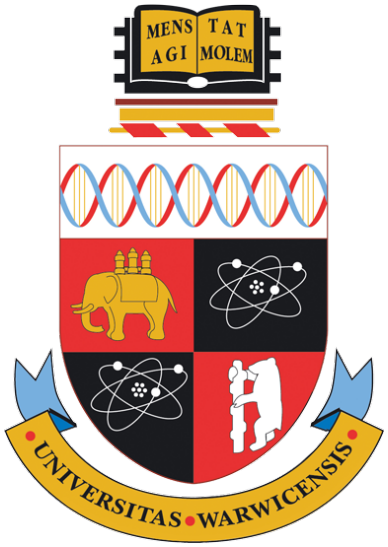
WARWICK
THE UNIVERSITY OF WARWICK



Graph Based Transforms for Block-based Predictive Transform Coding

by

Debaleena Roy

Thesis

Submitted to the University of Warwick

in partial fulfilment of the requirements

for admission to the degree of

Doctor of Philosophy in Computer Science

Department of Computer Science

January 2023

Contents

List of Tables

List of Figures

Acknowledgments

First and foremost I am extremely grateful to my supervisors, Dr. Victor Sanchez and Dr. Tanaya Guha for their invaluable advice, continuous support, and patience during my PhD study. Their immense knowledge and plentiful experience have encouraged me in all the time of my academic research. I am fortunate to work with such great supervisors.

I would also like to thank the SIPLAB team at Warwick University, whose insight and knowledge into the subject matter steered me through this research.

Finally, I would like to thank my husband and my daughter, Dhiraj Poddar and Pritisha for all the support they have shown me through this research. I owe my achievements to their love and encouragement.

Declarations

I hereby declare that this dissertation is original work and has not been submitted for a degree or diploma or other qualification at any other University. Parts of this thesis have been previously published by the author in the following:

1 Publications

Parts of this thesis have been previously published by the author in the following:

[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] 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

[8] 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

[9] ——, "Graph-based transform based on neural networks for intra-prediction of imaging data," in *2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP)*, 2021, pp. 1–6

[10] ——, "Graph-based transform based on 3d convolutional neural network for intra-prediction of imaging data," in *2022 Data Compression Conference (DCC)*, 2022, pp. 212–221

2 Sponsorships and Grants

This research was funded by Department of Computer Science, University of Warwick, UK.

A part of research is funded by IDENTITY Project, University of Warwick, UK.

Abstract

Orthogonal transforms are the key aspects of the encoding and decoding process in many state-of-the-art compression systems. The transforms in block-based predictive transform coding (PTC) is essential for improving coding performance, as it allows decorrelating the signal in the form of transform coefficients. Recently, the Graph-Based Transform (GBT), has been shown to attain promising results for data decorrelation and energy compaction especially for block-based PTC. However, in order to reconstruct a frame for GBT using block-based PTC, extra-information is needed to be signalled into the bitstream, which may lead to an increased overhead. Additionally, the same graph should be available at the reconstruction stage to compute the inverse GBT of each block.

In this thesis, we propose a set of a novel class of GBTs to enhance the performance of transform. These GBTs adopt several methods to address the issue of the availability of the same graph at the decoder while reconstructing video frames. Our methods to predict the graph can be categorized in two types: non-learning-based approaches and deep learning (DL) based prediction. For the first type our method uses reference samples and template-based strategies for reconstructing the same graph. For our next strategies we learn the graphs so that the information needed to compute the inverse transform is common knowledge between the compression and reconstruction processes. Finally, we train our model online to avoid the amount, quality, and relevance of the training data.

Our evaluation is based on all the possible classes of HEVC videos, consist of class A to F/Screen content based on their varied resolution and characteristics. Our experimental results show that the proposed transforms outperforms the other non-trainable transforms, such as DCT and DCT/DST, which are commonly employed in current video codecs in terms of compression and reconstruction quality.

Acronyms

3D-CNN 3D Convolutional Neural Network.

CAE Convolutional Autoencoder.

DCT Discrete Cosine Transform.

DFT Discrete Fourier Transform.

DL Deep Learning.

DSP Digital Signal Processing.

DSPG DSP on graphs.

DST Discrete Sine Transform.

GBST Graph-Based Separable Transform.

GBT Graph-Based Transforms.

GBT-ONL Online Learning of graphs for GBT.

GBT-NN GBT based on Neural Networks.

GBT-CNN GBT based on 3D Convolutional Neural Networks.

GBT-L GBT with Weighted Self-Loops.

GMRF Gaussian Markov Random Field.

HEVC High Efficiency Video Coding.

IGBT Inverse Graph-Based Transform.

ILC Invertible Lossy Compression.

JPEG Joint Photographic Experts Group.

KLT Karhunen Loéve Transform.

ML Machine Learning.

MSE Mean Squared Error.

NN Neural Network.

PCA Principal Components Analysis.

PI Prediction Inaccuracy (Modelling).

PSNR Peak Signal-to- Noise Ratio.

PTC Predictive Transform Coding.

SC Screen Content.

VVC Versatile Video Coding.

WSI Whole Slide (pathology) Images.

Chapter 1

Introduction

Since early age tremendous effort has been rendered on improvement of imaging technology and as a consequence in 90s revolutionary progress has been achieved the way we capture our movement and activities, interact with each other and represent our surroundings. Latest imaging technologies, however, require acquiring large amounts of data, which impacts on storage and communication infrastructures. Therefore, the improvement in compression techniques became essential to cope up with the increasing data sizes.

Digital signal processing (DSP) has been a successful pillar for many compression methodologies. In video and image compression standard of High Efficiency Video Coding (HEVC) [11], intra prediction process shares several high performance coding. A particularly important variation of the problem of image compression is to achieve an optimal size of compressed images without affecting the accuracy. HEVC has been used for lossless compression. On the other hand, to achieve an optimal size of compressed images, lossy compression is desirable. Lossy compression methods are based on the principle of expanding a signal into orthonormal bases using an orthogonal transform, with the expectation that most information is captured by a few basis functions. For a random signal with a known covariance function, it is well known that the Karhunen Loéve transform (KLT) [12] is the linear transform with the best energy compaction property. The KLT basis functions of typical natural images are close to the Discrete Cosine Transform (DCT), thus the DCT has been championed as the best suited transform for compression applications. Unfortunately, the DCT offers little adaptability to the characteristics of the signal, as a fixed transform is usually applied to all signals.

The increasing data sizes associated with new imaging technologies have encouraged new ways to improve compression methods. This has driven the emerging field of DSP on graphs (DSPG), which aims at extending to the graph domain the generalized operators and localized, multi-scale transforms defined for discrete scalar functions on regular Euclidean spaces [13, 14]. Although

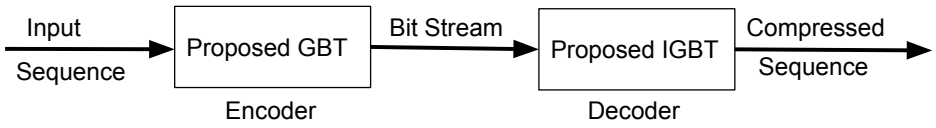


Figure 1.1: Principle block diagram of the research.

images (and frames of videos) are 2D regular signals, they can be formulated as graphs by connecting every pixel (node) with its neighboring pixels (nodes), and by interpreting pixel values as the values of the graph signal at each node. Representing signals as graph allows exploiting their underlying structure, which is in general not possible following conventional DSP methods. The Graph-Based Transforms (GBT) has been recently shown to attain promising results for data de-correlation and energy compaction. This comes as an intrinsic consequence of the underlying graph structure, which can accurately reflect the correlation among pixels. In general, there are two variants of GBTs. The first one is constructed based on the specific graph representing the signal to be transformed. This variant accurately reflects the characteristics of the signal, but may require signalling additional information so the decoder can reconstruct the graph [15]. The second variant consists in using separable transforms that can be applied to rows and columns of a matrix of signal values (i.e. a block pixel) [16]. This may require understanding the characteristics of the data from training data, but requires no additional information to be signalled to the decoder.

In this research, a novel next generation of graph-based compression methods for the latest imaging data is proposed. This research contribute to the realization of a predictive transform coding (PTC) framework based on DSPG, as schematically summarized in Fig. 1.1, where imaging data are predicted and coded, on a block-by-block basis, by using graph-based motion estimation, graph-based transformation (i.e. using GBTs). This research aims at designing effective and low-complexity GBTs by determining the most appropriate graph construction for different imaging data. This research has the potential to impact compression methods for both, new image formats, such as light-field images, and new video formats, such as Ultra high definition (UHD), high dynamic range (HDR), screen-content (SC) and free viewpoint videos.

1.1 Motivation behind this research

Making progress in representing an image as graphs became popular for discrete and mathematically simple representation that lends itself well to the development of efficient and provably correct methods. Graph is always a minimalistic image representation. Also, graphs are flexibility in representing

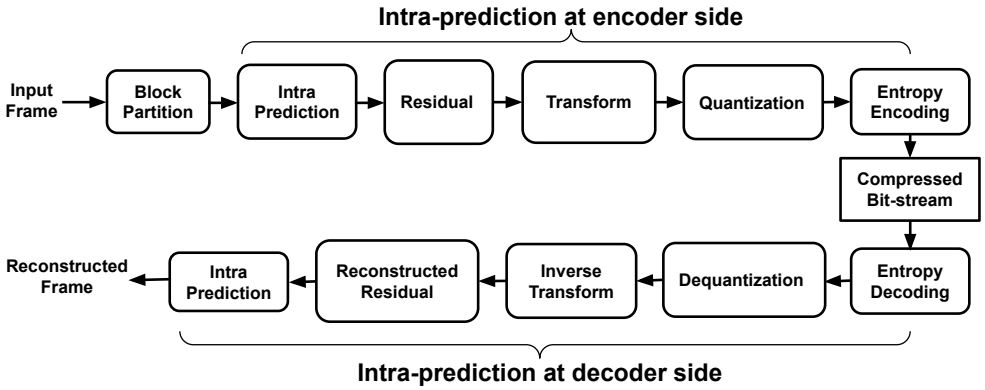


Figure 1.2: Video compression/decompression pipeline used by the HEVC and VVC standards for block-based intra-prediction.

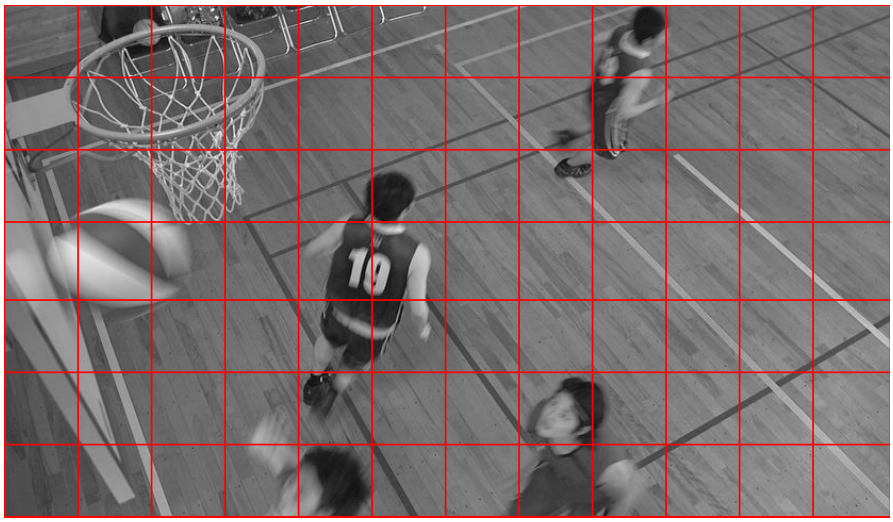


Figure 1.3: Partition of a frame into non-overlapping blocks.

different types of images. Further, a lot of work has been done on graph theory in other applications. As a consequence, the probability of re-use of existing algorithms and theorems developed for other fields in image analysis has been increased. In literature since long time an image was proposed to represent as a graph being a real-valued, non-negative function of two real variables and the value of this function at a point will be called the gray-level of the image at the point [17]. The advantages of adopting graph-based image representation are enormous, even in the modern era. The real driving force behind this research is to use GBT to circumvent the compression domain by utilising representing an image as a graph concepts. Consequently, our research explored the existing GBT for coding purpose since 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. In literature, block-based predictive PTC [11, 18] is an integral part of modern video codecs such as the HEVC [19] and the Versatile Video Coding (VVC) [20] 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 (see Fig. 1.3). 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 . 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. 1.2). However, in literature we found a *gap* where the reconstruction at decoder side needs to have the same graph information to perform GBT. More precisely, at the decoder 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. In this research we focus to address the gap to reduce this overhead with novel ideas so that the information needed to compute the inverse transform can be used as a common knowledge between the compression and reconstruction processes. As an instance, all of our methods used reference samples to avoid sending the information of the block to be encoded at the decoder.

Our research includes the luma component of modern video frames of HEVC standard (see Fig. 1.4, Fig. 1.5). Apart from the HEVC test sequences our research is applied on Whole Slide (WSI) Pathology images (see Fig. 1.6).

1.2 Research objective

In recent days GBTs shows promising results in terms of energy compaction and reconstruction. However, still it can not reach the level of KLT which is well known as an optimal transformation. Consequently, it is always encouraged to develop transformations which avoid the complexity of calculating eigenvector from covariance function for residual signal. Also, in both the cases of KLT and GBT, it require additional signalling to the decoder to reconstruct the image or video sequence which reduces the gain of compression. Thus the overall objective of this research is to develop a high-end compression technique for image and video sequences by exploiting the field of graph signal processing

and machine learning algorithms.

1.2.1 Research questions and approaches

Our research question is two-fold:

- Our first question is how this research helps to avoid overheads of sending additional information to the decoder to reconstruct the sequence? More precisely, we want to find ways to use the GBT without having to send extra information to the decoder to be able to reconstruct the transform coefficients.
- Our second research question is how to develop an image transformation in graph domain similar to KLT in terms of de-correlation and energy compaction which reduces the computational cost and increase the gain in compression?

As a solution to the 1^{st} research question this work encourage to develop improved graph structure and prediction methods which avoids signalling overhead by exploiting the prediction models such as template based prediction in several domains, prediction inaccuracy modeling and many more. Approaching to the 2^{nd} research question it involves machine learning algorithms. The objective is to exploit the optimization algorithm to minimise the gap in between the graph Laplacian and covariance function. More precisely, this research builds model with set of parameters which generates a signal with updated parameters of the graph signal by learning to produce an estimate towards minimizing the cost function with the covariance function of KLT.

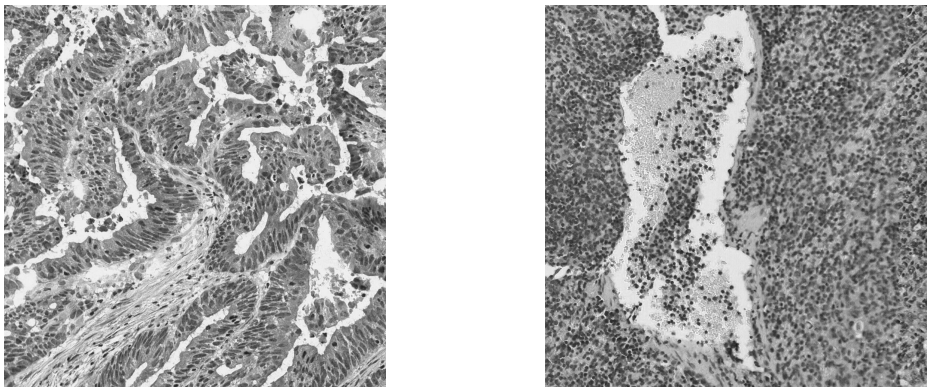
1.3 Contribution

1.3.1 Overall contribution to the thesis: Non-learning and learning based prediction

In this thesis we have addressed the issue of availability of the same graph at the decoder. As a solution we have provided two potential prediction strategy to predict the graph, each of which has advantages and disadvantages. We have proposed non-learning based methods for predicting the graph in Chapters 3 and Chapter 4. More precisely, these techniques heavily rely on mathematical derivations, including template-based strategy or modeling of prediction inaccuracy. The fundamental benefit of a non-learning based technique is that the results are independent of the quantity, amount, and applicability of training data. To infer any answer, however, requires a thorough understanding of mathematics because these approaches are solely reliant on mathematical optimizations. However, since ML/DL approaches have their

Figure 1.4: Class A: Luma component of Traffic (cropped). Resolution 2560 × 1600. Frame 30.

