * i - 1)
\]

else

\[
\text{RD}_{l, l, l} = \text{TreeSearch}(l + 1, 2 * i)
\]

if \(\text{RD}_{l} < \text{RD}_{\text{D,min}}\) then

\[
\text{RD}_{\text{D,min}} = \text{RD}_{l}, \ L = l_t, \ I = I_t
\]

else

\[
s = -1
\]

while \(s \leq 0\) do

\[
\text{RD}_{l, l, l} = \text{TreeSearch}(l + 1, 2 * i + s)
\]

if \(\text{RD}_{l} < \text{RD}_{\text{D,min}}\) then

\[
\text{RD}_{\text{D,min}} = \text{RD}_{l}, \ L = l_t, \ I = I_t
\]

\[
s = s + 1
\]

**return** \(\text{RD}_{\text{D,min}}, \ L, \ I\)

\(*N_{i}[i]\) is the \(i_{th}\) network at level \(l\).

Fig. 9. Prediction of a 16 × 16 block located at (296,232) (a) 8-line reference samples and original pixels. (b) original pixels (c) prediction via the best TreeNet mode at Level 1 (MSE: 2611.9) (d) prediction via the best VVC intra mode of index 48 (MSE: 10733.1) (e) predictions via the TreeNet modes at Level 1 (MSE: 2611.9) and Level 2 (MSE: 3728.7, 5273.6).

be observed that TreeNet produces more accurate predictions than the VVC intra modes. As shown in Fig. 8 and Fig. 9, TreeNet predicts better than VVC intra modes for the shadow at the edges of the blocks. In Fig. 10, the original block has an obvious directional pattern. TreeNet could still generate better predictions than VVC directional prediction modes. By observing the predictions from TreeNet modes at different levels, it is found that child networks will not always predict more accurately than their parent network. The generalization ability learned from a larger training dataset makes the parent network sometimes more applicable.

Since the textures of chroma blocks are usually smooth and simple, only the root network of TreeNet is used for predicting the chroma components. A chroma block will be predicted by the root network when its collocated luma block is predicted by a TreeNet mode and the chroma prediction mode is INTRA_DERIVED. We test the BD-rate gain on the chroma components when the luma component is predicted by FTN_D3 and FTN_D4, respectively. The results are summarized in Table VII. As shown in Table VII, the BD-rate improvements over VTM-17.0 on two chroma components reach 2.21% and 2.51% when the luma component is predicted by FTN_D3. When FTN_D4 is applied for the luma component, the BD-rate gains on chroma components slightly decrease, which may be caused by the inaccurate chroma prediction in some regions where a TreeNet mode is selected as the optimal luma mode. In our future work, more chroma network modes corresponding to the luma TreeNet modes could be integrated for more accurate prediction.

**Table VII**

| Luma Using Fast TreeNet | Video Class | Depth = 3 | Depth = 4 |
|-------------------------|-------------|-----------|-----------|
|                         | U          | V         | U          | V         |
| A                       | -2.66%     | -2.67%    | -2.67%     | -2.51%    |
| B                       | -1.39%     | -2.66%    | -1.27%     | -2.13%    |
| C                       | -1.95%     | -2.29%    | -1.62%     | -2.35%    |
| D                       | -0.61%     | -1.92%    | -0.32%     | -2.00%    |
| E                       | -3.03%     | -2.25%    | -3.36%     | -2.82%    |
| **Average**             | **-2.21%** | **-2.51%**| **-2.16%** | **-2.42%**|

**D. Coding Performance of TreeNet When Replacing VVC Intra Modes**

In [15], the network modes replace parts of the HEVC directional modes. In the experiments, all VVC intra modes,
including planar mode, DC mode, 93 angular modes, and MIP, are replaced with TreeNet for the luma component. As far as we know, this is the first work replacing all intra modes with neural networks.

The BD-rate results are presented in Table VIII. As shown in Table VIII, FTN_D3 and FTN_D4 can bring an average of 1.59% and 1.87% BD-rate reduction over VTM-17.0 on the luma component, respectively, which indicates TreeNet replacing all the VVC intra modes still performs well. The BD-rate gains on the high-resolution sequences are more significant, reaching 2.77% and 2.97% on average. By comparing Table VIII with Table VI, it is also found that more networks should be adopted when TreeNet is used to replace VVC intra modes than assisting them.

