2021 IEEE INTERNATIONAL WORKSHOP ON MACHINE LEARNING FOR SIGNAL PROCESSING, OCT. 25–28, 2021, GOLD COAST, AUSTRALIA

GRAPH-BASED TRANSFORM BASED ON NEURAL NETWORKS FOR INTRA-PREDICTION OF IMAGING DATA

Debaleena Roy, Tanaya Guha, and Victor Sanchez
debaleena.roy@warwick.ac.uk

Department of Computer Science, University of Warwick, Coventry, UK

ABSTRACT

This paper introduces a novel class of Graph-based Transform based on neural networks (GBT-NN) within the context of block-based predictive transform coding of imaging data. To reduce the signalling overhead required to reconstruct the data after transformation, the proposed GBT-NN predicts the graph information needed to compute the inverse transform via a neural network. Evaluation results on several video frames and medical images, in terms of the percentage of energy preserved by a sub-set of transform coefficients and the mean squared error of the reconstructed data, show that the GBT-NN can outperform the DCT and DST, which are widely used in modern video codecs.

Index Terms— intra-prediction, graph-based transform, template-based prediction, DCT, DST, KLT

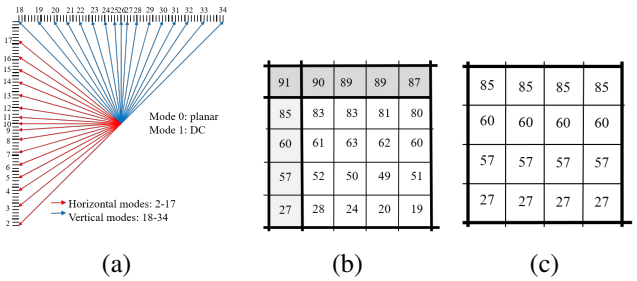
1. INTRODUCTION

Block-based predictive transform coding (PTC) [1, 2] is an integral part of modern video codecs such as the High Efficiency Video Coding (HEVC) [3] and the Versatile Video Coding (VVC) [4] standards. Intra-prediction is an important tool used by block-based PTC, where each video frame is divided into several non-overlapping blocks and processed in a block-wise manner. Specifically, each block is predicted based on the surrounding pixel values located immediately above and to the left by using one of several *intra-prediction modes*. These modes include several angular modes, a planar mode, and a DC mode. Each angular mode predicts a block using a specific direction to accurately model edges and directional patterns, while the planar and DC modes predict gradually-changing and smooth textures, respectively (see Fig. 1). A residual block is obtained for each block by computing the difference between the original and predicted block. Each residual block is then transformed, and the resulting transform coefficients are quantized and encoded to create a compressed bit-stream. To reconstruct the frame, the bit-stream is decoded, dequantized and inverse-transformed to recover the residual blocks. Each decompressed residual block is then added to the predicted block to recover the original block (with some losses due to the transformation and quantization - see Fig. 2). Within this compression pipeline, the transform reduces the correlation between the residual values, while quantization reduces the number of bits needed to store an integer value by reducing its precision [5]. Note that in order to reconstruct a frame, extra-information is needed to be signaled into the bitstream, which may lead to an increased overhead. This information includes the prediction mode used for each block, the block sizes, details of the inverse transform, and the level of quantization. To reduce this overhead, video codecs use well known transforms [6] so that the information needed to compute the inverse transform is common knowledge between the compression and reconstruction processes.

For any arbitrary signal with a known covariance matrix, it is well known that the Karhunen–Loève Transform (KLT) is the linear transform with the best energy compaction properties, i.e., it can represent most of the signal energy with only a few transform coefficients. The KLT is data-driven, as it depends on the data being transformed. To compute the inverse KLT, knowledge of the covariance matrix of each block must be signaled into the bitstream. The Discrete Cosine Transform (DCT) has been championed as the most suited transform for compression applications since the KLT basis functions of natural images are close to those of the DCT [7]. Unfortunately, the DCT offers little adaptability to the characteristics of the signal, as a fixed transform is usually applied to all residual blocks.