Figure 1.5: Class Screen Content: Luma component of Map. Resolution 1280 × 720. Frame 60.



(a) (b)

Figure 1.6: (a) G component of *colon* tissue. Resolution 1024 × 1024. (b) G component of *lymphatic* tissue. Resolution 1024 × 1024.

foundation on the relevance of training data, it is quite straightforward and more versatile to improve the accuracy of prediction for the graphs generated by such methods compared to other non-learning based approaches. We do not necessarily require mathematicians to implement any Ml/DL based prediction methods. The proposed methods in the following chapters (Chapter 5, Chapter 6, and 7) are based on DL for predicting graphs. These DL based approaches are able to automatically learn from data and make predictions without any human intervention.

1.3.2 Chapter-wise contribution

The contribution of this thesis area as follows:

1. **Methods for designing a framework to avoid signaling overhead while reconstructing in decoder for pathology image (Chapter 3)**

    1.1 This chapter focuses on construction based on the specific graph representing the signal to be transformed and its performance for data de-correlation and energy compaction of images, within the context of block-based PTC using intra-prediction. Specifically, we introduce a new framework that eliminates the need to signal additional information to the decoder. This is achieved by computing the GBT based on a predicted residual signal, which is computed

using only the reference samples used to predict a block. This variant accurately reflects the characteristics of the signal.

1.2 This framework is evaluated on a wide range of pathology images; specifically WSIs depicting different tissue types.

1.3 In this chapter, results are reported in terms of the energy compaction properties of the GBT and the Mean Squared Error (MSE) of the reconstructed images.

2. **Designing a GBT with Weighted Self-Loops (GBT-L) for PTC Based on Template Matching (Chapter 4)**

   2.1 Proposed a novel class of GBT within the context of block-based PTC. The GBT-L is constructed using a 2D graph with unit edge weights and weighted self-loops in every vertex. The weighted self-loops are selected based on the residual values to be transformed.

   2.2 To avoid signalling any additional information required to compute the inverse GBT-L, we also introduce a coding framework that uses a template-based strategy to predict residual blocks in the pixel and residual domains.

   2.3 Evaluation results on several video frames of natural and screen-content video frames and medical images, in terms of the percentage of preserved energy and mean square error, show that the GBT-L can outperform the Discrete Sine Transform (DST), DCT and the Graph-based Separable Transform (GBST).

3. **Offline learning of graphs based on deep neural networks for GBTs for Intra-Prediction of Imaging Data (Chapter 5 and Chapter 6)**

   3.1 In Chapter 5 we 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. To make the same graph available to the decoder for reconstruction, a template-based prediction strategy is used to predict the residual followed by a neural network (NN) that estimates the graph to perform the GBT. Specifically, this approach involves two prediction methods: predicting the residuals and using the predicted residuals as an input to the NN to predict the graph. This approach, unfortunately, tends to degrade the quality of the reconstructed residual at the decoder.

3.2 In Chapter 6 we attempted to address the issue of predicting twice and present a novel class of GBT based on 3D convolutional neural networks (GBT-CNN) within the context of block-based predictive transform coding of imaging data which uses the 3 reconstructed blocks surrounding the block to be encoded as input. The proposed GBT-CNN uses a 3D convolutional neural network (3D-CNN) to predict the graph information needed to compute the transform and its inverse, thus reducing the signalling cost to reconstruct the data after transformation.

3.3 In both Chapter 5 and Chapter 6 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, and the transform coding gain, show that the GBT-NN and GBT-CNN can outperform the DCT and DST, which are widely used in modern video codecs. GBT-CNN outperforms GBT-NN.

4. **Online Learning of graphs for GBT(GBT-ONL) (Chapter 7)**

4.1 In general, the idea of learning GBTs offline for compression by using ML has gained increasing popularity recently as discussed in Chapter 5 and Chapter 6.1. However, the performance of such Machine Learning (ML) and Deep Learning (DL)-based methods depends on the amount, quality, and relevance of the training data. This Chapter 7 address the problem of offline training and leverage online training to learn GBTs without requiring of any training data or offline training processes. We specifically propose an online GBT, hereinafter called GBT-ONL, for block based PTC in the context of intra-prediction. The GBT-ONL predicts the graph Laplacian needed to compute the GBT of each block by using an over-fitted NN that is optimized online. This allows the model to adapt to each block to accurately predict the graph needed to compute the GBT.

4.2 Since the training is performed online, it can be replicated at the decoder, thus avoiding the need to signal extra information to compute the inverse GBT for reconstruction.

1.4 Thesis outline

This thesis is organised as follows:

- **Chapter 2: Literature Review**
  This chapter reviews the history and the current state-of-the-art in four

key areas of image and video compression, predominant transforms for image and video coding, Graph based Transforms for image and video coding, and ML/DL for compression.

- **Chapter 3: Graph-Based Transforms based on Prediction Inaccuracy Modelling**
  This chapter introduces the idea of graph prediction for GBTs to avoid signaling overhead by proposing the idea of inaccuracy modeling for prediction (GBT-PI). Additionally, a framework is proposed for encoding-decoding based on proposed model.

- **Chapter 4: Graph-Based Transforms with Weighted Self-loops on Template based Prediction Strategy**
  This chapter introduces the idea of emphasizing on vertex weight of the graph to perform more accurate signaling of the residual block by proposing a GBT with self-loop (GBT-L). Additionally, this chapter proposes a coding framework for template matching and template pooling techniques in residual and pixel domain.

- **Chapter 5: Graph-Based Transforms based on Neural Networks for Intra-Prediction of Imaging Data**
  This chapter elaborates the necessity of learning graphs for GBTs exploiting the deep neural network architecture of multi-layer perceptron by proposing a class of GBTs based on Neural Networks (GBT-NN).

- **Chapter 6: Graph-Based Transforms based on 3D Convolutional Neural Network for Intra-Prediction of Imaging Data**
  This chapter introduces the idea of predicting graphs for GBTs by using the surrounded blocks of the residual block based on 3D convolutional neural network (GBT-CNN).

- **Chapter 7: Online Graph-based Transforms for Intra-Predicted Imaging Data**
  This chapter introduces the idea of online training of the graphs for GBTs (GBT-ONL) which avoids any pre-trained model for prediction.

- **Chapter 8: Conclusion and Future Work**
  In the final chapter, we summarise the contributions of this thesis. We then discuss applications of our work and new research directions made possible by the presented contributions.

1.5 Research Summary

In this research we have mainly focused on implementing novel ideas in the areas of GBT for block-based PTC. Each chapter of this thesis is a novel idea to

enhance the performance of GBT to enhance the compression framework. The journey of the thesis initially started with various non-learning based methods to predict the graph for GBT, followed by offline deep learning based ideas and ended up to online learning of GBT. Additionally, apart from predicting the graph we propose a complete framework of encoding-decoding process. While building ideas we emphasized to avoid signaling overhead while reconstructing the signal at the decoder. Evaluation results shows, each chapter provides us a gradual improvement in predicting the graph. Our publication establishes the novelty in this field of work.

Chapter 2

Literature Review

In this chapter, we survey works related to the contributions of this thesis. The chapter is organised as follows: Section 2.1 presents a overview of image and video compression techniques. Section 2.2 explains the predominant transforms for compression. Section 2.3 demonstrate the GBT in details related to our research, and Section 2.4 elaborates learning for compression.

2.1 Lossy and lossless image/video compression

Recently, the use of large volumes of image data in many applications like internet has been increasing rapidly. So, to make an effective use of storage space and also bandwidth of the network, image compression [21, 22] is required (see Fig. 2.1). We have two kinds of image compression - one is lossy and other is lossless image compression. With lossless compression, every bit of data initially in a file remains once it is decompress, and every information is restored. Further, lossless compression retains raster values during compression. At the same time, it still manages to reduce file size. On the other hand, lossy compression reduces a file by permanently removing specific information, particularly redundant information. As an example, LZ77 is a lossless compression file type and JPEG is a format that uses lossy compression. Indeed, lossless compression algorithms allow the original data to be perfectly

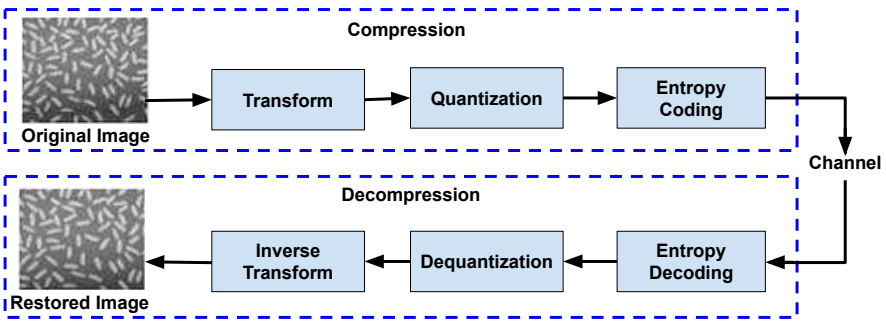


Figure 2.1: A general compression scheme. [1]

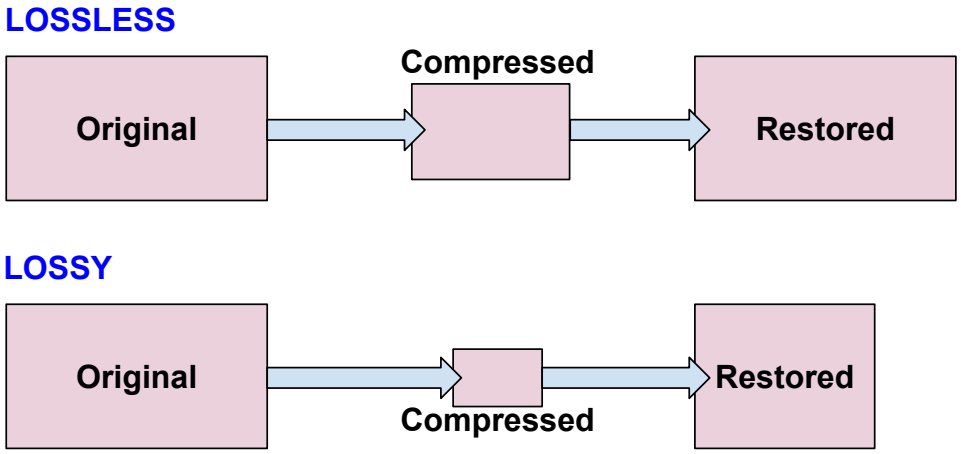


Figure 2.2: Lossy and lossless compression scheme. [2]

reconstructed from the compressed data. Basically, lossless compression more efficiently rewrites the original file's data [23]. The most likely application is for acquisition. So, anywhere high bit depth precision is required for retaining dynamic range, including with cameras and their proprietary RAW formats. The resulting files, however, are often significantly larger than image and audio files compressed with lossy compression because no quality is lost. On the other hand, lossy image compression produces a compressed image where quality of the image is maintained with some data loss. Lossy compression is widely used compared to lossless compression. One of the major difficulties encountered in lossy image compression is how to preserve image quality in such a way that the compressed image is always identical to the authentic, as opposed to the types of methods that exist in lossless image compression that can maintain the quality of the images authenticity (see Fig. 2.2). There are numerous methods for compressing images that can be used with various algorithms such as Huffman code [24], Chaudhuri and Hocquengham (BCH) codes [25], Multiple-Tables Arithmetic Code [26], Fractal Coding [27], Block Truncation Coding [28], and many others. The image is transformed in the transform domain to gain a rarely coefficient matrix using Discrete Wavelet Transform (DWT), DCT, and Fast Fourier Transform (FFT). The DCT method is very similar to the Discrete Fourier Transform (DFT), which converts a signal or image from the spatial domain to the frequency domain. Because of the wide range of images, including binary images, RGB images, and image intensities. Then use image compression to reduce the size of the data or techniques to reduce the number of bits required to reflect an image.

2.1.1 Related work on lossy image compression

There is a need for compression algorithms that are more adaptable than current codecs due to new media formats, evolving hardware technologies, and

a variety of requirements and content types. Now a days most of the image and video compression techniques adopt machine learning or deep learning concepts. In [29] the first conduct a comprehensive literature survey of learned image compression methods. The literature is organized based on a number of factors, such as network architecture, entropy model, and rate control, to jointly improve the rate-distortion performance with a neural network. By reviewing wide range of previous publications, the fundamental problems with picture compression techniques are exposed in this survey, along with potential solutions using cutting-edge advanced learning techniques. This study offers a chance to further the development of more effective image compression. They increase rate-distortion performance, particularly on high-resolution images, by incorporating a coarse-to-fine hyper-prior model for entropy estimation and signal reconstruction. In [30] authors propose a new approach to the problem of optimizing autoencoders for lossy image compression. They show autoencoders have the potential to fill this gap, but they are challenging to directly tune because the compression loss is inherently non-differentiable. In [31] the authors present a lossy image compression architecture, which utilizes the advantages of convolutional autoencoder (CAE) to achieve a high coding efficiency. First, they design a novel CAE architecture to replace the conventional transforms and train this CAE using a rate-distortion loss function. Second, to generate a more energy- compact representation, we utilize the principal components analysis (PCA) to rotate the feature maps produced by the CAE, and then apply the quantization and entropy coder to generate the codes. In [32] the authors propose a method for lossy image compression based on recurrent, convolutional neural networks that outperforms BPG (4:2:0), WebP, JPEG2000, and JPEG as measured by MS-SSIM. They introduce three improvements over previous research that lead to this state-of-the-art result using a single model. First, they modify the recurrent architecture to improve spatial diffusion, which allows the network to more effectively capture and propagate image information through the network's hidden state. Second, in addition to lossless entropy coding, they use a spatially adaptive bit allocation algorithm to more efficiently use the limited number of bits to encode visually complex image regions. Finally, they show that training with a pixel-wise loss weighted by SSIM increases reconstruction quality according to multiple metrics. In [33] the authors introduce a novel wavelet difference reduction (WDR) and singular value decomposition (SVD)-based lossy picture compression method (WDR). The performance of the WDR compression is enhanced by the combination of these two approaches. WDR compression delivers high compression whereas SVD compression offers very high image quality but low compression ratios. An input picture is first compressed using SVD in the Proposed approach, and then it is compressed once more using WDR. The WDR method is also used to

obtain the system's required overall compression ratio. Several test photos were used to evaluate the suggested image compression approach, and the outcomes were compared to those of WDR and JPEG2000. In [34] authors propose a fast solving method of fuzzy relational equation and applied to lossy compression and reconstruction problem, where it is confirmed that the computation time of the reconstructed image is decreased to 1/335.6 the compression rate being 0.0351, and it achieves almost equivalent performance for the conventional lossy image compression methods based on DCT and vector quantization. In [35], authors propose a novel invertible framework called Invertible Lossy Compression (ILC) to largely mitigate the information loss problem. Specifically, ILC introduces an invertible encoding module to replace the encoder-decoder structure to produce the low dimensional informative latent representation, meanwhile, transform the lost information into an auxiliary latent variable that won't be further coded or stored. The latent representation is quantized and encoded into bit-stream, and the latent variable is forced to follow a specified distribution, i.e. isotropic Gaussian distribution. In this way, recovering the original image is made tractable by easily drawing a surrogate latent variable and applying the inverse pass of the module with the sampled variable and decoded latent features. Experimental results demonstrate that with a new component replacing the auto-encoder in image compression methods, ILC can significantly outperform the baseline method on extensive benchmark datasets by combining with the existing compression algorithms. In [36] three lossy image compression techniques - Discrete DCT, Singular Value Decomposition (SVD) and DWT are used to perform image compression. These techniques are compared using some performance measures such as Peak Signal-to- Noise Ratio(PSNR), Compression Ratio(CR), Structural Similarity Index Measure(SSIM) and Mean Square Error(MSE). In [37] the authors aim to evaluate (1) storage needs, (2) subjective image quality, and (3) accuracy of caries detection in digital radiographs compressed to various levels by a lossy compression method. In [38] authors present a study of image compression methods algorithm for compare the best techniques on lossy image compression. Based on the findings of this study, four alternative ways of measuring the percentage of picture compression for each of the three methods—DCT, FFT, and DWT—starting with compressing images with sizes of 10%, 30%, 50%, and 70% have been developed. The optimal approach for compressing the image of the current percentage size can be determined by comparing the three ways with four distinct presentation measurement changes.