### TABLE IX

| Class | Sequence            | TreeNet | VTM-17.0 |
|-------|---------------------|---------|----------|
| A (4K) |                             |         |          |
|       | Tango2               | -1.74%  | -1.59%   |
|       | FoodMarket4          | -1.75%  | -1.79%   |
|       | Campfire             | -1.28%  | -1.47%   |
|       | CatRobot             | -1.70%  | -1.91%   |
|       | DaylightRoad2        | -1.70%  | -1.91%   |
|       | ParkRunning3         | -1.81%  | -1.90%   |
| B (1080P) | MarketPlace         | -1.74%  | -1.90%   |
|       | RitualDance          | -1.78%  | -1.79%   |
|       | Cactus               | -1.28%  | -1.01%   |
|       | BasketballDrive      | -2.26%  | -2.58%   |
|       | BQTerrace            | 0.43%   | 1.00%    |
| C (WVGA) | BasketballDrill     | 2.86%   | 2.62%    |
|       | BQMall               | -0.95%  | -0.71%   |
|       | PartyScene           | -1.70%  | -1.78%   |
|       | RaceHorses           | -1.81%  | -1.13%   |
| D (WQVGA) | BasketballPass       | 0.02%   | 2.61%    |
|       | BQSquare             | 0.09%   | 1.70%    |
|       | BlowingBubbles       | -2.49%  | -0.51%   |
|       | RaceHorses           | -2.75%  | -0.36%   |
| E (720P) | FourPeople           | -1.87%  | -2.75%   |
|       | Johnny               | -1.97%  | -2.17%   |
|       | KristenAndSara       | -1.22%  | -0.32%   |
| Average | -2.77%              | -1.43%  | -0.98%   |

### TABLE IX

| Class | Sequence            | TreeNet | VTM-17.0 |
|-------|---------------------|---------|----------|
| A (4K) |                             |         |          |
|       | Tango2               | 77.9%   | 79.7%    |
|       | FoodMarket4          | 75.1%   | 76.7%    |
|       | Campfire             | 72.0%   | 75.9%    |
|       | CatRobot             | 61.4%   | 60.4%    |
|       | DaylightRoad2        | 63.2%   | 63.3%    |
|       | ParkRunning3         | 65.5%   | 67.2%    |
| B (1080P) | MarketPlace         | 67.3%   | 68.2%    |
|       | RitualDance          | 56.1%   | 48.1%    |
|       | Cactus               | 62.7%   | 69.9%    |
|       | BasketballDrive      | 61.4%   | 64.8%    |
|       | BQTerrace            | 49.3%   | 53.3%    |
| C (WVGA) | BasketballDrill     | 43.6%   | 47.7%    |
|       | BQMall               | 52.2%   | 57.2%    |
|       | PartyScene           | 60.6%   | 66.5%    |
|       | RaceHorses           | 65.4%   | 70.2%    |
| D (WQVGA) | BasketballPass       | 48.9%   | 31.0%    |
|       | BQSquare             | 47.9%   | 52.3%    |
|       | BlowingBubbles       | 61.9%   | 65.6%    |
|       | RaceHorses           | 60.7%   | 63.4%    |
| E (720P) | FourPeople           | 59.4%   | 62.7%    |
|       | Johnny               | 57.2%   | 55.5%    |
|       | KristenAndSara       | 59.5%   | 57.3%    |
| Average | 60.4%               | 62.3%   | 63.6%    |

### E. Usage Ratio

Table IX tabulates the pixel-level usage ratios of TN_D1, TN_D2, FTN_D3, and FTN_D4 for each test sequence when assisting the VVC intra modes. The pixel-level usage ratio in an image can be calculated as

$$\eta = \frac{\sum_{i=1}^{17} (n_i \times w_i \times h_i)}{W \times H} \times 100\%$$

where $\eta$ denotes the usage ratio, $W$ and $H$ denote the width and height of each frame, $n_i$ represents the number of CUs with a size of $w_i \times h_i$ coded by TreeNet in each frame. The usage ratio is the average value for all tested frames under QP = 22, 27, 32, and 37. As shown in Table IX, the usage ratios of TreeNet in all sequences are remarkable, especially in Class A. Taking FTN_D3 as an example, the average usage ratio of TreeNet is around 63.6% (up to 80.8%), demonstrating its efficiency. In addition, it is found that the usage ratio of TreeNet is positively correlated with its depth.