Recently, the Graph-Based Transform (GBT) has been shown to attain promising results [8] for data decorrelation and energy compaction for block-based PTC using intra-prediction. The GBT is quite adaptive to the signal since for each residual block a unique graph is generated to accurately reflect the correlation among residual values [9]. In [10, 6], we show that the GBT can outperform the DCT and the combination of DCT/DST (Discrete Sine Transform), as used in modern video codecs, in terms of energy compaction and reconstruction quality. The work in [11] shows an attractive solution for learning a mapping function to design a GBT without requiring to signal additional information. Pavez *et al*. [12] propose the Graph Template Transform (GTT) to ap-

978-1-7281-6338-3/21/$31.00 ©2021 IEEE



(a) (b) (c)

Fig. 1. (a) Direction of the 33 angular modes used in the HEVC standard. (b) Sample 4×4 block, where the surrounding pixels located above and to the left are used for prediction. (c) Corresponding predicted block using the pure horizontal mode (Mode 10).

proximate the KLT by using a priori information about signals represented by a graph template. In a recent work, Egilmez *et al*. propose a GBT based on Gaussian-Markov Random Field (GMRF) models that learn graphs from data and provide solutions for optimal separable and non-separable GBTs [13].

When the GBT is used in block-based PTC, the same graph used to compute the GBT during compression should be available at the reconstruction stage to compute the inverse GBT of each block. This extra information should be then signaled into the bitstream, hence increasing the overhead. To address this issue, this paper proposes a GBT based on a neural network approach (GBT-NN). Our proposed method uses an encoding-decoding neural network (NN) to map a graph obtained from a set of similar blocks to the block to be encoded, to the graph of the corresponding residual block. Specifically, our method adopts a template-based strategy to first predict a residual block from a set of similar blocks, from which a graph can be computed. The corresponding graph Laplacian of such a graph is then used by a NN to predict the graph Laplacian associated with the current residual block, from which the inverse transform can be computed. To avoid signalling extra information into the bitstream, the template-based strategy is replicated during reconstruction to compute the same graph Laplacian and hence the inverse GBT. To the best of our knowledge, no method has been proposed before to learn a graph Laplacian by using deep learning and a template-based strategy within the context of block-based PTC and GBTs.

Our evaluations over several video frames and medical images show that the proposed GBT-NN outperforms the DCT/DST, DCT and other related GBTs [10] in terms of percentage of preserved energy (PE), mean squared error (MSE), and Peak Signal-to-Noise Ratio (PSNR).

2. PROPOSED GBT-NN

Let us denote a (square) residual block with zero mean as $\mathbf{S} \in \mathbb{R}^{\sqrt{N} \times \sqrt{N}}$, with a total of N residual values. Recall that $\mathbf{S}$ is computed by subtracting the predicted block from

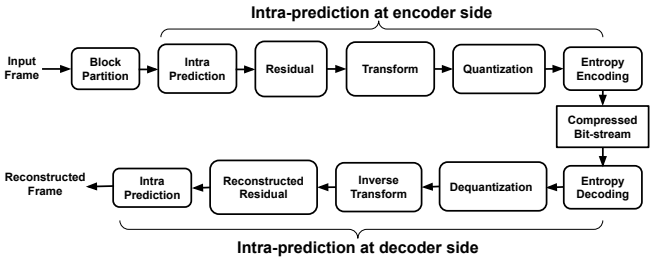
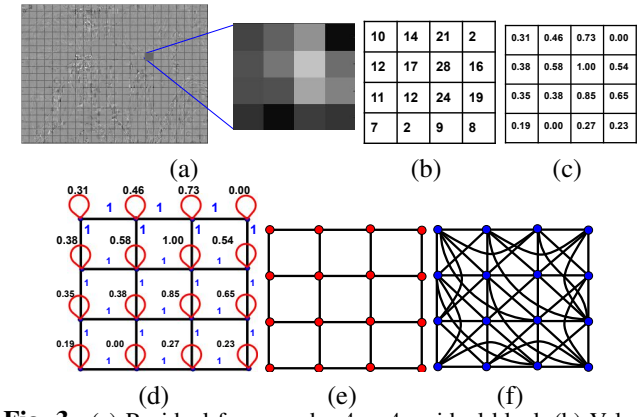


Fig. 2. Video compression/decompression pipeline used by the HEVC and VVC standards for block-based intra-prediction.