2.1.2 Modern image and video compression schemes

Digital video has grown pervasive in our daily lives; there are gadgets that can show, capture, and send video everywhere we look. UHD resolution video may now be recorded and displayed thanks to recent technological advancements. The capacity of the current Internet and TV networks is insufficient to transport vast amounts of HD video, let alone UHD, at this time [39]. In January 2010, a formal request for proposals (CfPs) on video compression technologies was made, and 27 proposals were submitted in response. These ideas were put forth at the initial JCT-VC gathering in April 2010. According to the evaluations that followed, some ideas may achieve the same visual quality as H.264/MPEG-4 advanced video coding (AVC) high profile at just half the bit rate and at a cost of a two- to tenfold increase in computational complexity. Compared to the reference AVC high-profile encoder, some alternative ideas could achieve good subjective quality and bit rates with less computational complexity. Since then, JCT-VC has made a significant effort to establish the HEVC standard, a new compression standard that aims to significantly outperform the current H.264/AVC high profile standard in terms of compression efficiency. The JCT-VC group's initial goal was to combine the salient elements of the top seven highly effective solutions into a single test model under consideration (TMuC), which served as the foundation for the first HEVC software codec known as HM [40]. JCT-VC has since hosted a number of meetings and assessed hundreds of submissions from both industry and academia. The best of these submissions underwent comprehensive evaluation and were incorporated into the HEVC standard. The main structure of the HEVC similar to the H.264/AVC encoder. Each picture in H.264/AVC is divided into 16×16 macroblocks, with the ability for each macroblock to be further divided into smaller blocks (as tiny as 4×4) for prediction [41]. The development of larger block structures with adaptable subpartitioning techniques is one of the key factors in HEVC's improved compression efficiency. Each picture in HEVC is partitioned into square picture areas called largest coding unit (LCU)s, which can be as large as 64×64. In general, the LCU concept in HEVC is similar to that of a macroblock in previous coding standards. LCUs are further subdivided into smaller units known as coding unit (CU)s, which serve as the basic unit for intra- and intercoding. Depending on the picture content, CUs can be as large as LCUs or recursively split into four equally sized CUs and become as small as 8×8. In HEVC, a content-adaptive coding tree structure comprised of CUs is created as a result of recursive quarter-size splitting [42, 43]. Each CU can be further subdivided into smaller units, which serve as the foundation for prediction. These are known as prediction unit (PU)s. Each CU can have one or more PUs, and each PU can be as big as the root CU or as small as

4×4 in luma block sizes. While an LCU can be recursively split into smaller and smaller CUs, a CU cannot be split into PUs (it can be done only once). Asymmetric or symmetric PUs can exist. Symmetric PUs can be square or rectangular (nonsquare) and are used in **intraprediction** (only square PUs are used) as well as **interprediction**. HEVC, like other video-coding standards, applies a discrete cosine transform (DCT)-like transformation to decorrelated residuals. A transform unit (TU) is the fundamental unit for the transform and quantization processes in HEVC. The size and shape of the TU are determined by the size of the PU. The size of square-shaped TUs can range from 4×4 to 32×32. Nonsquare TUs are available in 32×8, 8×32, 16×4, and 4×16 luma samples. Each CU can have one or more TUs, and each square CU can be divided into smaller TUs using a quad-tree segmentation structure.

It is well-known that intra prediction is the technology to predict a block, using what we already have decoded in the neighboring blocks of the same frame. Our research framework assumes a block-based PTC method that employs the set of intra-prediction modes currently used in the HEVC standard. This set comprises 33 angular prediction modes that model 33 different directional patterns; a DC mode and a PLANAR mode that generate smooth surfaces. For our research frame it is useful to categorise the type of residual block sharing some characteristics of interest, such as, the blocks predicted with same mode might have edge structure with similar orientation. Fig. 2.3(a) illustrates the prediction directions associated with the angular modes. The basic prediction principle for all angular modes is exemplified in Fig. 2.3(b) . Application of intra-prediction method to a 4×4 residual block is illustrated in Fig. 2.3(c) and Fig. 2.3(d). The Versatile Video Coding (VVC) standard's intra prediction and mode coding are discussed in [44, 45]. The Joint Video Experts Team produced this standard together (JVET). It adheres to the established hybrid block-based codec architecture that served as the foundation for earlier standards. Nearly all of the intra prediction aspects of VVC either have significant changes compared to its forerunner H.265/HEVC or are brand-new. In VVC, there are 65 intra directional prediction options as opposed to 33 in HEVC. Fig. 2.4 provides an example of the increasing directions. DC and planar modes are still in use in addition. To eliminate division operations, only samples from the longer side of the non-square block are used to calculate the average DC value for the DC mode. Similar to HEVC, intra mode coding consists of two components: 64 non-MPM modes using six-bit fixed length coding and three MPM modes from spatial neighbours. For non-square blocks, some of the traditional intra prediction modes are adaptively replaced by wide-angle directions, keeping the total number of intra prediction modes unchanged (67) [12]. The new prediction directions for non-square blocks are shown in Fig. 2.5, where the block width is smaller than block height. In general, more modes

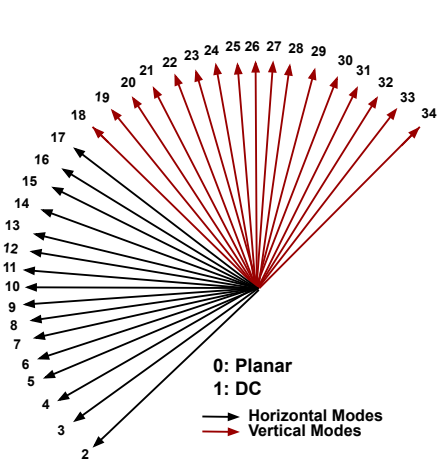
Table 2.1: Characteristics of tested HEVC video sequences.

Sequence	Resolution	Frame count	Frame rate(fps)	Bit depth	Source
4:2:0 YUV sequence					
Class A					
Traffic (Fig. 1.4)	2560×1600	150	30	8	CTC 2D
People_on_street	2560×1600	150	30	8	CTC 2D
Nebuta_festival	2560×1600	150	30	10	CTC 2D
Class B					
Kimono	1920×1080	240	24	8	CTC 2D
Cactus	1920×1080	500	50	8	CTC 2D
Park_scene	1920×1080	240	24	8	CTC 2D
BQTerrace	1920×1080	600	60	8	CTC 2D
Class C					
Race_horse	832×480	300	30	8	CTC 2D
BQMall	832×480	600	60	8	CTC 2D
Party_scene	832×480	500	50	8	CTC 2D
Basketball_drill	832×480	500	50	8	CTC 2D
Class D					
Race_horse_D	416×240	300	30	8	CTC 2D
Blowing_bubble	416×240	500	50	8	CTC 2D
BQ_square	416×240	600	60	8	CTC 2D
Basketball_pass	416×240	500	50	8	CTC 2D
Class E					
Kristine_and_Sara	1280×720	600	60	8	CTC 2D
Four_people	1280×720	600	60	8	CTC 2D
Jhonny	1280×720	600	60	8	CTC 2D
Class F/SC					
China_speed	1024×768	500	30	8	CTC 2D
Slide_show	1280×720	500	20	8	CTC 2D
Map (Fig. 1.5)	1280×720	600	60	8	CTC 2D
Programming	1280×720	600	60	8	CTC 2D

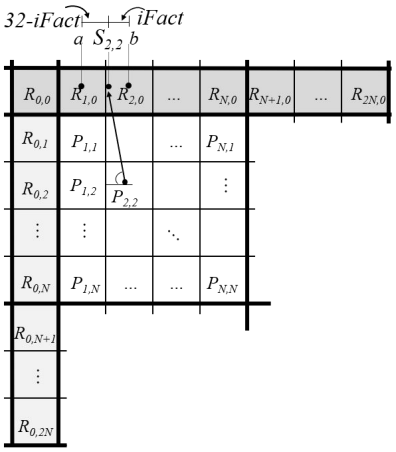
will be coming from the longer side of the block. In the case in Fig. 2.4, some modes near the top-right angular mode (mode 66 in Fig. 2.4) are replaced by additional angular mode below the bottom-left angular mode (mode 2 in Fig. 2.4). To support these prediction directions, the top reference with length 2W + 1, and the left reference with length 2H + 1, are defined as shown in Fig. 2.5. Table 2.1 of HEVC test sequences are of several class based on the resolution and characteristics which motivated us to test our experiments on various types of contents i.e. natural images of different resolutions, and screen content images.

2.1.3 Intra-prediction of HEVC and VVC standard

HEVC and VVC uses block-based intraprediction to take advantage of spatial correlation within a picture. The Joint Collaborative Team on Video Coding's HEVC standard's intra coding methods are described in [46] in general terms (JCT-VC). The HEVC intra coding system is founded on spatial sample prediction, transform coding, and postprocessing processes, similar to traditional hybrid codecs. A quad tree-based variable block size coding structure, block-



(a)



(b)

91	90	89	89	87
85	83	83	81	80
60	61	63	62	60
57	52	50	49	51
27	28	24	20	19

(c)

85	85	85	85
60	60	60	60
57	57	57	57
27	27	27	27

(d)

56	54	53	52	51
59	59	58	55	53
60	62	63	66	67
62	61	60	62	59
67	70	72	73	73

(e)

54	53	52	51
54	53	52	51
54	53	52	51
54	53	52	51

(f)

Figure 2.3: HEVC intra-prediction modes (a) Prediction direction, (b) Prediction principle [3], (c) A sample residual block with left and above reference samples, (d) Predicted block with *ideal* Horizontal prediction mode (Mode 10), (e) Another sample residual block with left and above reference samples, (f) Predicted block with *ideal* Vertical prediction mode (Mode 26).

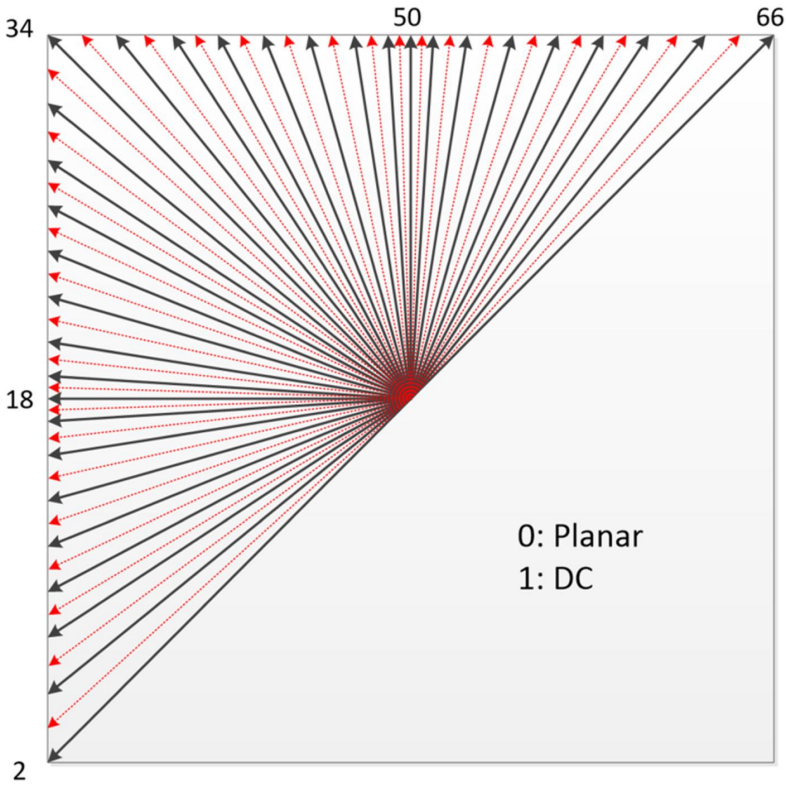


Figure 2.4: VVC modes. [4]

size agnostic angular and planar prediction, adaptive pre- and post filtering, and prediction direction-based transform coefficient scanning are novel features that help to boost compression efficiency. The work in [46] examines the design concepts used in the creation of the new intra coding techniques and evaluates how well each tool compresses data. Both operational cycle counts and bench marking an optimised implementation are used to determine the computational complexity of the newly introduced intra prediction methods. The bitrate reduction offered by the HEVC intra coding over the H.264/advanced video coding standard is reported using objective measurements. The Joint Exploration Model (JEM) algorithm and a matching software implementation were developed by the JVET, which was established following the development of HEVC, as part of its exploration of video coding technologies with improved coding efficiency. According to the Bjontegaard delta bit rate (BD-rate) metric, the technology investigated in the most recent JEM version further improves the compression capabilities of the hybrid video coding approach by introducing new tools, reaching up to 30% bit rate reduction compared to HEVC, and going above and beyond that in terms of subjective visual quality. As a result, a joint CfP for a new standardisation initiative known as VVC. All of the technology that was suggested in the CfP responses was based on the traditional block-based hybrid video coding design, but it was extended by new components like partitioning, intra- and inter-picture prediction, predic-

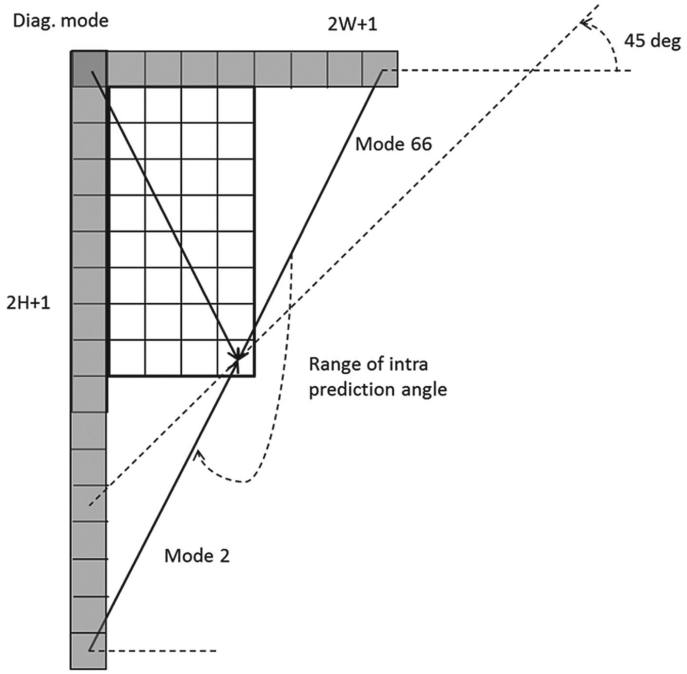


Figure 2.5: VVC angles. [4]

tion signal filtering, transforms, quantization/scaling, entropy coding, and in-loop filtering. An overview of the technology that was suggested in the CfP answers is given in [47], with an emphasis on methods that have not yet been investigated in the context of JEM. A video sequence can effectively use intra prediction to reduce the coded information included in an image or intra frame. Today's common procedure is to extrapolate the reconstructed pixels surrounding the target block to be coded to build a sample predictor block. The target block is subtracted from the sample predictor block, and the residual data is then transformed, quantized, and encoded using entropy. In the majority of sequences, this technique works well for creating sample predictor blocks. However, sample prediction blocks with complex texture cannot be represented using the extrapolation method. Additionally, pixels that are far distant from the pixel location are typically inadequately predicted. In [48] an innovative method for creating sample predictors by template matching in a region of reconstructed pixels is provided. Real-time mobile video applications have become challenging to create in recent times due to low latency and power constraints. A fast decision method for intra-coding unit size based on a new fuzzy support vector machine classifier is proposed in [49] in order to solve the above problems.