To further analyze the usage of modes at each level of TreeNet under different QPs, the CU-level usage ratio of FTN_D3 in encoding the first frame of ParkRunning3 is provided. As shown in Fig. 11, TreeNet is effective in predicting CUs of all sizes. For blocks of size $4 \times 4, 4 \times 8, 8 \times 4,$ and
8 × 8, the usage ratios of TreeNet are more than 90%, indicating that TreeNet has more advantage in predicting small CUs than VVC intra modes. In addition, it is found that the modes at Level 1 are the most selected in TreeNet. Generally, TreeNet modes at deeper levels account for a higher proportion of the overall usage under low-QP test conditions than high-QP.

Fig. 12 shows the visual distribution of CUs coded by FTN_D3. Yellow, cyan-blue, and red boxes represent CUs encoded by networks at Level 1, 2, and 3, respectively. The VVC intra modes encode the parts not enclosed by boxes. As shown in Fig. 12 (a) (QP 22), the amounts of CUs encoded by TreeNet modes at different levels look similar. In Fig. 12 (b) - (d) (QP 27, 32, 37), the area covered by yellow blocks is more than blocks of the other two colours. In Fig. 12 (d), the background parts, whose textures are smooth and simple, are encoded mainly by the modes at Level 1. The modes at Level 3 are only applied to areas with sharp edges, such as the man’s hair and face. The above observations further demonstrate that TreeNet can be applied flexibly at various bitrate conditions and in areas with different textures.
F. Comparison With the State-of-the-Art Methods

In order to further evaluate the coding performance of TreeNet, it is compared with the latest deep learning-based intra prediction methods [12] and [16], [17], [18], and [19]. The comparison results for the luma component are presented in Table X.

As shown in Table X, the LFNST mode prediction in [16] brings an average of 3.66% BD-rate gain over VTM-8.0. The conditional autoencoder proposed in [19] brings 1.09% BD-rate gain over VTM-6.2 on average. A single network trained with iteratively updated dataset in [17] brings an average of 1.9% BD-rate reduction over VTM-5.0 when testing on Class A, B, C, and D. The GAN-based approach proposed in [18] brings around 6.75% over VTM-1.1. In [12], an average of 3.66% BD-rate gain over VTM-1.0 is reached by integrating plenty of simplified network modes (up to 35) for each block size. As observed, the GAN-based approach [18] performs much better on Class C and E. Our proposed tree-structured data clustering-driven training can be applied to combine with these state-of-the-art methods to further improve the coding performance, e.g. [18].

G. Complexity

The running times of applying TreeNet to assist VVC Intra modes are summarized in Table XI. All tests are conducted on an AMD Ryzen7 5800H CPU@3.2GHz. The anchor is VTM-17.0. Due to the numerous floating-point operations from the feed-forward of TreeNet modes, the computational cost on both the encoder and decoder sides increases significantly. Taking TreeNet with depth = 3 as an example, the encoding time is around 21 times that of the anchor. After applying the fast termination strategy, the encoding time is reduced to around 16 times that of the anchor. As deeper TreeNet is applied, the encoding time increases exponentially. The decoding time of TreeNet with depth = 3 is roughly 30 times that of the anchor. The encoding and decoding with TreeNet can be accelerated by applying a GPU device.