(a) (b) (c)

(d) (e) (f)

Fig. 3. (a) Residual frame and a 4×4 residual block (b) Values of the example residual block. (c) Normalized residual values to the range $[0, 1]$. (d) Corresponding graph with a 4-connected topology with unit edge weights and self-loops in each vertex. (e) 4-connected topology with no self-loops. (f) All-connected topology with no self-loops (i.e., each node is connected to every node in the graph)

the original block [14]. $\mathbf{S}$ can be represented as an undirected weighted graph, $G = (V, E, \mathbf{A})$, where V is the set of N nodes $V = \{v_n\}_{n=1}^{N}$, E is the set of edges, and $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the symmetric adjacency matrix. The adjacency matrix of a weighted graph stores the weights of the edges. The GBT for $\mathbf{S}$ can be computed by the eigen decomposition of the graph Laplacian, $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D}$ is the diagonal degree matrix. The eigen decomposition of $\mathbf{L}$ can be used as an orthogonal transform for $\mathbf{S}$, since it has a complete set of eigenvectors with real, non-negative eigenvalues [15]. The connectivity and the edge weights of the graph are generally inferred from the data (see Fig. 3).

As the graph Laplacian requires the computation of the symmetric adjacency matrix, our objective is to develop a one-to-one mapping between symmetric adjacency matrices: one computed based on previously encoded and reconstructed blocks within the same frame and the other one associated with the current block:

$$\mathbf{A}^b \approx f(\mathbf{A}^p), \tag{1}$$

where $\mathbf{A}^p$ is computed based on the graph of a residual block predicted for the current residual block and $\mathbf{A}^b$ is the symmetric adjacency matrix of the current residual block, i.e., the block to be encoded. Our solution to learn the mapping function in Eq. (1) is based on an encoding-decoding NN, as illus-

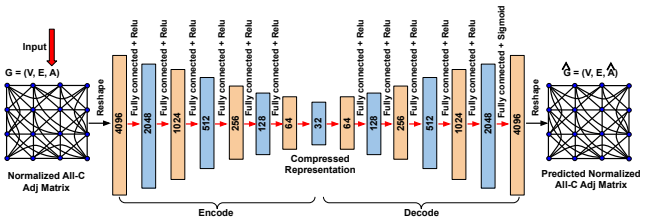


Fig. 4. Architecture of the proposed GBT-NN for 8×8 blocks and a normalized all-connected (All-C) symmetric adjacency matrix.

trated in Fig. 4 for the case of 8×8 blocks with all-connected graphs. The encoder consist of 4096 input neurons and 7 fully connected hidden layers, while the decoder consists of 6 fully connected hidden layers and an output layer. For each hidden layer, we apply the *ReLu* activation function, while the *Sigmoid* activation function is applied to the output layer. Note that the architecture in Fig. 4 is also suitable for graphs with other topologies, e.g., 4-connected with self-loops. Also note that the input is normalized to the range $[0, 1]$.

2.1. Prediction strategy

The matrix used as input to the network is generated from a residual block predicted by a template-based prediction strategy [10]. Such a strategy uses a weighted average of all the previously encoded and reconstructed blocks within a specific region of the same frame to predict the current block (see Fig. 5 (a)). The weight assigned to the j^{th} reconstructed block is:

$$w_j = e^{\frac{\|\mathbf{x}-\mathbf{t}_j\|_2^2}{h^2}}, \tag{2}$$

where $\mathbf{t}_j$ is the j^{th} reconstructed template, h is the average of the standard deviation of the pixel values of the $j-1$ reconstructed templates, and $\mathbf{x}$ contains the pixel values of the target template. We use templates of 4 rows and 4 columns, which results in 72 samples surrounding an 8×8 block to the left and above (see Fig. 5(b)).

The predicted current block is subtracted from the corresponding predicted block computed by intra-prediction to compute a predicted residual block (see Fig. 5(c)). From this predicted residual block, the normalized symmetric adjacency matrix, $\mathbf{A}^p$, is computed, vectorized and normalized into $\mathbf{a}^p$. The encoder NN transforms $\mathbf{a}^p$ into a hidden representation, $\mathbf{h}$ as follows:

$$\mathbf{h}^{(l_e)} = ReLU(\mathbf{W}^{(l_e)}\mathbf{h}^{(l_e-1)}), \tag{3}$$

where $\mathbf{h}^{(0)} = \mathbf{a}^p$, $\mathbf{W}^{(l_e)}$ is a weight matrix and $\mathbf{h}^{(l_e)}$ is the hidden representation for the encoder layer (l_e). Then, $\mathbf{h}$ is transformed back to a reconstructed vector $\hat{\mathbf{a}}^b$ by the decoder NN over a number of hidden layers until the output layer:

$$\hat{\mathbf{a}}^b = Sigmoid(\mathbf{W}^{(l_d)}\mathbf{h}^{(l_d-1)}), \tag{4}$$

where l_d denotes the last layer of the decoder, $\mathbf{W}^{(l_d)}$ is a weight matrix for the decoder layer l_d, and $\mathbf{h}^{(l_d-1)}$ is the hidden representation of decoder layer (l_d-1). Note that $\hat{\mathbf{a}}^b$ is an approximation of the vectorized and normalized symmetric adjacency matrix of the current residual block. Also note that the architecture in Fig. 4 differs from that of an autoencoder, as our NN does not reconstruct the same input.