2.2 Predominant transformations for image compression

Image compression technology is the basis of all kinds of media compression and transmission, and its compression effect is directly related to the compression effect of media. In image compression framework transformation is a near-reversible process (due to finite precision arithmetic) that provides an image representation that is more amenable to the efficient extraction and coding of relevant information. The predominant transforms for image compression also includes DWT along with mentioned KLT and DCT. If one can find a reversible transformation that removes the redundancy by de-correlating the data, then an image can be stored more efficiently. The KLT is the linear transformation that accomplishes this. The basis vectors of the KLT are the eigen-vectors of the image covariance matrix. Its effect is to diagonalize the covariance matrix, removing the correlation of neighboring pixels. Another popular block-based linear transformation is DCT. DCT coefficients can be viewed as weighting functions that, when applied to the n^2 cosine basis functions of various spatial frequencies ($n \times n$ templates), will reconstruct the original block. DWT has a number of applications for signal coding, to represent a discrete signal in a more redundant form, often used to denoise two dimensional image signals. Wavelet decompresses an image as a whole. On the contrary, as our research framework adapts a block-based PTC there is no use of wavelet transform in PTC. Among several commonly used image compression coding methods Transform coding is one of the methods which is used to compress still images [50]. In the following subsections we discuss the details of the predominant transforms.

2.2.1 Karhunen-Loève Transform

Karhunen-Loève Transforms are has many names [51] cited in literature as Karhunen-Loève Expansion [52], PCA [53], Principal (or Principle) Factor Analysis (PFA) [54], SVD [55], Proper Orthogonal Decomposition (POD) [56]. Further KLT is also cited in literature as Galerkin Method [57] where this variation is used to find solutions to certain types of Partial Differential Equations, specially in the field of Mechanical Engineering and electromechanically coupled systems. Additionally, KLT is also cited as Hotelling Transform [58, 59] and Collective Coordinates [60] in few literature. KLT has been widely used in several sectors as studies of turbulence [61, 62], thermal/chemical reactions [63, 64], feed-forward and feedback control design applications [65, 66] where KLT is used to obtain a reduced order model for simulations or control design, and data analysis or compression [67–72] mostly in characterization

of human faces, map generation by robots and freight traffic prediction etc. In [67] authors propose the use of natural symmetries (mirror images) in a well-defined family of patterns (human faces) is discussed within the framework of the Karhunen-Loeve expansion. This results in an extension of the data and imposes even and odd symmetry on the eigenfunctions of the covariance matrix, without increasing the complexity of the calculation. The resulting approximation of faces projected from outside of the data set onto this optimal basis is improved on average. In [68] authors study a method to query large online database using image content as the basis of queries. They address the problems of vectors with high dimensionality with KLT that helps to reduce dimensionality without introducing the dismissals. In [69] the authors prove the KLT for a class of signals to be a set of periodic sine functions and this KL series expansion is obtained via an FFT algorithm. This fast algorithm obtained is useful in data compression and other mean-square signal processing application. In [70] the authors analyze a Karhunen-Loeve transform technique for ECG data compression. This transform has been, applied in two different ways: to the entire beat signal and to independent windows (P wave, QRS complex and ST-T complex). The optimum number of coefficients and bits for coding the signal is analyzed for the MIT-BIH Arrythmia database. The data compression performance of both choices are: for the entire beat a mean compression ratio of 12.1 with a mean MSE of 0.3% and for shorter windows a mean compression ratio of 17.21 with a mean value of MSE of 0.44%. In [71] the authors propose a neural model approach for performing adaptive calculation of the principal components (eigen-vectors) of an input sequence's covariance matrix. The algorithm is based on applying Oja's modified Hebbian learning rule to each new covariance matrix that results from calculating the previous eigen-vectors. It is demonstrated that the approach converges to the next dominant component that is linearly independent of all previously determined eigen-vectors. By minimising an error function of the learning rate along the gradient descent direction, the optimal learning rate is calculated. The method is used to adaptively encode grey-level images by calculating a limited number of KLT coefficients that meet a specified performance criterion. In [73] the authors provide the reasoning behind the found discrepancies of spatial PCA for network-wide anomaly detection. The authors revisit PCA for anomaly detection and evaluate its performance on their data. They develop a slightly modified version of PCA that uses only data from a single router. Instead of correlating data across different spatial measurement points, they correlate the data across different metrics. With the help of the analyzed data, they explain the pitfalls of PCA and underline our argumentation with measurement results. They show that the main problem is that PCA fails to capture temporal correlation. They propose a solution to deal with this

problem by replacing PCA with the KLT. They find that when they consider temporal correlation, anomaly detection results are significantly improved.

The concepts of KLT is based on eigen-values and eigen-vectors. If $\mathbf{C}$ is a matrix of dimension $n \times n$ then the scalar λ is called eigen-value if $\mathbf{C}$ if there is a non-zero vector $\mathbf{e} \in \mathbb{R}^n$ such that:

$$\mathbf{C}\mathbf{e} = \lambda\mathbf{e} \tag{2.1}$$

where the vector $\mathbf{e}$ is the is called an eigen-vector of the matrix $\mathbf{C}$ corresponding to the eigen-value λ. Lets consider a population of random vectors of the following form:

$$\underline{\mathbf{x}} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} \tag{2.2}$$

Here the quantity x_i may represent the value (grey level) of the image i. Let us consider the mean vector of the population as:

$$\underline{\mathbf{m}}_{\mathbf{x}} = E\{\underline{\mathbf{x}}\} = \begin{bmatrix} m_1 & m_2 & \cdots & m_n \end{bmatrix}^T = \begin{bmatrix} E\{x_1\} & E\{x_2\} & \cdots & E\{x_n\} \end{bmatrix}^T \tag{2.3}$$

The covariance matrix of the population is defined as:

$$\mathbf{C} = E\{(\underline{\mathbf{x}} - \underline{\mathbf{m}}_{\mathbf{x}})(\underline{\mathbf{x}} - \underline{\mathbf{m}}_{\mathbf{x}})^T\} \tag{2.4}$$

For M vectors of a random population, where M is large enough

$$\underline{m}_x = \frac{1}{M}\sum_{k=1}^{M} \underline{x}_k \tag{2.5}$$

Lets assume, $\mathbf{R}$ be a matrix whose rows are formed from the eigen-vectors of the covariance matrix $\mathbf{C}$ of the population and they are ordered so that the first row of $\mathbf{R}$ is the eigen-vector corresponding to the largest eigen-value, and the last row the eigen-vector corresponding to the smallest eigen-value. Then the transform we can define as

$$\underline{\mathbf{y}} = \mathbf{R}(\underline{\mathbf{x}} - \underline{\mathbf{m}}_{\mathbf{x}}) \tag{2.6}$$

is defined as KLT. For reconstruction the original vectors $\underline{\mathbf{x}}$ from its corresponding $\underline{\mathbf{y}}$ we apply

$$\underline{\mathbf{x}} = \mathbf{R}^T\underline{\mathbf{y}} + \underline{\mathbf{m}}_{\mathbf{x}} \tag{2.7}$$

We form a matrix $\mathbf{R}_p$ from the p eigen-vectors which correspond to the p

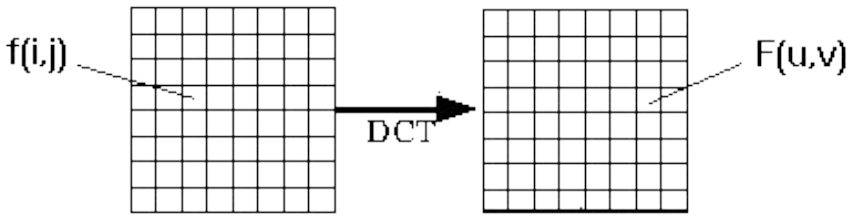


Figure 2.6: DCT transform. [5]

largest eigen-values, yielding a transformation matrix of largest eigen-values, yielding a transformation matrix of size $p \times n$. The $\underline{\mathbf{y}}$ vectors would then be p dimensional. The reconstruction of the original vector $\hat{\underline{\mathbf{x}}}$ is

$$\hat{\underline{\mathbf{x}}} = \mathbf{R}_p^T \underline{\mathbf{y}} + \underline{\mathbf{m}}_{\mathbf{x}} \tag{2.8}$$

It can be proven that the mean square error between the perfect reconstruction $\underline{\mathbf{x}}$ and the approximate reconstruction $\hat{\underline{\mathbf{x}}}$ is given by the expression

$$e_{ms} = \| \underline{\mathbf{x}} - \hat{\underline{\mathbf{x}}} \|^2 = \sum_{j=1}^{n} \lambda_j - \sum_{j=1}^{p} \lambda_j = \sum_{p+1}^{p} \lambda_j \tag{2.9}$$

By using $\mathbf{R}_p$ instead of $\mathbf{R}$ for the KL transform we can achieve compression of the available data.

The KLT is not implemented in practise despite its excellent theoretical features for the following reasons:

- Since its basis functions depend on the image's covariance matrix, they must be recalculated and communicated for each image.
- Perfect de-correlation is not possible, since images can rarely be modelled as realisations of ergodic fields.
- There are no fast computational algorithms for its implementation.

2.2.2 DCT

DCT has emerged as a image transformation in most visual system. DCT has been widely developed by video coding standards, as, MPEG, JVET etc and also been used for modern standards as HEVC, and VVC. It is a lossless transform. However, due to it's properties (de-correlation of an input and concentration of most of the information in lower bins) it is being used in lossy algorithms. DCT represents an image as a sum of sinusoids of varying magnitudes and frequencies. The DCT2 function computes the two-dimensional DCT of an image. DCT has several properties, such as, decorrelation, energy compaction, separability, symmetry, orthogonality. The DCT helps separate the

image into parts (or spectral sub-bands) of differing importance (with respect to the image's visual quality). The DCT is similar to the discrete Fourier transform: it transforms a signal or image from the spatial domain to the frequency domain (Fig 2.6). In [74] propose a new framework for digital image processing; it relies on inexact computing to address some of the challenges associated with the DCT compression. The proposed framework has three levels of processing; the first level uses approximate DCT for image compressing to eliminate all computational intensive floating-point multiplications and executing the DCT processing by integer additions and in some cases logical right/left shifts. The second level further reduces the amount of data (from the first level) that need to be processed by filtering those frequencies that cannot be detected by human senses. Finally, to reduce power consumption and delay, the third level introduces circuit level inexact adders to compute the DCT. For assessment, a set of standardized images are compressed using the proposed three-level framework. Different figures of merits (such as energy consumption, delay, power-signal-to-noise-ratio, average-difference, and absolute-maximum-difference) are compared to existing compression methods; an error analysis is also pursued confirming the simulation results. In [75] authors introduce a fast JPEG image compression algorithm based on DCT. The algorithm introduces the process of image coding and decoding for JPEG. The encoding part of the image can process the BMP format image by JEPG, and compress it into a binary file for real-time storage. The image can be decompressed by the corresponding decoding program. In addition, in the process of image transmission, taking advantage of the fact that human vision is not sensitive to chroma, JPEG format can be used to encode static image, and the color RBG of JPEG image can be changed into brightness y, chroma Cr and CB, which can not only effectively reduce chroma data, but also achieve compression. In [76] authors propose a hybrid Integer wavelet transform (IWT) and DCT based compression technique to obtain increased quality of decompressed image compared to DWT+ DCT based compression technique. The proposed combined IWT + DCT based compression technique reduces the fractional loss compared to DWT based compression so the proposed technique provides better image quality of decompressed image on high compression ratio compared to DWT based and hybrid DWT DCT based image compression techniques. In [77] the authors propose a low-complexity 8-point orthogonal approximate DCT . The proposed transform requires no multiplications or bit-shift operations. The derived fast algorithm requires only 14 additions, less than any existing DCT approximation. Moreover, in several image compression scenarios, the proposed transform could outperform the well-known signed DCT, as well as state-of-the-art algorithms. In [77] authors propose a block transform for image compression, where the transform is inspired by DCT but achieved by training

CNN models. Specifically, the authors adopt the combination of convolution, nonlinear mapping, and linear transform to form a non-linear transform as well as a non-linear inverse transform. The transform, quantization, and inverse transform are jointly trained to achieve the overall rate-distortion optimization. For the training purpose, the authors propose to estimate the rate by the l1-norm of the quantized coefficients. They also explore different combinations of linear/non-linear transform and inverse transform. Experimental results show that our proposed CNN-based transform achieves higher compression efficiency than fixed DCT, and also outperforms JPEG significantly at low bit rates. In [78] authors propose a block transform for image compression, where the transform is inspired by DCT but achieved by training convolutional neural network (CNN) models. Specifically, they adopt the combination of convolution, nonlinear mapping, and linear transform to form a non-linear transform as well as a non-linear inverse transform. The transform, quantization, and inverse transform are jointly trained to achieve the overall rate-distortion optimization. For the training purpose, they propose to estimate the rate by the l_1 norm of the quantized coefficients. They also explore different combinations of linear/non-linear transform and inverse transform. Experimental results show that the proposed CNN-based transform achieves higher compression efficiency than fixed DCT, and also outperforms JPEG significantly at low bit rates. In [79] DCT based image compression using blocks of size 32x32 is considered. An effective method of bit-plane coding of quantized DCT coefficients is proposed. Parameters of post-filtering for removing of blocking artifacts in decoded images are given. The efficiency of the proposed method for test images compression is analyzed. It is shown that the proposed method is able to provide the quality of decoding images higher than for JPEG2000 by up to 1.9 dB. In [80] a model is developed to approximate visibility thresholds for DCT coefficient quantization error based on the peak-to-peak luminance of the error image. Experimentally measured visibility thresholds for R, G, and B DCT basis functions can be predicted by a simple luminance-based detection model. This model allows DCT coefficient quantization matrices to be designed for display conditions other than those of the experimental measurements: other display luminances, other veiling luminances, and other spatial frequencies (different pixel spacings, viewing distances, and aspect ratios). In [81] authors attempt to implement basic JPEG compression using only basic MATLAB functions. In this paper the lossy compression techniques have been used, where data loss cannot affect the image clarity in this area. Image compression addresses the problem of reducing the amount of data required to represent a digital image. It is also used for reducing the redundancy that is nothing but avoiding the duplicate data. It also reduces the storage area to load an image. For this purpose the authors are using JPEG. JPEG is a still frame compression standard,

which is based on, the DCT and it is also adequate for most compression applications. The DCT is a mathematical function that transforms digital image data from the spatial domain to the frequency domain. In [82] authors present an application of the DCT compression technique on medical images of the IRM type. The arithmetic coding method is used to encode the coefficients. The tests of this lossy compression/ decompression technique are performed on two IRM images representing the brain, in axial and sagital views, of a patient suffering from a cerebral hemorrhage. The obtained results on these images show that the DCT technique permits to considerably improve the compression rate while maintaining a good image quality when threshold varies in the interval: $0 \leq TH \leq 20$ for block sizes: 16×16 and 32×32. However, a severe degradation of the quality of the reconstructed medical image is observed when the threshold is greater than 30. In [83] the authors show DCT is a technique for converting a signal into elementary frequency components. Here we develop some simple functions to compute the DCT and to compress images. These functions illustrate the power of mathematical in the prototyping of image processing algorithms.