In addition to the encoding and decoding time, we also provide information on the model size and the required multiply-accumulate (MAC) operations for predicting per pixel of each TreeNet model, as shown in Table XII. The model sizes for 32 × 32, 16 × 16, 32 × 4, 16 × 8, 16 × 4, 8 × 8, 8 × 4, and 4 × 4 blocks are 14.8MB, 11.3MB, 9.9MB, 7.5MB, 8.1 MB, 6.6 MB, 7.2 MB, 6.2 MB, and 5.9 MB, respectively. If TreeNet with depth = 3 is applied, there will be 7 models for each block size, and overall 7 × 80.0 = 560.0 MB is needed to store TreeNet in memory. Among all TreeNet models, the model for 4 × 4 blocks has the highest multiply-accumulate operations for predicting per pixel (worst case), reaching 97.2 kMAC/pixel. The multiply-accumulate operations of CNNs, which have deeper network structures than FCNs, are only 11.3 kMAC/pixel and 16.7 kMAC/pixel in predicting 32 × 32 and 16 × 16 blocks, respectively.

V. Conclusion

In this paper, a novel tree-structured data clustering-driven neural network for intra prediction is proposed, which builds the networks and clusters the training data in a tree-structured manner. Specifically, in each network split and training process of TreeNet, each parent network on a leaf node is split into two child networks by adding or subtracting Gaussian random noise. Then data clustering-driven training is applied to train the two derived child networks. On the one hand, the networks at the same level in TreeNet are trained with non-overlapping clustered datasets, and thus they can learn different prediction abilities. On the other hand, the networks at different levels are trained with hierarchically clustered datasets, and thus they will have different generalization abilities. TreeNet could be...
applied to assist or replace VVC intra prediction modes and has shown its advantage compared to multi-parallel networks. In the future, we will focus on simplifying TreeNet and making it more practical for industrial applications.

APPENDIX

NETWORKS STRUCTURES IN TREE-NET

In the appendix, we provide details on the networks structures shown in Fig. 3 (Section III-A). The structure of the FCN is listed in Table XIII. The structures of convolutional layers and fully connected layers in the CNN are listed in Table XIV and Table XV. $a$ and $b$ are two variables in the structure of CNN. In the CNN for $16 \times 16$, $32 \times 16$, and $16 \times 32$ blocks, $a$ and $b$ are set to 1 and 512, respectively. In the CNN for $32 \times 32$ and $64 \times 64$ blocks, $a$ and $b$ are increased to 2 and 1024.