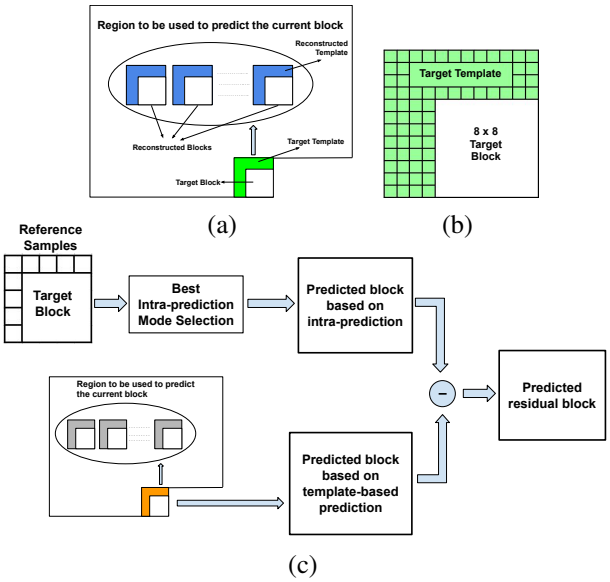


Fig. 5. (a) Region to be used to predict the target block; i.e, the current block. (b) Sample target template and target block. (c) Template-based prediction (TBP).

2.2. Optimization process

Optimization of the NN aims to find the parameters $\mathbf{W}^{(1_e)}, \cdots, \mathbf{W}^{(l_e)}, \mathbf{W}^{(1_d)}, \cdots, \mathbf{W}^{(l_d)}$ that minimize following loss function:

$$L = L_{recon} + \alpha L_{sym} + \lambda \parallel \mathbf{W}(:) \parallel_1, \tag{5}$$

where $\| \, . \, \|$ is the $L1$ matrix norm, $\mathbf{W}(:)$ represents the learnable parameters in vector form, α is the weight of the second loss component, and λ controls the amount of $L1$ regularization on the learnable parameters. Here L_{recon}, L_{sym} are the losses for reconstruction and symmetry, respectively. We use the MSE for the reconstruction loss:

$$L_{recon} = \parallel \hat{\mathbf{a}}^b - \mathbf{a}^b \parallel_2^2, \tag{6}$$

where $\mathbf{a}^b$ is the vectorized ground truth for the normalized symmetric adjacency matrix of the current residual block.

An essential property of an adjacency matrix is to be symmetric. We then use the following loss to enforce symmetry:

$$L_{sym} = \frac{\parallel \hat{\mathbf{a}}^{b(anti)} \parallel_1}{\parallel \hat{\mathbf{a}}^{b(sym)} \parallel_1 + \parallel \hat{\mathbf{a}}^{b(anti)} \parallel_1}, \tag{7}$$

where $\hat{\mathbf{a}}^{b(sym)} = (\hat{\mathbf{a}}^b + (\hat{\mathbf{a}}^b)^T)/2$ and $\hat{\mathbf{a}}^{b(anti)} = (\hat{\mathbf{a}}^b - (\hat{\mathbf{a}}^b)^T)/2$, i.e., they measure the symmetry and anti-symmetry of the predicted matrix. $L_{sym} \in [0, 1]$, which tends to the upper bound for a symmetric matrix and to the lower bound for an asymmetric matrix. The graph used to compute the GBT for the current residual block is then $\hat{G} = (V, E, \hat{\mathbf{A}}^b)$, after de-normalizing the predicted symmetric adjacency matrix.

To reconstruct the current block, the same graph used to compute the GBT should be used to compute the inverse GBT.

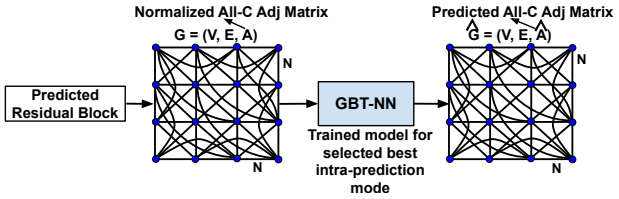


Fig. 6. GBT-NN used to produce a predicted symmetric adjacency matrix for the current residual block.