The most common DCT used for compression are of 2 types. The general equation for a 1D (N data items) DCT is defined by the following equation:

$$F(u) = \left(\frac{2}{N}\right)^{\frac{1}{2}} \sum_{i=0}^{N-1} \Lambda(i) \cos\left[\frac{\pi u}{2N}(2i+1)\right] f(i) \tag{2.10}$$

where

$$\Lambda(i) = \begin{cases} \dfrac{1}{\sqrt{2}} & \text{for } \varepsilon = 0 \\ 1 & \text{otherwise} \end{cases} \tag{2.11}$$

The general equation for a 2D (N by M image) DCT is defined by the following equation:

$$F(u,v) = \left(\frac{2}{N}\right)^{\frac{1}{2}} \left(\frac{2}{M}\right)^{\frac{1}{2}} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} \Lambda(i,j) \cos\left[\frac{\pi u}{2N}(2i+1)\right] \cos\left[\frac{\pi v}{2M}(2j+1)\right] f(i,j) \tag{2.12}$$

where

$$\Lambda(i) = \begin{cases} \dfrac{1}{\sqrt{2}} & \text{for } \varepsilon = 0 \\ 1 & \text{otherwise} \end{cases} \tag{2.13}$$

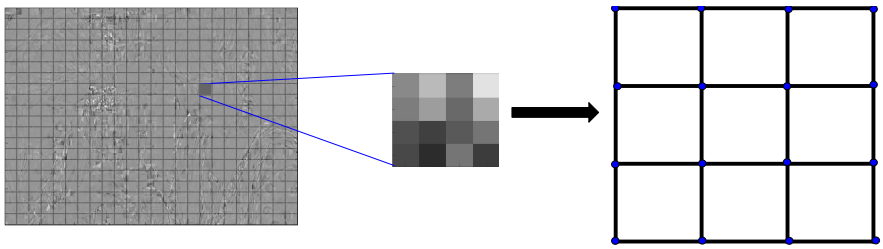


Figure 2.7: A block (4×4) of a residual signal represented as a 4-connected graph.

2.2.3 DCT/DST

DCT/DST transform is one of the predominant transforms use for video and image compression. The energy compaction of spatial domain data into frequency domain data during video compression depends heavily on transformation. The spatial residual signals in the HEVC intra prediction are concentrated into low-frequency components using DCT and DST techniques. DCT has demonstrated strong compression performance in both intra and inter residual coding, although its coding effectiveness declines when the spatial residual signals are not evenly distributed. The work in [84] that uses residual-rearranged DST or DCT to boost HEVC intra coding's coding effectiveness. For all block sizes, the suggested technique chooses the DCT or residual-rearranged DST with the highest coding efficiency. The experimental results show that, compared with the HEVC intra coding, the proposed method reduces the luma BD rates by 2.6%. Similarly, in [85], authors present a mode-dependent transform scheme that applies either the conventional DCT or type-7 DST for all the video-coding intra-prediction modes: vertical, horizontal or oblique. Their approach is applicable to any block-based intra prediction scheme in a codec, that employs transforms along the horizontal and vertical direction separably. Here the authors prove that this is indeed the case for the other oblique modes. The choice of using DCT/DST is based on intra-prediction modes, and requires no additional signaling information or Rate-Distortion search. Simulations are conducted for the DCT/DST algorithm in TMuC 0.9, the reference software for the ongoing HEVC standardization. The authors show that the DCT/DST scheme provides significant BD-Rate improvement over the DCT for intra prediction in video sequences. In [86] a refined generalized signal flow graph for the direct 2-D DCT and 2-D DST computation (the so-called 2-D DCT/DST universal computational structure) is described.

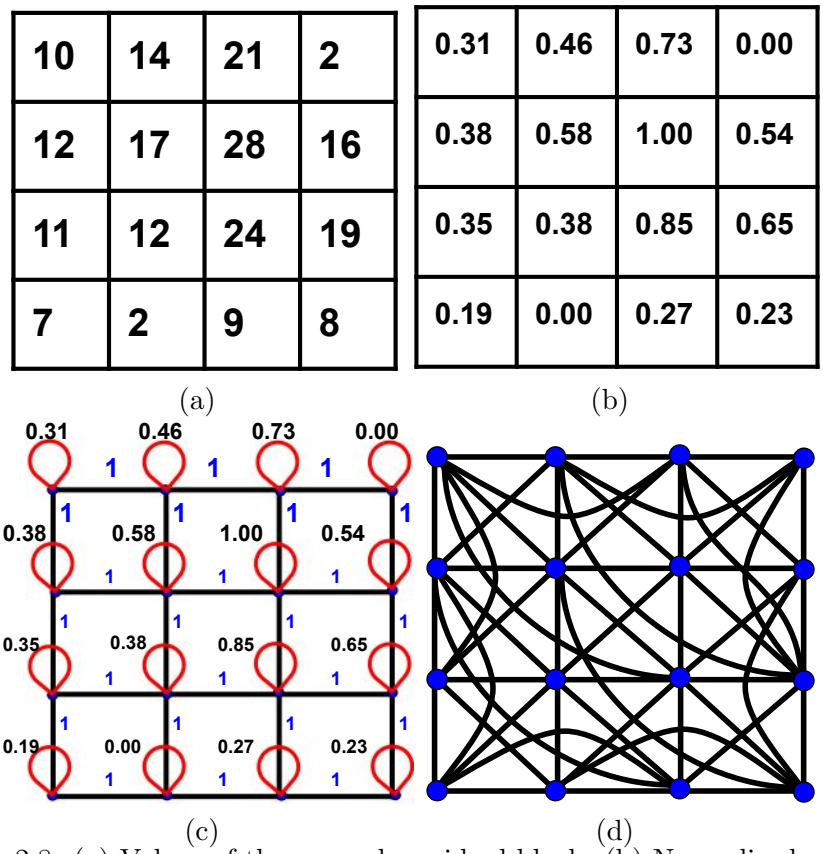


Figure 2.8: (a) Values of the example residual block. (b) Normalized residual values to the range $[0, 1]$. (c) Corresponding graph with a 4-connected topology with unit edge weights and self-loops in each vertex. (d) All-connected topology with no self-loops (i.e., each node is connected to every node in the graph).

2.3 GBT for image and video coding

In this section, we first describe how GBTs are computed for blocks of residuals. For example as shown in Fig. 2.7, a residual block is represented as a graph by 4 connectivity pattern which lead to particular interpretations in graph transform domain. The review is followed by a summary of how GBTs are used the context of block-based PTC. We then review several works that have attempted to learn GBTs offline, followed by relevant works that have attempted to learn other transforms used for image and video compression.

2.3.1 GBTs for blocks of residuals

The GBT of a residual (square) block $\mathbf{S} \in \mathbb{R}^{\sqrt{N}\times\sqrt{N}}$ with N residual values is usually constructed by eigendecomposition of the graph Laplacian, $\mathbf{L}$, of its undirected 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 weighted adjacency matrix. The entry $\mathbf{A}_{ij}$ in $\mathbf{A}$ represents the weight of the edge e_{ij} connecting vertices v_i and v_j, with $\mathbf{A}_{ij} = \mathbf{A}_{ji}$. If there is no edge $e = (i, j)$ connecting pixel locations

i and j, $\mathbf{A}_{ij} = 0$. Large values in $\mathbf{A}$ usually represent a high similarity between the connected nodes, according to a given criterion. The graph Laplacian $\mathbf{L}$, is computed as $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D}$ is the diagonal degree matrix, whose n^{th} diagonal element is equal to the sum of the weights of all edges incident onto node v_n. The eigendecomposition of $\mathbf{L}$ is used as the orthogonal transform for the residual block, since it has a complete set of eigen-vectors with real, non-negative eigen-values. Let us denote the eigendecompostion of $\mathbf{L}$ by $\{\lambda_q, \mathbf{u}_q\}$ where $\lambda_q/\mathbf{u}_q$ is the q^{th} eigen-value/eigen-vector pair and $\mathbf{U}$ is the set of eigen-vectors. Analogous to the classical Fourier transform, one can define the GBT, $\hat{\mathbf{s}}$ of signal $\mathbf{s} \in \mathbb{R}^N$ which resides on the nodes of G, as the expansion of $\mathbf{s}$ in terms of the eigen-vectors of $\mathbf{L}$:

$$\hat{\mathbf{s}}(\lambda_q) = \langle \mathbf{s}, \mathbf{u}_q \rangle = \sum_{k=0}^{|N|-1} \mathbf{s}(k)\mathbf{u}_q(k) = \mathbf{F}\mathbf{s} \tag{2.14}$$

where $\mathbf{F} = \mathbf{U}^{-1}$ is the graph Fourier transform and the set of eigen-values of $\mathbf{L}$, denoted by $\sigma(\mathbf{L}) = (\lambda_0, \lambda_1, \cdots, \lambda_{N-1})$, is the entire corresponding spectrum. The original signal can be reconstructed by the inverse GBT, which is given by $\mathbf{s} = \mathbf{F}^{-1}\hat{\mathbf{s}} = \mathbf{U}\hat{\mathbf{s}}$. As a graph is defined by an adjacency matrix, $\mathbf{A}$, it is possible to generate different transforms for the same block by using different graph connectives and weights of G [15]. In general, the graph connectivity and the edge weights are inferred from the data (see Fig. 2.8). A Gaussian kernel is usually used to define the edge weights of the graph, w_{ij} for vertices i and j:

$$\mathcal{W}_{ij} = \begin{cases} e^{-\frac{[dist(i,j)]^2}{2\theta^2}}, & \text{if } dist(i,j) \leq a \\ 0, & \text{otherwise} \end{cases}$$

where $\mathcal{W}$ is the calculated weight of two connecting vertices, i and j, $dist(i,j)$ represents the Euclidean distance between the residual value associated with nodes i and j, θ is the kernel width, and a is a hyper-parameter.

2.3.2 GBTs in the context of PTC

Within the context of block-based PTC, the GBT performs significantly well in generating de-correlated coefficients that can compact the signal's energy into a few significant coefficients [87]. In [88], the authors show a theoretical analysis of optimal PTC based on the Gaussian Markov Random Field (GMRF) model to contruct GBTs. It is shown that PTC for graph-based models is optimal as long as the image follows the GMRF model closely since the eigen-analysis of the precision matrix of the GMRF model is optimal in de-correlating the signal. Several works on the Graph Fourier Transform (GFT), which is also a GBT, has

been conducted within the context of PTC [18, 89, 90]. As an instance in [91] the authors propose an optimized transform for the prediction residual, based on GFT by introducing a intra-prediction schemes exploiting the cluster differences between neighboring pixel pairs and the cluster mean to predict the block. The cluster indices are transmitted per block, allowing the decoder to mimic the same intra-prediction which outperforms combinations of their previous intra-prediction and ADST coding by 2.5 dB in PSNR on average. In our previous work [7], we propose novel class of GBT in PTC which is constructed using a 2D graph with unit edge weights and weighted self-loops in every vertex. The weighted self-loops are selected based on the residual values to be transformed. We have shown the self-loop for each vertex of the graph for GBTs can accurately represent the residual signal. To avoid signalling any additional information required to compute the inverse GBT, we also introduce a coding framework that uses a template-based strategy to predict residual blocks in the pixel and residual domains. Our evaluation results on several video frames and medical images show that this approach can outperform the DCT/DST, DCT and the Graph-based Separable Transform (GBST). Our another work [6] for GBT in PTC we introduce a novel framework that eliminates the need to signal graph information to the decoder to recover the coefficients. This is accomplished by computing the GBT using predicted residual blocks, which are predicted by a modeling approach that employs only the reference samples and information about the prediction mode. Evaluation results on several pathology images, in terms of the energy preserved and MSE when a small percentage of the largest coefficients are used for reconstruction, show that the GBT can outperform the DCT/DST and DCT. GBTs for inter-prediction [15, 92] has also gained popularity in video coding domain by significantly outperforming traditional DCT and KLT in terms of rate-distortion performance. The authors propose novel graph-based transforms (GBTs) for coding inter-predicted residual block signals by developing edge adaptive GBTs (EA-GBTs) derived from graphs estimated from residual blocks design template adaptive GBTs (TA-GBTs) by introducing simplified graph templates generating different set of GBTs with low transform signaling overhead. The experimental results show their methods significantly outperform traditional DCT and KLT in terms of rate-distortion performance. In a recent paper by S. Bagheri and *et.*al [93], authors we pursue a hybrid model-based / data-driven approach, to encode an intra-prediction residual block: the first few eigenvectors of a transform matrix are derived from a statistical model, e.g., the asymmetric discrete sine transform (ADST), for stability, while the remaining are computed from emirical covariance matrix for data adaptivity. The transform computation is posed as a graph learning problem, where we seek a graph Laplacian matrix minimizing a graphical lasso objective inside a convex cone sharing the first K eigenvectors in a Hilbert

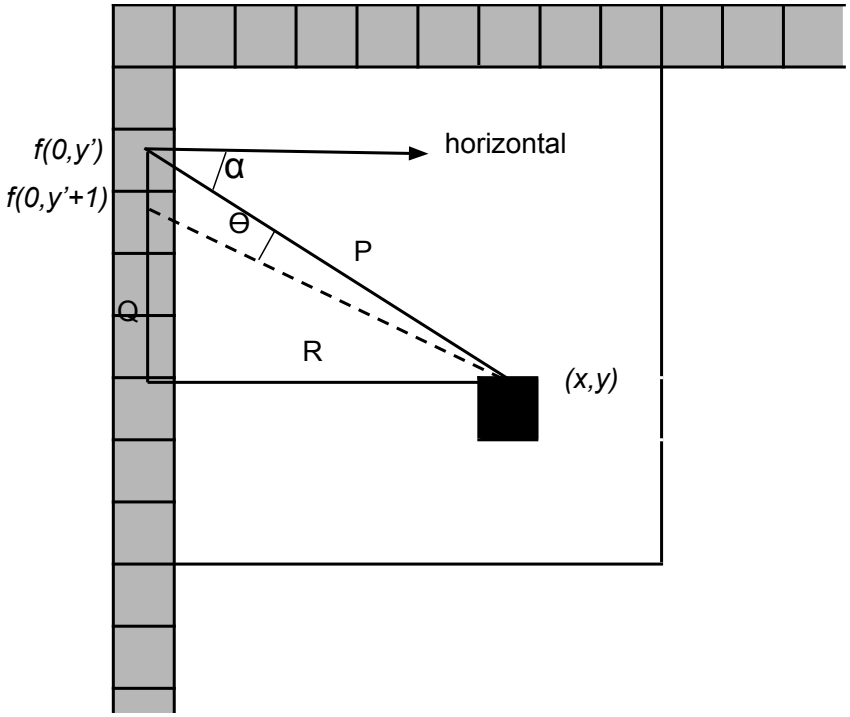


Figure 2.9: Prediction inaccuracy modeling technique (explained in details in Chapter 3).

space of real symmetric matrices. Authors efficiently solve the problem via augmented Lagrangian relaxation and proximal gradient.