REFERENCES

[1] Versatile Video Coding, document ITU-T Rec. H.266 and ISO/IEC 23090-3, ITU-T and ISO/IEC, 2020.
[2] High Efficiency Video Coding, document ITU-T Rec. H.265 and ISO/IEC 23008-2, V.1, ITU-T and ISO/IEC, 2013.
[3] G. J. Sullivan, J. Ohm, W. Han, and T. Wiegand, “Overview of the high efficiency video coding (HEVC) standard,” IEEE Trans. Circuits Syst. Video Technol., vol. 22, no. 12, pp. 1649–1668, Dec. 2012.
[4] B. Bross et al., “Overview of the versatile video coding (VVC) standard and its applications,” IEEE Trans. Circuits Syst. Video Technol., vol. 31, no. 10, pp. 3736–3764, Oct. 2021.
[5] J. Pfaff et al., “Intra prediction and mode coding in VVC,” IEEE Trans. Circuits Syst. Video Technol., vol. 31, no. 10, pp. 3834–3847, Oct. 2021.
[6] C. T sai et al., “Adaptive loop filtering for video coding,” IEEE J. Sel. Topics Signal Process., vol. 7, no. 6, pp. 934–945, Dec. 2013.
[7] D. Liu, Z. Chen, S. Liu, and F. Wu, “Deep learning-based technology in responses to the joint call for proposals on video compression with capability beyond HEVC,” IEEE Trans. Circuits Syst. Video Technol., vol. 30, no. 5, pp. 1267–1280, May 2020.
[8] J. Li, B. Li, J. Xu, R. Xiong, and W. Gao, “Fully connected network-based intra prediction for image coding,” IEEE Trans. Image Process., vol. 27, no. 7, pp. 3230–3247, Jul. 2018.
[9] W. Cui et al., “Convolutional neural networks based intra prediction for HEVC,” in Proc. Data Compress. Conf. (DCC), Apr. 2017, pp. 436.
[10] Y. Wang, X. Fan, S. Liu, D. Zhao, and W. Gao, “Multi-scale convolutional neural network-based intra prediction for video coding,” IEEE Trans. Circuits Syst. Video Technol., vol. 30, no. 7, pp. 1803–1815, Jul. 2020.
[11] J. Pfaff et al., “Neural network based intra prediction for video coding,” Proc. SPIE, vol. 10752, pp. 326–336, Sep. 2018.
[12] P. Helle et al., “Intra picture prediction for video coding with neural networks,” in Proc. Data Compress. Conf. (DCC), Mar. 2019, pp. 448–457.
[13] J. Pfaff et al. Non-CES: Simplications of MIP, document JVET-O0925, Gothenburg, Jul. 2019.
[14] Y. Hu, W. Yang, M. Li, and J. Liu, “Progressive spatial recurrent neural network for intra prediction,” IEEE Trans. Multimedia, vol. 21, no. 12, pp. 3024–3037, Dec. 2019.
[15] L. Dumas, A. Rouzy, and C. Guillemot, “Context-adaptive neural network-based prediction for image compression,” IEEE Trans. Image Process., vol. 29, pp. 679–693, 2020.
[16] T. Dumas, F. Galpin, and P. Bordes, “Combined neural network-based intra prediction and transform selection,” in Proc. Picture Coding Symp. (PCS), Jun. 2021, pp. 1–5.
[17] T. Dumas, F. Galpin, and P. Bordes, “Iterative training of neural networks for intra prediction,” IEEE Trans. Image Process., vol. 30, pp. 697–711, 2021.
[18] L. Zhu, S. Kwong, Y. Zhang, S. Wang, and X. Wang, “Generative adversarial network-based intra prediction for video coding,” IEEE Trans. Multimedia, vol. 22, no. 1, pp. 45–58, Jan. 2020.
[19] F. Brand, J. Seiler, and A. Kaup, “Intra-frame coding using a conditional autoencoder,” IEEE J. Sel. Topics Signal Process., vol. 15, no. 2, pp. 354–365, Feb. 2021.
[20] L. Zhu, Y. Zhang, S. Wang, S. Kwong, X. Jin, and Y. Qiao, “Deep learning-based chroma prediction for intra versatile video coding,” IEEE Trans. Circuits Syst. Video Technol., vol. 31, no. 8, pp. 3168–3181, Aug. 2021.
[21] N. Yan, D. L iu, H. Li, B. Li, L. Li, and F. Wu, “Convolutional neural network-based fractional-pixel motion compensation,” IEEE Trans. Circuits Syst. Video Technol., vol. 29, no. 3, pp. 840–853, Mar. 2019.
[22] Y. Wang, X. Fan, R. Xiong, D. Zhao, and W. Gao, “Neural network-based enhancement to inter prediction for video coding,” IEEE Trans. Circuits Syst. Video Technol., vol. 32, no. 2, pp. 826–838, Feb. 2022.
[23] C. Jia et al., “Content-aware convolutional neural network for in-loop filtering in high efficiency video coding,” IEEE Trans. Image Process., vol. 28, no. 7, pp. 3343–3356, Jul. 2019.
[24] L. Zhou, X. Song, J. Yao, L. Wang, and F. Chen, Convolutional Neural Network Filter (CNNF) for Intra Frame, document JVET-I0022, Gwangju, Jan. 2018.
[25] C. Ma, D. Liu, X. Peng, L. Li, and F. Wu, “Convolutional neural network-based arithmetic coding for HEVC intra-predicted residues,” IEEE Trans. Circuits Syst. Video Technol., vol. 30, no. 7, pp. 1901–1916, Jul. 2020.
[26] S. Puri, S. Lasserre, and P. Le Callet, “CNN-based transform index prediction in multiple transforms framework to assist entropy coding,” in Proc. 25th Eur. Signal Process. Conf. (EUSIPCO), Aug. 2017, pp. 798–802.
[27] E. W. Forgy, “Cluster analysis of multivariate data: Efficiency versus interpretability of classifications,” Biometrics, vol. 21, pp. 768–780, Jan. 1965.
[28] L. Zhao et al., “Wide angular intra prediction for versatile video coding,” IEEE Trans. Circuits Syst. Video Technol., vol. 28, no. 4, pp. 53–62, Apr. 2018.
[29] A. Said, X. Zhao, M. Karcewicz, J. Chen, and F. Zou, “Position dependent prediction combination for intra-frame video coding,” in Proc. IEEE Int. Conf. Image Process. (ICIP), Sep. 2016, pp. 534–538.
[30] B. Bross et al. Multiple Reference Line Intra Prediction, document JVET-L0283, Macao, Oct. 2018.
[31] M. Karcewicz and A. Alshina, JVET AHG Report: Tool Evaluation (AHG1), document JVET-G0001, Geneva, May 2016.
[32] J. Li, B. Li, J. Xu, and R. Xiong, “Efficient multiple-line-based intra prediction for HEVC,” IEEE Trans. Circuits Syst. Video Technol., vol. 28, no. 4, pp. 947–957, Apr. 2018.
[33] A. L. Maas, A. Y. Hannun, and A. Y. Ng, “Rectifier nonlinearities improve neural network acoustic models,” in Proc. Int. Conf. Mach. Learn. (ICML), 2013, pp. 1–6.
[34] Y. Linde, A. Buzo, and R. Gray, “An algorithm for vector quantizer design,” IEEE Commun. Vol. COM-28, no. 1, pp. 84–95, Jan. 1980.
D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in Proc. Int. Conf. Learn. Represent. (ICLR), 2015, pp. 1–15.