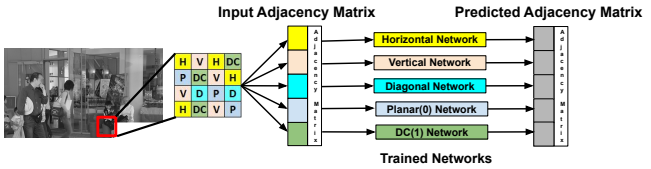


Fig. 7. A specific trained network is selected based on the intra-prediction mode used for each block. The figure shows a section of a frame predicted by several prediction modes.

To this end, the template-based prediction strategy described in Section 2.1 is also used to predict the residual block of the current block during reconstruction. The predicted residual block is used to compute the symmetric adjacency matrix to be used as the input to the trained GBT-NN after normalization, which produces a predicted symmetric adjacency matrix for the current residual block (see Fig. 6). Our method assumes that the trained GBT-NN is common knowledge between the compression and reconstruction processes. Therefore, our method does not require to signal any extra information. Based on the prediction mode used, the reconstruction process uses a specific trained GBT-NN associated with that mode. Fig. 7 explains this mechanism assuming an HEVC codec. Namely, our framework relies on five trained GBT-NNs: one for horizontal (H) modes, one for vertical (V) modes, one for diagonal (D) modes, one for the DC mode, and one for the planar (P) mode.

3. PERFORMANCE EVALUATION

We train 5 different networks (H, V, D, DC, P) based on the 35 HEVC intra-prediction modes. We use 8×8 blocks and graphs with an all-connected (All-C) topology with no self loops with unit edge (UE) weights. Each training example is represented by a tuple: $\{\mathbf{A}^p, \mathbf{S}, \mathbf{A}^b\}$, where $\mathbf{A}^p$ is the predicted symmetric adjacency matrix for residual block $\mathbf{S}$ (as computed by the template-based prediction strategy) and $\mathbf{A}^b$ is the ground truth symmetric adjacency matrix for $\mathbf{S}$. The networks are trained only with $\{\mathbf{A}^p\}$ and $\{\mathbf{A}^b\}$ with $\alpha = 0.5$ and $\lambda = 0.002$ (see Eq. 5). The hyper-parameters are selected based on cross-validation. We train each network for 100 epochs using Adam optimizer with a learning rate $= 0.0001$.

We use 40 different gray level YUV frames of Class A, B, C, D, E and Screen Content, which are video sequences widely used to test modern video codecs [3]. We also use the green (G) component of 10 color pathology images from the Center for Biomedical Informatics and Information Technology of the US National Cancer Institute [16]. In total, for the five networks, we use $61,440$ samples of symmetric adjacency matrices. We use 80% of the data for training and 20% for testing. There is no overlap in the training and testing sets.

Table 1. Characteristics of the evaluated GBTs.

Name	Topology		Edge Weight			Residual	
	4-C	All-C	DIFF	UE	SL	IP	TBP
GBT$_A$-All*		✓	✓			✓	
GBT-L$_A$-All*		✓		✓	✓	✓	
GBT$_{Wpix}$-All		✓	✓				✓
GBT-NN (ours)		✓		✓			✓
GBT-L$_{Wpix}$-All		✓		✓	✓		✓
GBT-L$^*_A$	✓			✓	✓	✓	
GBT-L$_{Wpix}$	✓			✓	✓		✓

4-C: 4-connected. **All-C**: all-connected.
DIFF: edge weights computed as the squared difference between the connected nodes.
UE: unit edge weights. **SL**: self-loops with normalized weights.
IP: actual residual block used (no prediction). **TBP**: residual block predicted by template-based prediction.
* Requires information about the graph of each block to compute the inverse transform for reconstruction.

Table 2. Performance of the NNs on the test data.

NN	Loss function	Metrics		
		MSE	MAE	Ψ
Horizontal	L	165.95	4.66	0.99
	L_{rcon} (only)	271.10	5.29	0.94
Vertical	L	170.4	4.35	0.97
	L_{rcon} (only)	258.59	6.43	0.93
Diagonal	L	169.65	5.44	0.92
	L_{rcon} (only)	234.98	6.91	0.82
DC	L	184.28	8.40	0.95
	L_{rcon} (only)	284.16	15.4	0.88
Planar	L	156.52	5.21	0.98
	L_{rcon} (only)	258.5	6.99	0.93