2.3.3 Non-learning based prediction in GBTs

Within the context of block-based PTC, to avoid the signalling overhead of graph information to the decoder for reconstruction our research work [6, 7] exploits on several non-learning based prediction models which only use reference samples and intra-prediction mode at the decoder side for reconstruction of image and video sequences as those are always available to the decoder. As an

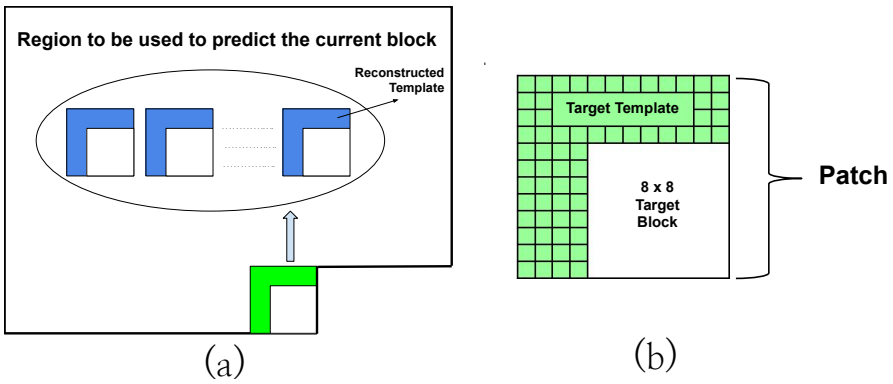


Figure 2.10: Template based prediction strategy (explained in details in Chapter 4).

instance in [94] the authors develop transforms for directional intra prediction residuals (see Fig. 2.9). In particular, authors observe that the directional intra prediction is most effective in smooth regions and edges with a particular direction. In the ideal case, edges can be predicted fairly accurately with an accurate prediction direction. In practice, an accurate prediction direction is hard to obtain. Based on the *inaccuracy of prediction direction* that arises in the design of many practical video coding systems, authors estimate the residual covariance and propose a class of transforms based on the estimated covariance function. Similarly, in [95–97] authors used template based strategy where the surrounded residual values are considered as a reference for predicting residuals instead of using the actual residual information (see Fig. 2.10).

2.3.4 Offline learning of GBTs

Recently, several works that attempt to learn optimal GBTs in the context of block-based PTC have been proposed. In [98], the author proposes two different techniques to design GBTs. In the first technique, they formulate an optimization problem to learn graphs from data and provide solutions for optimal separable and non-separable GBT designs, called GL-GBTs. The optimality of the proposed GL-GBTs is also theoretically analyzed based on GMRF models for intra and inter predicted block signals. The second technique develops edge-adaptive GBTs (EA-GBTs) in order to flexibly adapt transforms to block signals with image edges (discontinuities). The advantages of EA-GBTs are both theoretically and empirically demonstrated. The experimental results show that the proposed transforms can significantly outperform the traditional KLT. To accomplish this task, they train a large model offline with a large dataset collected by predicting blocks of several sizes with several intra-prediction modes. In [99], the authors propose a new class of transform named as graph template transforms (GTT) that approximates the KLT by exploiting a priori information known about signals represented by a graph-template. In order to construct a GTT (i) a design matrix leading to a class of transforms is defined, then (ii) a constrained optimization framework is employed to learn graphs based on given graph templates structuring a priori known information. The experimental results show that some instances of the proposed GTTs can closely achieve the rate-distortion performance of KLT with significantly less complexity. The work in [100] proposes a new edge model for edge adaptive graph-based transforms (EA-GBTs) in video compression. More specifically, the authors consider step and ramp edge models to design graphs used for defining transforms, and compare their performance on coding intra and inter predicted residual blocks. In order to reduce the signaling overhead of block-adaptive coding, a new edge coding method is introduced for

the ramp model. The experimental results show that the proposed methods outperform classical DCT-based encoding and that ramp edge models provide better performance than step edge models for intra predicted residuals. In [16], the authors introduce a novel class of transforms, called GBSTs, based on two line graphs with optimized weights. For the optimal GBST construction, the authors formulate a graph learning problem to design two separate line graphs using row- wise and column-wise residual block statistics, respectively. They analyze the optimality of resulting separable transforms for both intra and inter predicted residual block models. The work shows that separable DCT and ADST (DST-7) are special cases of the GBSTs. The experimental results demonstrate that the proposed optimized transforms outperform 2-D DCT/ADST and separable KLT. Our previous work [10] proposed a novel class of GBT based on 3D convolutional neural networks (GBT-CNN) within the context of block-based PTC of imaging data. The proposed GBT-CNN uses a 3D convolutional neural network (3D-CNN) to predict the graph information needed to compute the transform and its inverse, thus reducing the signalling cost to reconstruct the data after transformation. The GBT-CNN outperforms the DCT and DCT/DST in terms of the percentage of energy preserved by a subset of transform coefficients, the mean squared error of the reconstructed data, and the transform coding gain according to evaluations on several video frames and medical image. Further, in [101], a GBT is learned for predictive light field compression. In [102], the authors address a problem of learning graph Laplacians by adopting a factor analysis model for the graph signals that enforces minimizing the variations of the signals on the learned graph. It is important to note that the methods reviewed here require to train a model offline with the appropriate training data.

2.4 Machine Learning/ Deep Learning for compression

Apart from GBTs, other methods to learn transforms for compression purposes may be found in the literature [103, 104]. For example, in [105], the authors propose a fully unsupervised deep-learning framework that is able to extract a meaningful and sparse representation of raw high frequency signals by embedding important properties of the fast discrete wavelet transform (FDWT) in the architecture. With their framework, the denoising FDWT becomes a fully learnable unsupervised tool that does not require any type of pre- or postprocessing or any prior knowledge on wavelet transform. The application of wavelet transform is broadly used on imaging data [106, 107]. In [106] the authors learn the DNA structure using wavelet transform. In [107] an optimal

wavelet transform is proposed for the detection of micro-aneurysms in retina photographs by learning adapted wavelet filters. However, for block-based PTC Wavelet is not very popular since wavelet transform decompresses an image as a whole. On the contrary, as our research framework adapts a block-based PTC there is no use of wavelet transform in PTC. Since the DCT is widely used for block-based PTC, many works attempt to learn the mapping relationship between JPEG images and original images to reduce compression artifacts by using deep neural network [108–112]. Since the KLT is considered as the optimal transform that generates the most de-correlated coefficients, a huge volume of research groups works on learning the KLT offline [113]. For example, in [114] the authors propose a novel signal-independent separable transform based on the KLT to improve the efficiency of both intra and inter residual coding. In the proposed method, the drawbacks of the traditional KLT are addressed. A group of mode-independent intra transform matrices is calculated from abundant intra residual samples of all intra modes, while the inter separable KLT matrices are trained with the residuals that cannot be efficiently processed by the discrete cosine transform type II (DCT-II). KLT matrix are trained offline by combining all the residual blocks with different intra modes to take sufficient residual characteristics into the covariance matrix. In [115] the authors again propose a framework to design separable transforms from prediction residual statistics. The work model the data as a 2D GMRF and approximate its inverse covariance by a matrix with a separable structure, thus explicitly constructing a separable orthonormal matrix that approximates the KLT. The designed transforms can adapt to prediction residual statistics, have low complexity (compared to non separable transforms), require selecting few parameters and outperform hybrid DCT/ADST separable transform for intra coding of AV1 residuals. In literature DL models are also exploited for learning KLT [116].

2.5 Summary

This chapter presented an overview of existing research related to our contributions in graph-based signal processing for image and video compression. Firstly, we contextualised the importance of compression for images and videos. We discussed the literature on recent compression schemes. We ave reviewed several related works on the modern compression standard. At the end, we discussed on the mostly used PTC, i.e., intra-prediction of modern video coding standard.

Secondly, we discussed the importance of predominant transforms in the field of compression. We reviewed the recent and relevant works on KLT, DCT, and DCT/DST which are considered as mostly used transforms in the filed

of compression. Here we discussed the advantage and disadvantages of those transforms.

Thirdly, we elaborated our contribution on GBT for image and video coding. We reviewed the literature on the following perspective. At first we explained the modus operandi of GBT on residual blocks of an image. Additionally, we discuss the way GBT works in the context of PTC. Here we reviewed recent papers on the related work to provide an overview to the reader about the work. Further, we discussed on the non-learning based prediction of graph for GBT, followed by, learning based prediction. In Chapter 3 we introduced idea of a non-learning based graph prediction which is based on prediction inaccuracy modeling. In Chapter 4 the same trend of non-learning based graph prediction has been continued. However, for this chapter we used template based prediction strategy. In Chapter 5 and Chapter 6.1 we proposed the DL based architectures for graph prediction. We noticed that there are several works on learning graphs used for GBT by exploiting the ML and DL ideas which are offline, such as our works in Chapter 5 and Chapter 6.1 are offline learning. At the end, we reviewed the contribution by other trasnforms for learning graphs for GBTs.

Chapter 3

Graph-Based Transforms based on Prediction Inaccuracy Modeling for Pathology Image Coding

3.1 Introduction

Thanks to the introduction of high-throughput slide scanners, microscope glass slides can now be digitized to produce color images, which are called WSIs. This has fueled the emerging area of digital pathology imaging and resulted in novel ways to share medical imaging data and collaborate remotely [117, 118]. WSIs are multi giga-pixel color images that usually require large amounts of bandwidth to be transmitted and stored. Compression is therefore an attractive solution for data access and transmission of these images [119–123]. Recent proposals in this area include lossless compression methods based on the intra-prediction mode of the HEVC standard [11, 120] , and lossy methods based on the JPEG2000 standard [123, 124]. Although lossless compression guarantees perfect reconstruction of the image, it fails to attain high compression ratios. Lossy compression is then more advisable, especially since it has been shown that compression ratios of up to 60:1 can be used on WSIs without negatively affecting the diagnosis process [123].

In this chapter, we introduce a new framework that eliminates the need to signal additional information to the decoder. This is achieved by computing the GBT based on a predicted residual signal, which is computed using only the reference samples used to predict a block. This framework is evaluated on a wide range of pathology images depicting different tissue types. Results are reported in terms of the energy compaction properties of the GBT and

the MSE of the reconstructed images. The results are compared to those attained by the KLT, DCT, GBT when information is needed the be signaled to the decoder, and the DST), as implemented in the intra-prediction mode of HEVC [125]. Evaluations show that the GBT attains better energy compaction properties than the DST and the DCT for the evaluated pathology images, with a very similar performance in terms of MSE.

3.2 Proposed GBT-PI

The prediction inaccuracy modeling for the residual blocks computed by employing any of the 33 angular modes depicted in Fig. 2.3(a) is based on the work by X. Cai *et al.* in [126]. This modeling approach predicts a residual block using the reference samples to the left and above the block, and the information about the angular mode used by the encoder. The approach is based on the argument that residual blocks computed after angular intra-prediction can be approximated by using the gradient of reference samples and the distance between the position of the reference samples used and the position of the value to be approximated within a block. For horizontal modes, i.e. modes $2-17$ in Fig. 2.3(a) the predicted residual signal, at position (x,y) within an $N \times N$ block is approximated as follows:

$$res(x,y) \approx \frac{P}{cos\alpha} \frac{\delta f(0,y)}{\delta y} \mid_{y'} \theta(x,y) \tag{3.1}$$

where $\theta(x,y)$ is the angle between two consecutive reference samples to the left and depends on the position being predicted, $\frac{\delta f(0,y)}{\delta y}$ denotes a partial derivative with respect to the reference samples to the left, P is the distance between the reference sample $f(0,y')$ and the position to be predicted, which is calculated as $P^2 = R^2 + Q^2$; and α is the angle between P and the horizontal. Fig. 3.1 graphically represents the variables used in Eq. 3.1. For the vertical modes, e.g., modes $18-34$ in Fig. 2.3(a), the same calculations can be applied by appropriately rotating the block.

For the DC and PLANAR modes, we propose an extension to the model proposed in [126]. In the case of the DC mode, we note that residual values tend to increase in the horizontal, vertical and diagonal direction proportionally to the distance from the reference samples, since the predicted value is computed as the average of all reference samples located above and to the left of the block. This is based on the observation that samples in the first row and first column of the block are expected to have a more similar value to that of the reference samples than those samples located far from them. Our proposed modeling approach is based on this observation. Specifically, for the DC mode we propose to average the prediction for the pure horizontal mode (10), pure

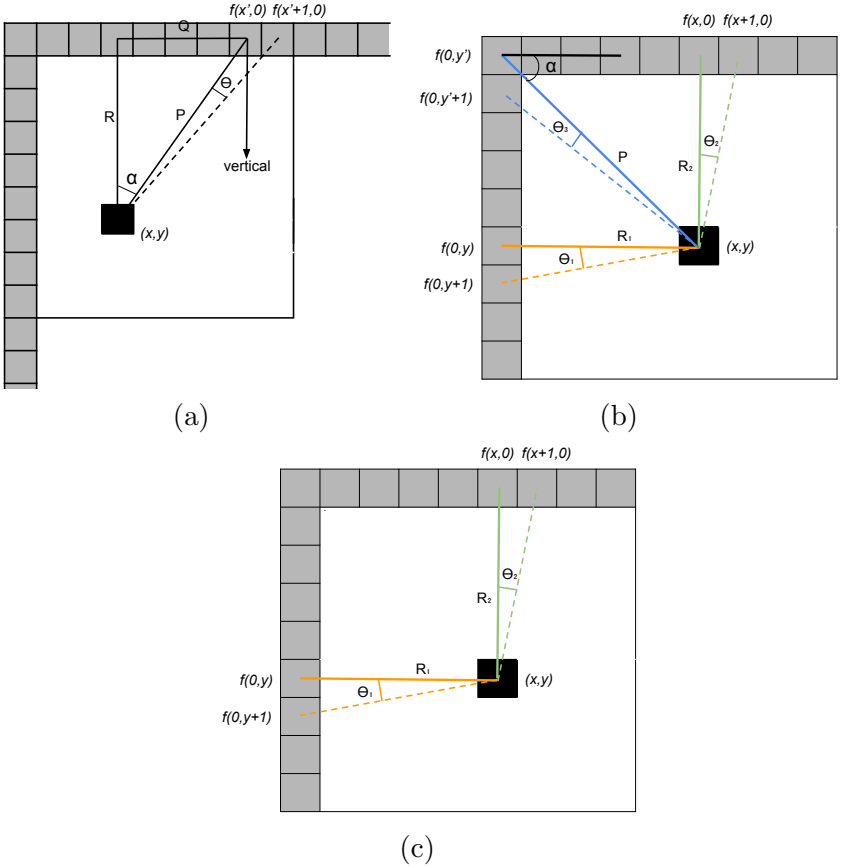


Figure 3.1: Prediction Inaccuracy modeling for (a) Vertical mode, (b) Extended DC mode (c) Extended Planar mode.