X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in Proc. 13th Int. Conf. Artif. Intell. Statist., 2010, pp. 249–256.

G. J. Sullivan and T. Wiegand, “Rate-distortion optimization for video compression,” IEEE Signal Process. Mag., vol. 15, no. 6, pp. 74–90, Nov. 1998.

S. De-Luxán-Hernández et al. CE3: Intra Sub-Partitions Coding Mode (Tests 1.1.1 and 1.1.2), document JVET-M0102, Marrakech, Jan. 2019.

D. Marpe, H. Schwarz, and T. Wiegand, “Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard,” IEEE Trans. Circuits Syst. Video Technol., vol. 13, no. 7, pp. 620–636, Jul. 2003.

A. Paszke et al., “PyTorch: An imperative style, high-performance deep learning library,” in Proc. Adv. Neural Inf. Process. Syst., 2019, pp. 8026–8037.

J. Boyce, K. Suehring, and X. Li. JVET Common Test Conditions and Software Reference Configurations, document JVET-J0100, Marrakech, Jan. 2019.

G. Bjøntegaard. Improvements of the BD-PSNR Mode, document ITU-T SG16 Q.6, VCEG-A111, Berlin, Jul. 2008.

O. Russakovsky et al., “ImageNet large scale visual recognition challenge,” Int. J. Comput. Vis., vol. 115, no. 3, pp. 211–252, Dec. 2015.

E. Agustsson and R. Timofte, “NTIRE 2017 challenge on single image super-resolution: Dataset and study,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), Jul. 2017, pp. 1122–1131.

J. Boyce, K. Suehring, and X. Li, JVET Common Test Conditions and Software Reference Configurations, document JVET-J0100, Marrakech, Jan. 2019.

Ruiqin Xiong (Senior Member, IEEE) 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. From 2002 to 2007, he was a Research Intern with Microsoft Research Asia. From 2007 to 2009, he was a Senior Research Associate with The University of New South Wales, Sydney, NSW, Australia. In 2010, he joined the School of Electronic Engineering and Computer Science, Peking University, Beijing, where he is currently a Professor. He has authored or coauthored more than 140 technical articles in refereed international journals and conferences. His research interests include image and video processing, statistical image modeling, deep learning, neuromorphic camera, and computational imaging. He was a recipient of the Best Student Paper Award from the SPIE Conference on Visual Communications and Image Processing in 2005 and the Best Paper Award from the IEEE Visual Communications and Image Processing in 2011. He was a co-recipient of the Best Student Paper Award from the IEEE Visual Communications and Image Processing in 2017.