Table 1 summarizes the characteristics of all the GBTs we use in the evaluations. Namely, it tabulates the topology used to construct the graph, the edge weights, and how the residual for the current block is computed. We also evaluate the KLT, the DCT, and DCT/DST as used in the HEVC and VVC standards, where the DCT/DST are used as separable transforms for rows and columns of the residual block depending on the prediction mode used. We use the MSE and Mean-absolute-error (MAE) to measure how well the values of the symmetric adjacency matrix are predicted compared to the ground truth. We measure the symmetrical property of the predicted matrices as [17, 18]:

$$\Psi = \frac{|\hat{\mathbf{a}}^{b(sym)}\|_1 - \|\hat{\mathbf{a}}^{b(anti)}\|_1}{\|\hat{\mathbf{a}}^{b(sym)}\|_1 + \|\hat{\mathbf{a}}^{b(anti)}\|_1} \in [-1, 1], \qquad (8)$$

where a value of 1 means perfect symmetry.

Table 2 tabulates the performance of the five trained GBT-NNs on the test data. We perform an ablation study by removing the L_{sym} component of the loss function. This table shows that L_{sym} is vital to enhance the performance of the networks since the MSE and MAE values increase and Ψ values decrease if L_{sym} is removed.

Since the efficiency of a transform is measured by its

decorrelating properties and the maximum energy it concentrates in only a few transform coefficients, we compute the percentage of PE and the MSE of the reconstructed frames/images using only a few coefficients under the assumption that no quantization is applied. The sub-set of coefficients used for reconstruction is selected by setting a threshold that indicates the minimum absolute value that the coefficients in the sub-set should have. By gradually decreasing an initial large threshold, this approach gradually includes in the sub-set the largest coefficients [10]. We also compute the reconstruction quality achieved by the evaluated transforms, in terms of the PSNR, when there is quantization. To this end, we use the 4 quantization parameters (QPs) (22, 27, 32, 37) widely used by the HEVC and VVC standards.

3.1. Results

Table 3 presents the average PE (%) and MSE values for all evaluated data using a small percentage of coefficients. As expected, the KLT provides the best performance. The GBT-NN preserves 10.46%, 6.37%, and 5.42% more energy than the DCT/DST, the DCT, and the GBT-L$_{Wpix}$ [10], respectively, if only 5% of the largest coefficients are used. We observe that the GBT-L$_A$ outperforms our proposed GBT-NN, however, as the GBT-L$_A$ requires information about the graph to compute the inverse transform needed to reconstruct each block, this transform is not practical as this entails greatly increasing the overhead. Fig. 8 plots the PE (%) and MSE values vs. the percentage of coefficients used for reconstruction of a frame of sequence *BQTerrace* (Class B) and *PeopleOnStreet* (Class A). Table 4 tabulates average PSNR values for the evaluated frames/images when 4 different QPs are applied to the transform coefficients. Note that the proposed GBT-NN outperforms both the DCT and DCT/DST. Fig. 9 plots the PSNR values for a frame of the *ChinaSpeed* (Class Screen Content) and *BlowingBubbles* (Class D) sequences. Fig. 10 shows a reconstructed frame of the sequence *RaceHorse* (Class D) after transformation by the KLT, DCT and our proposed GBT-NN, and quantization with QP= 37. As depicted, the GBT-NN achieves a higher visual reconstruction quality than the DCT.

3.2. Computational complexity

Any GBT involves eigendecomposition of the graph Laplacian. Hence, the GBT is as computationally complex as the KLT. However, the GBT-NN does not need to signal any extra information for reconstruction thanks to the template-based prediction strategy and the trained NNs. For any fully connected layer l, the number of learnable parameters, i.e., the size of matrix $\mathbf{W}^{(l)}$ is $k \times d$, where $\{d, k\}$ are the number of input and output neurons, respectively. Once the networks are trained offline, the learned weights are assumed to be common knowledge between the transformation and reconstruction stages.