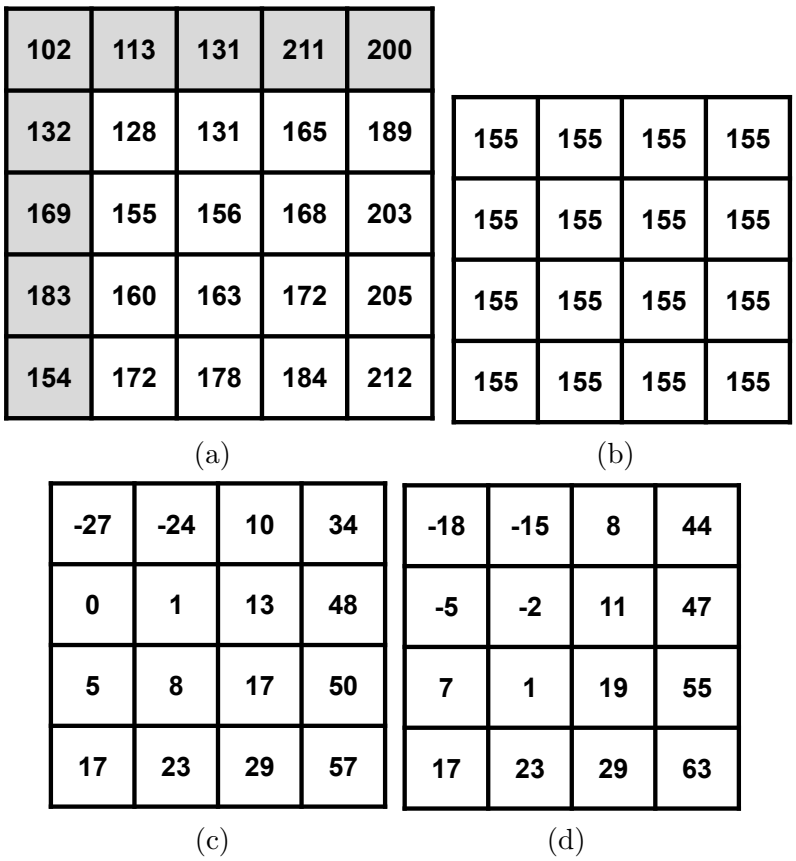


Figure 3.2: Illustration of the proposed modeling approach for the DC mode. The average of all reference samples is 155. (a) Original block, (b) predicted block, (c) actual residual block and (d) predicted residual block.

vertical mode (26), and diagonal mode (18), as follows:

$$resDC(x,y) \approx (resH(x,y) + resV(x,y) + 2.resD(x,y)) >> 2 \tag{3.2}$$

$$resH(x,y) \approx R_1 \frac{\delta f(0,y)}{\delta y} \mid_{y'} \theta_1(x,y) \tag{3.3}$$

$$resV(x,y) \approx R_2 \frac{\delta f(0,y)}{\delta x} \mid_{x'} \theta_2(x,y) \tag{3.4}$$

$$resD(x,y) \approx \frac{P}{cos(\frac{\pi}{4})} \frac{\delta f(0,y)}{\delta y} \mid_{y'} \theta_3(x,y) \tag{3.5}$$

where $resH(x,y)$, $resV(x,y)$, $resD(x,y)$ are the are the predicted residual values in the horizontal, vertical and diagonal directions, respectively, $>>$ represents a bit shift to the right, R_1 and R_2 are the distances between s between the predicted position and the references samples to the left and above, respectively. Note that the modeling approach in Eq. 3.5 is just a case of Eq 3.1 when $\alpha = \frac{\pi}{4}$. The modeling approach for the DC mode is depicted in Fig. 3.1 (b) and exemplified in Fig. 3.2 by using an example 4×4 block, where the average of all reference samples is 155. From Fig. 3.2, it can be observed that the residual signal indeed tends to increase for samples located far from the reference samples. Our prediction inaccuracy modelling effectively approximates the residual based on this observation.

In the case of the PLANAR mode, we follow a similar approach to the one followed for the DC mode. Specifically, we propose to average the prediction for the pure horizontal mode (10) and pure vertical mode (26), as follows:

$$resPlanar(x,y) \approx (resH(x,y) + resV(x,y)) >> 1 \tag{3.6}$$

3.2.1 Proposed framework

Our framework is depicted in Fig. 3.3 and Fig. 3.4. At the encoder side 3.3, we employ a prediction inaccuracy modelling to predict the residual block for each $N \times N$ block by only using the prediction mode selected by the encoder and the references samples of the block [127]. Each predicted residual block is represented by a 4-connected weighted graph and the corresponding GBT is computed by eigendecomposition, as detailed in Section 2.3. This GBT is then used to transform the actual residual block. Coefficients may then be quantized and subsequently entropy coded. At the decoder side, we re-compute the predicted residual block as done by the encoder. Note that this is possible without having to signal any additional information, as the reference samples and prediction mode of each block are readily available at the decoder. Based on the predicted residual block, the corresponding 4-connected weighted graph and GBT is computed, which allow us to compute the inverse GBT to be

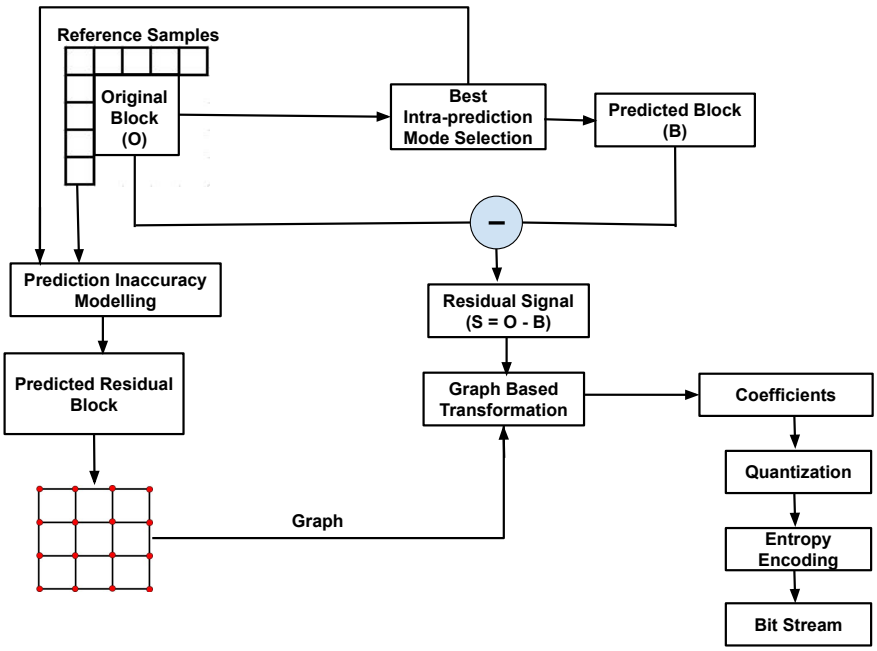


Figure 3.3: Block diagram for proposed encoder framework with PI modeling.

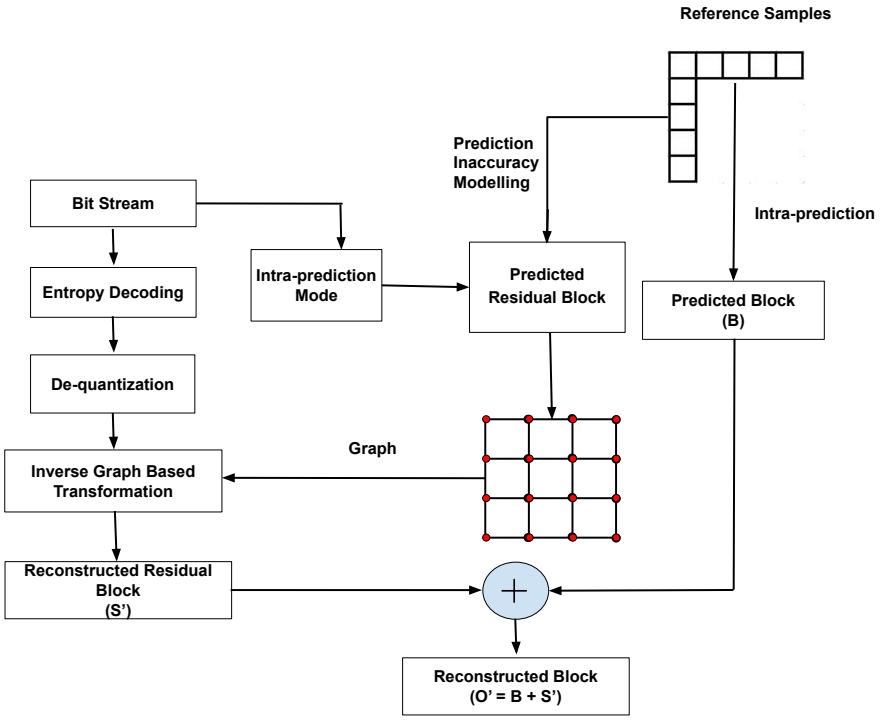


Figure 3.4: Block diagram for proposed decoder framework with PI modeling.

applied to the decoded coefficients (after entropy decoding) and obtain the reconstructed residual block. Finally, the predicted block and re-constructed residual block are added to obtain the reconstructed block.

3.3 Performance evaluation

The proposed framework is tested on ten 1024×1024 sections of WSIs depicting lymphatic, pancreatic, colon and brain tissue. The images are obtained from the Center for Biomedical Informatics and Information Technology of the US National Cancer Institute [128, 129] We employ intra-prediction using all 35 modes depicted in Fig. 2.3 (a), with a block size of 8×8 on the G component. We compare the performance of the GBT using our framework (GBT_PI) against the GBT when the graphs are computed using the actual residual blocks (GBT_A), the DCT and the DST, as implemented in HEVC. The performance of all transforms is measured in terms of the energy preserved by reconstructing the image using a sub-set of the largest coefficients and the corresponding MSE. In other words, we evaluate the energy compaction properties of the transforms and the quality of the reconstructed images. The coefficients are selected by setting a threshold that indicates the minimum absolute value that the coefficients in the sub-set must have (See Fig. 3.5). A large threshold allows to include the largest coefficients in the sub-set, while a threshold close to zero results in including most of the coefficients in the sub-set. By gradually decreasing an initial large threshold, this approach gradually includes in the sub-set the largest coefficients. Note that this approach differs from one that selects the DC and low frequency AC coefficients first, and gradually include the high frequency AC coefficients. The approach used in this work allows selecting the largest coefficients, regardless of their frequency type. This is advantageous for pathology images, as they usually depict strong edges and non-smooth regions, resulting in several AC coefficients with large values. Our evaluations also include the KLT, as the baseline transform. Table 3.1 tabulates the average energy preserved, in percentage, by the transforms and the corresponding average MSE values, for all evaluated images, using a small percentage of coefficients of 5% and 10%.

As expected, the KLT provides the best performance. Note that the GBT_A outperforms the GBT_PI. This is also expected as the GBT_A is constructed based on the graphs of the actual residual signals. However, it is important to recall that the GBT_A requires signaling information about the graphs to the decoder. The GBT_PI outperforms the DST in terms of preserved energy, which is used in HEVC for intra-predicted residuals. For example, on average, 11.47% more energy can be preserved by the GBT_PI than the DST by using only 1.0% of the coefficients. In terms of MSE values, the GBT_PI attains lower

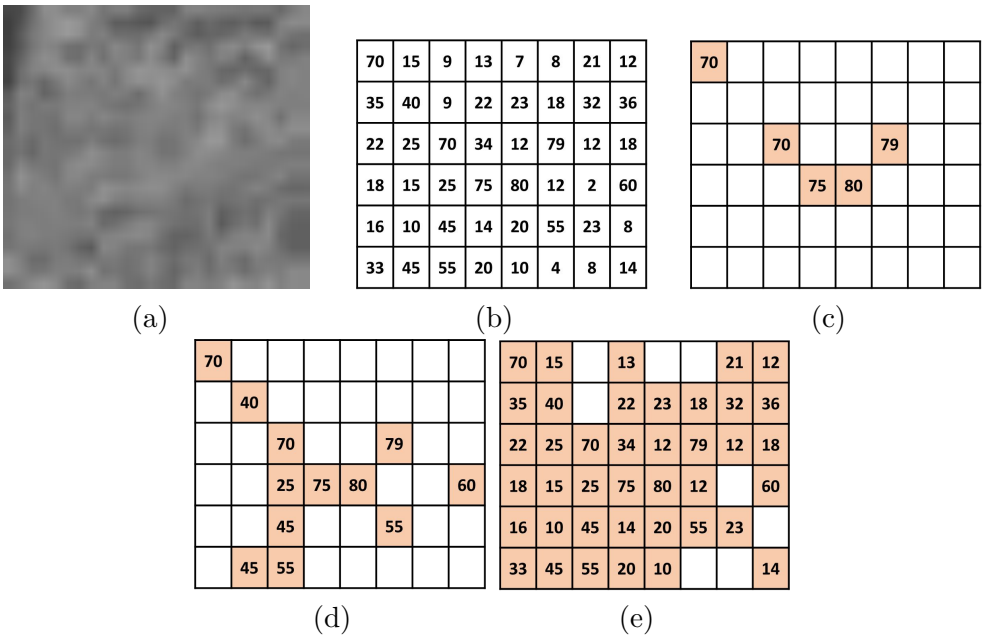


Figure 3.5: (a) A block of a pathology image. (b) Coefficients values of the image in (a). (c) Sub-set of coefficients for threshold value 70. (d) Number of coefficients in the sub-set increased for decreasing the threshold to 40 (e) Majority of the coefficients are in the sub-set when the threshold value is lowered to 10.

values than the DST for most of the cases tabulated in Table 3.1. Note that in average the GBT_PI slightly outperforms the DCT in terms of preserved energy, for all percentages of coefficients tabulated in Table 3.1. However, the corresponding average MSE values are slightly higher than those attained by DCT. Fig. 3.6 plots the percentage of preserved energy vs. the percentage of coefficients used for four different images. The improvements of the GBT_PI over the DCT/DST and the DCT can be visually appreciated in these plots.

3.4 Summary

WSIs are multigiga-pixel color images that usually require large amounts of bandwidth to be transmitted and stored. To facilitate the widespread of these images in clinical settings, compression is needed to reduce storage and bandwidth requirements. Block-based PTC using intra-prediction has been shown to be capable of efficiently compress these images. In this chapter, we presented a framework that allows employing the GBT to transform intra-predicted residual signals of these images without the need to signal information about the graphs to the decoder. The framework is based on computing the necessary graphs using predicted residual blocks, which can be re-computed at the decoder using only the reference samples and information about the

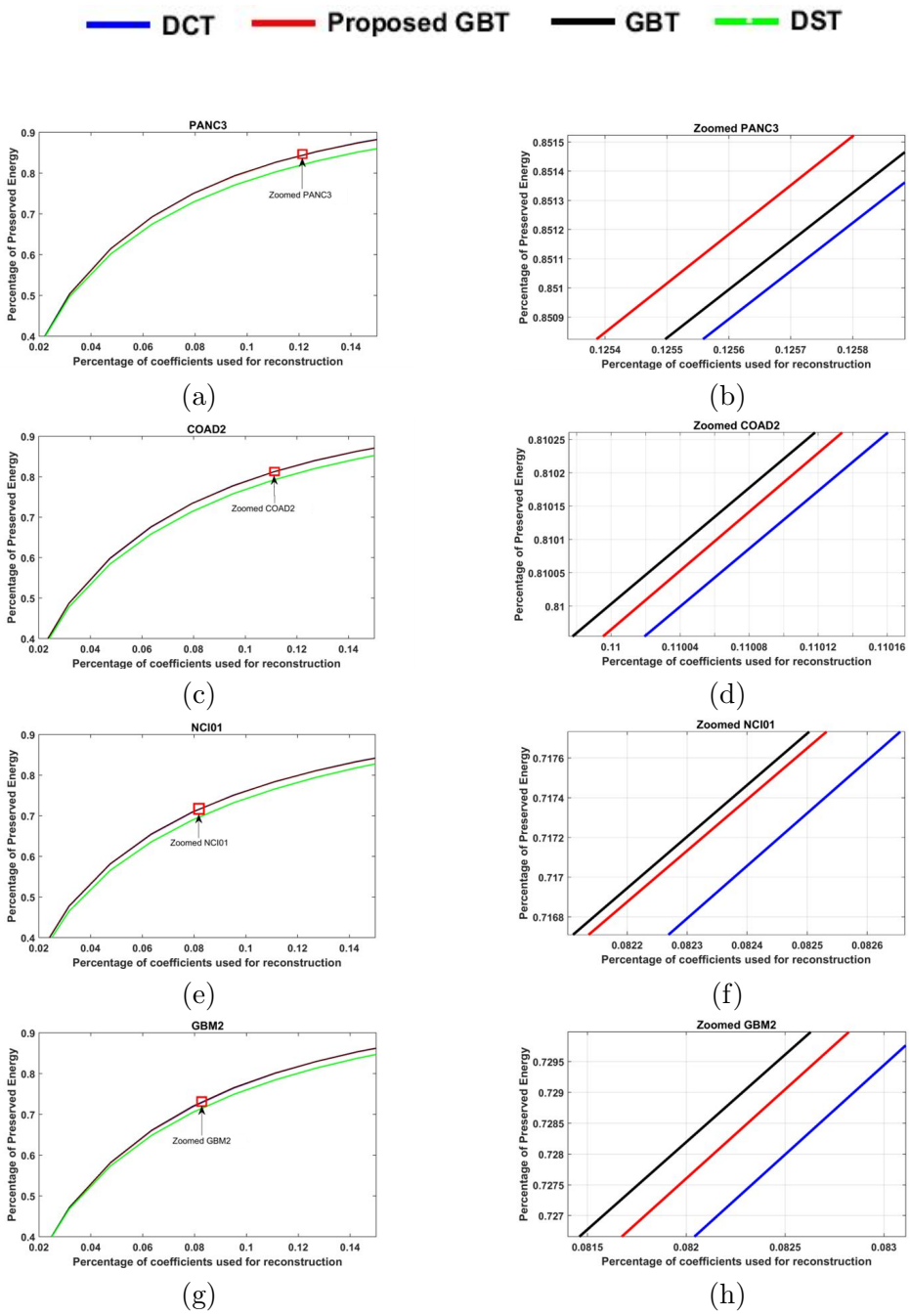


Figure 3.6: Energy compaction performance of different transforms for digital pathology images depicting (a-b) pancreatic tissue, (c-d) colon tissue, (e-f) lymphatic tissue, and (g-h) brain tissue.

prediction mode used. We evaluated the performance of the GBT computed in this fashion in terms of the energy preserved and MSE when a small percentage of the largest coefficients are used for reconstruction of several pathology images. Evaluation results show that the GBT can outperform the DST, while slightly outperforming the DCT, in terms of energy preserved, for the evaluated images.