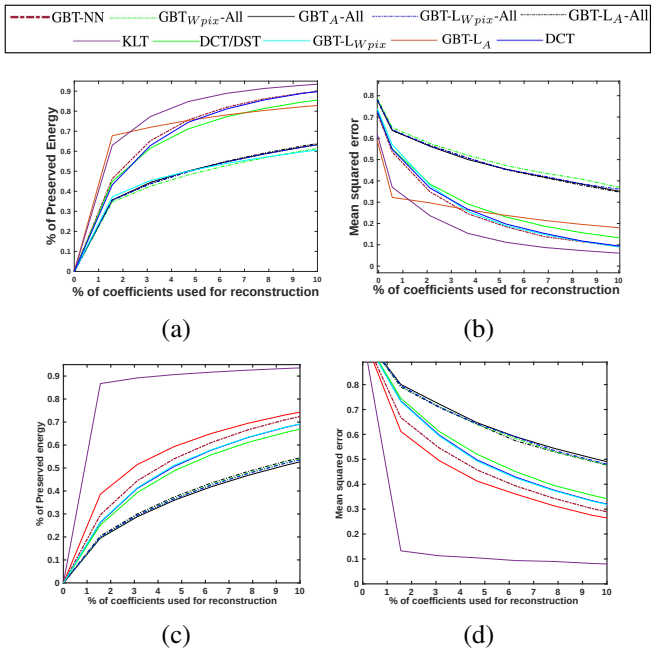


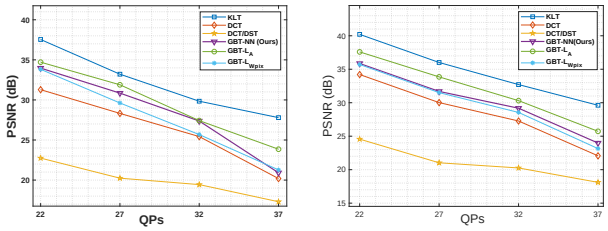
Fig. 8. (a,c) PE (%) and (b,d) MSE vs. percentage of coefficients used for reconstruction of (1^{st} row) a frame of sequence *PeopleOnStreet* (Class A) and (2^{nd} row) a frame of sequence *BQTerrace* (Class B).

Table 3. Average PE (in %) and MSE using a small percentage of the largest coefficients.

	Percentage of coefficients used					
	1%		5%		10%	
	PE	MSE	PE	MSE	PE	MSE
KLT	55.51	44.49	90.85	10.22	93.56	07.97
DCT	16.84	82.99	52.11	48.38	69.07	32.03
DCT/DST	16.14	83.69	50.18	50.64	66.88	34.24
GBT$_A$-All	12.47	87.23	37.14	63.57	52.66	49.04
GBT-L$_A$-All	13.07	83.69	38.62	50.64	54.42	34.24
GBT$_{Wpix}$-All	12.71	86.78	38.00	62.85	53.86	47.79
GBT-NN (ours)	**18.97**	**78.72**	**55.43**	**44.46**	**72.40**	**28.94**
GBT-L$_{Wpix}$-All	12.79	86.91	37.77	63.21	53.58	48.05
GBT-L$_A$	24.71	75.17	60.47	40.21	74.28	26.47
GBT-L$_{Wpix}$	17.01	82.82	52.58	47.93	69.18	31.86

4. CONCLUSIONS

In this paper, we proposed the GBT-NN, a new class of GBTs that performs efficiently in block-based PTC with intra-prediction. The GBT-NN is based on a deep encoding-decoding NN that learns a mapping function to approximate a symmetric adjacency matrix associated with the graph of the residual block to be encoded. Moreover, thanks to a template-based prediction strategy, the GBT-NN does not require to explicitly compute the graph Laplacian for each residual block during reconstruction. We evaluate the performance of the GBT-NN in terms of the PE (%) and MSE when a small percentage of the largest coefficients are used for reconstruction, as well as in terms of the PSNR when different quantization levels are applied to the transform coefficients. Evaluation results show that the proposed GBT-NN outperforms DCT and DCT/DST, which are widely used by modern video codecs.

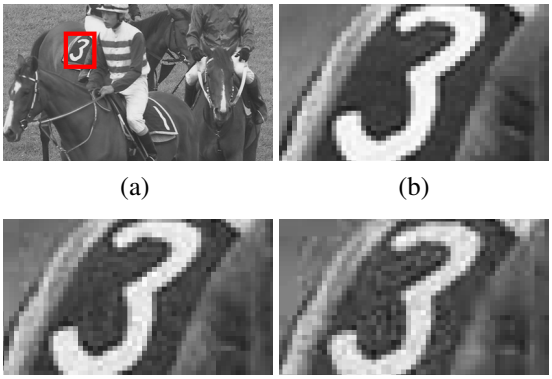


(a) (b)

Fig. 9. PSNR vs. QP for a frame of (a) sequence *ChinaSpeed* (Class Screen Content) and (b) sequence *BlowingBubbles* (Class D).

Table 4. Average reconstruction PSNR values when using quantization on the transform coefficients.

	Quantization Parameters			
	QP=22	QP=27	QP=32	QP=37
KLT	40.22	36.05	32.71	29.62
DCT	35.21	31.02	28.29	23.07
DCT/DST	20.56	19.02	18.25	17.10
GBT$_A$-All	8.62	7.65	6.55	6.96
GBT-L$_A$-All	16.38	15.47	14.73	12.73
GBT$_{W_{pix}}$-All	10.14	9.83	9.11	8.56
GBT-NN (ours)	**35.86**	**31.69**	**29.18**	**23.93**
GBT-L$_{W_{pix}}$-All	11.25	11.49	9.18	8.89
GBT-L$_A$	36.69	33.84	30.31	25.73
GBT-L$_{W_{pix}}$	35.71	31.58	28.55	23.16



(a) (b)

(c) (d)

Fig. 10. (a) An original frame of sequence *RaceHorse* (Class D). (b) An area reconstructed after using the KLT (PSNR = 28.45 dB), (c) the proposed GBT-NN (PSNR = 23.92 dB), and (d) the DCT (PSNR = 22.67). In all cases, QP=37.

5. REFERENCES

[1] J. Lainema, F. Bossen, W.J. Han, J. Min, and K Ugur, "Intra coding of the hevc standard," *IEEE Trans. Circuits and Systems for Video Tech*, vol. 22, no. 12, pp. 1792–1801, 2012.

[2] W. Hu, G. Cheung, and A. Ortega, "Intra-prediction and generalized graph fourier transform for image coding," *IEEE Signal Processing Letters*, vol. 22, no. 11, pp. 1913–1917, 2015.

[3] J.R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards—including high efficiency video coding (hevc)," *IEEE Trans. Circuits and Systems for Video Tech*, vol. 22, no. 12, pp. 1669–1684, 2012.

[4] B. Bross, J. Chen, J.R. Ohm, G. J. Sullivan, and Y. Wang, "Developments in international video coding standardization after avc, with an overview of versatile video coding (vvc)," *Proceedings of the IEEE*, pp. 1–31, 2021.

[5] A. Said and W. A. Pearlman, "An image multiresolution representation for lossless and lossy compression," *IEEE Transactions on Image Processing*, vol. 5, no. 9, pp. 1303–1310, 1996.

[6] D. Roy and V. Sanchez, "Graph-based transforms based on prediction inaccuracy modeling for pathology image coding," in *Data Compression Conference*, 2018, pp. 157–166.

[7] A. Saxena and F. C. Fernandes, "Dct/dst-based transform coding for intra prediction in image/video coding," *IEEE Trans. Image Processing*, vol. 22, no. 10, pp. 3974–3981, 2013.

[8] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, 2013.

[9] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Transactions on Signal Processing*, vol. 61, no. 7, pp. 1644–1656, 2013.

[10] D. Roy, T. Guha, and V. Sanchez, "Graph-based transform with weighted self-loops for predictive transform coding based on template matching," in *2019 Data Compression Conference (DCC)*, 2019, pp. 329–338.

[11] D. Roy, T. Guha, and V. Sanchez, "Graph based transforms based on graph neural networks for predictive transform coding," in *2021 Data Compression Conference (DCC)*, 2021, pp. 367–367.

[12] E. Pavez, H. E. Egilmez, Y. Wang, and A. Ortega, "Gtt: Graph template transforms with applications to image coding," in *2015 Picture Coding Symposium (PCS)*, 2015, pp. 199–203.

[13] H. E. Egilmez, Y. H. Chao, and A. Ortega, "Graph-based transforms for video coding," *IEEE Transactions on Image Processing*, vol. 29, pp. 9330–9344, 2020.

[14] J. Lainema, F. Bossen, W. Han, J. Min, and K. Ugur, "Intra coding of the hevc standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1792–1801, 2012.

[15] S. Boyd, "Convex optimization of graph laplacian eigenvalues," *In Proc.Int. Congr. Math., volume 3*, vol. 3, no. 1, pp. 1311–1319, 2006.

[16] K. Tomczak, P. Czerwińska, and M. Wiznerowicz, "The cancer genome atlas (tcga): an immeasurable source of knowledge," *Contemporary oncology*, vol. 19, no. 1A, pp. A68, 2015.

[17] "Metric for how symmetric a matrix is," https://math.stackexchange.com/questions/2048817/metric-for-how-symmetric-a-matrix-is, [Online].

[18] Y. Li and Z. Zhang, "Digraph laplacian and the degree of asymmetry," *Internet Mathematics*, vol. 8, no. 4, pp. 381–401, 2012.