In this chapter, we have restricted ourselves to using the collection of pathology images for the purpose of our research. We included other HEVC sequences for the same method in the following chapter to assess the outcome because we observed only modest improvement in our proposed method when compared to DCT. Additionally, since the connection of our graphs in this chapter is only available in a 4-connected form, we intend to put greater emphasis on various graph connectivity in the next chapters.

Table 3.1: PE (in %) and MSE using a small percentage of the largest coefficients.

Percentage of coefficients used for reconstruction																
	KLT				GBT_PI				GBT_A				DCT				DCT/DST			
	PE↑		MSE ↓		PE ↑		MSE ↓		PE↑		MSE ↓		PE ↑		MSE ↓		PE ↑		MSE ↓	
	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%
Pathology Image																				
Lymphatic Tissue																				
NCI01	88.2	92.0	11.4	09.6	58.4	76.8	40.7	22.2	58.6	77.0	40.9	22.6	**60.97**	**8.1**	**38.62**	**2.0**	59.4	76.2	40.7	24.3
NCI20	88.1	91.9	11.2	09.3	**58.27**	**6.7**	40.5	22.4	58.4	76.8	40.6	22.4	60.8	78.0	**38.32**	**1.8**	59.3	76.0	40.5	24.2
NCI26	89.7	91.8	10.5	07.7	**59.77**	**6.8**	**39.92**	**1.7**	59.8	76.7	39.9	22.3	57.8	75.9	41.0	22.3	57.1	74.4	41.2	23.2
Brain Tissue																				
GBM2	88.0	91.7	11.0	09.2	58.1	76.8	40.3	22.2	58.1	76.7	40.4	21.1	**60.67**	**7.9**	**38.22**	**1.5**	59.2	76.0	40.4	24.0
GBM3	89.5	91.7	10.3	07.7	**59.57**	**6.7**	**39.72**	**1.7**	59.6	76.7	39.7	22.3	57.7	75.9	40.9	22.4	56.9	74.4	41.0	23.2
Pancreatic Tissue																				
PANC3	91.0	93.4	11.8	09.4	**61.07**	**8.4**	**41.2**2	4.4	61.1	78.4	41.2	24.0	59.1	77.5	42.3	**24.0**	58.4	76.0	42.5	24.9
PANC2	91.6	93.6	12.4	09.6	**61.67**	**8.6**	**41.8**2	4.6	61.7	78.6	41.8	24.2	59.8	77.8	43.0	**24.2**	59.0	76.3	43.1	25.0
Colon Tissue																				
COAD1	89.1	92.9	12.2	10.6	58.9	77.6	41.4	23.3	58.9	77.4	41.1	23.2	**61.87**	**9.0**	**39.52**	**2.9**	60.1	76.9	41.6	25.7
COAD2	91.7	93.6	12.5	09.6	**61.77**	**8.6**	**41.9**2	4.5	61.8	78.6	41.9	24.1	59.9	77.8	43.1	**24.2**	59.1	76.3	43.2	25.0
COAD3	91.2	93.7	12.0	09.8	**61.27**	**8.7**	**41.4**2	4.8	61.3	78.7	41.4	24.4	59.4	77.9	42.5	**24.4**	58.6	76.4	42.7	25.2
Avg of pathology	89.8	92.6	11.5	09.2	**59.87**	**7.6**	40.9	23.2	59.9	77.6	40.9	23.2	59.7	77.5	**40.72**	**3.0**	58.7	75.9	41.7	24.5

Chapter 4

Graph-Based Transform with Weighted Self-loops based on Template Matching

4.1 Introduction

In general, lossy compression is based on applying an orthogonal transform on the signal to expand it into a set of orthogonal bases, with the expectation that most of the signal's information is captured by a few basis functions. This is followed by quantization of the resulting coefficients. For any arbitrary signal with a known covariance function, it is well known that the KLT is the linear transform with the best energy compaction property. The KLT basis functions of natural images are close to those of the DCT [125]. Hence, the DCT is widely considered as the best transform for image compression. Unfortunately, the DCT offers little adaptability to the characteristics of the data as a fixed transform is usually applied to all images.

The GBT [14] is proposed as an attractive option to address some of the issues of the DCT. Thanks to the fact that the GBT accounts for the data correlation through the use of a graph structure, it has excellent data de-correlation and energy compaction properties. Recently, Pavez *et al.* [130]

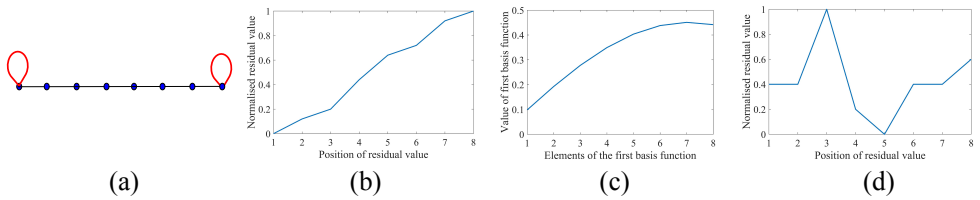


Figure 4.1: (a) Line graph with self-loops in the first and last vertices. (b) 1D residual signal predicted by the horizontal mode. (c) First basis function of a GBST designed for the horizontal mode. (d) 1D residual signal predicted by the horizontal mode for a noisy signal.

showed that several variants of the 1D DCT and 1D Discrete Sine Transform (DST) can be computed as a GBT based on a line graph with unit edge weights and self-loops in the first and last vertices (see Fig. 4.1(a)). Based on this fact, the authors learn the self-loop weights that produce efficient GBSTs for block-based PTC of intra-predicted video frames. They show how the first basis function of their learned transforms can accurately represent the residual signal. This is exemplified in Fig. 4.1(b)-(c), where the first basis function of their learned GBST for the horizontal prediction mode is plotted. One can easily note that the function vanishes on the left side and increases on the right side. This behavior resembles the shape of this *ideal* 1D residual signal, in which the error is expected to be small in the leftmost pixel location (i.e., the one closest to the reference pixel) and increase with the distance from the reference. In practice, however, the residual signals may not always have an ideal behavior. For example, a row of residual values computed by the horizontal prediction mode may have a relatively flat shape if the image is smooth, or several peaks and valleys if the image is noisy (see Fig. 4.1(d)). A GBT whose first basis accurately represents the residual signal, irrespective of the prediction mode, has the potential to provide better data de-correlation and energy compaction properties.

This chapter thus proposes the GBT-L, a novel class of GBT based on a 2D graph with unit edge weights and weighted self-loops in every vertex. The GBT-L accurately captures the characteristics of a residual block by computing the self-loop weights according to the residual values. Since the GBT-L is based on a 2D graph, it accounts for the correlation among all values to be transformed. To avoid signaling additional information required to compute the inverse GBT-L within the context of block-based PTC, we also propose a coding framework that uses a template-based strategy to predict the residual blocks to be transformed. The GBT-L is evaluated on a wide range of video frames and medical images. Our results show that the GBT-L attains better energy compaction properties and a higher reconstruction quality than the DST, DCT and the GBST.

The rest of this chapter is organized as follows. Section 4.2 describes the proposed GBT-L. Section 4.2.1 explains the coding framework that integrates the template-based strategies to predict residual blocks. Section 4.3 presents and discusses the performance evaluation results and Section 4.4 concludes this chapter.

4.2 Proposed GBT-L

Let us consider an image to be encoded using block-based PTC via angular intra-prediction, which is a common prediction method used in many modern

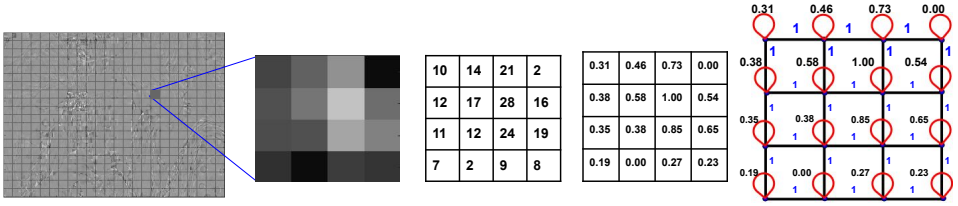


Figure 4.2: (a) Residual signal generated by intra-prediction for the Y component of a video frame from the sequence *BlowingBubbles*. (b),(c) A sample 4×4 residual block and its actual values. (d) Normalized residual values. (e) 2D graph (4-connect) with unit edge weights and self-loops in each vertex.

video codecs, including the High HEVC standard [11, 131]. For each block in the imaging data, intra-prediction yields a residual block computed as the difference between the predicted and the original block. Let us recall such a (square) residual block as $\mathbf{S} \in \mathbb{R}^{\sqrt{N} \times \sqrt{N}}$, where N is the total number of residual values. The block $\mathbf{S}$ 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 all edges, and $\mathbf{A} \in \mathbb{R}^{N \times N}$ is the symmetric weighted adjacency matrix. The entry $\mathbf{A}_{ij}$ in $\mathbf{A}$ represents the weight of the edge e_{ij} connecting vertices v_i and v_j. The GBT-L assumes a 4-connected pattern with self-loops in every vertex as shown in Fig. 4.2. Consequently, $\mathbf{A}_{ij} = 1$ for $i \neq j$, i.e., the weight of any edge connecting two adjacent nodes is always 1. The self-loop weights, i.e., the diagonal entries of $\mathbf{A}$, are computed based on the normalized residual values. For node v_i, A_{ii} is given by:

$$A_{ii} = \frac{v_i^r - \min V}{\max V - \min V}, \tag{4.1}$$

where v_i^r is the residual value of v_i and $\min V$, $\max V$ are the minimum and maximum residual value of the nodes in set V.

The GBT-L is constructed by the eigendecomposition of the generalized Laplacian, $\mathbf{L}$, computed as $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D}$ is the diagonal degree matrix, whose n^{th} diagonal element is equal to the sum of the weights of all edges incident onto node v_n. The eigendecomposition of $\mathbf{L}$ is used as the orthogonal transform for the residual block, since it has a complete set of eigen-vectors with real, non-negative eigen-values. Fig. 4.2 shows a residual frame with 4×4 blocks and the 4-connected graph with self-loops in each vertex for a sample block. Note that the self-loop weights are between 0 and 1.

In order to attain excellent data de-correlation and energy compaction properties, the GBT-L is based on a 2D graph. Moreover, the first basis function of the GBT-L should accurately resemble the behavior and overall-shape of the residual signal. The work in [130] shows that the eigendecomposition of a Laplacian of a line graph with unit edge weights and no self-loops corresponds to the DCT. By setting the self-loop weight to 1 for the first vertex of such line graph, the resulting transform is equivalent to the DST-7. That work also

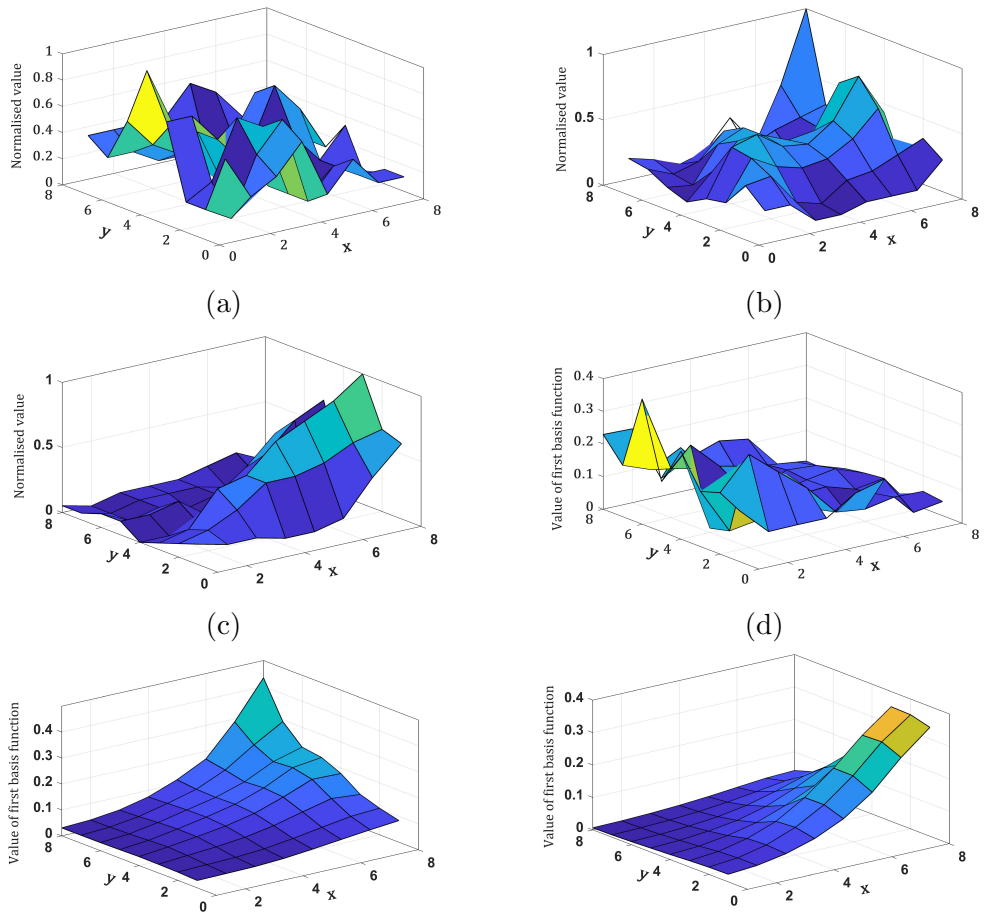


Figure 4.3: Normalized residual values of an 8×8 block computed by the (a) DC, (b) vertical, and horizontal modes. (d-f) First basis function of the corresponding GBT-L.

shows that by varying the self-loop weights of the first and last vertices, one can produce GBTs whose first basis function closely resembles the characteristics of the residual data being transformed. This is the motivation behind adding a self-loop weight to each node of the 2D graph used by the GBT-L. These weights are computed based on the residual values, as specified by Eq. (4.1). Fig. 4.3 shows the 2D plot of various 8×8 residual blocks and the 2D plot of the first basis function of their corresponding GBT-L. One can note that the first basis function indeed resembles the residual signal and follows its general shape. This, as will be shown in Section 4, allows to preserve more of the signal's energy with only a few coefficients.

4.2.1 Proposed coding framework

As mentioned in previous section, each variant of GBT requires sending additional information to the decoder to reconstruct the 2D graph needed to compute the inverse GBT-L. To tackle this issue, we propose a coding framework that does not require sending such additional information.

Our framework is depicted in Fig. 4.4. At the encoder side, we employ a template-based strategy [95, 132] to predict each $\sqrt{N} \times \sqrt{N}$ residual block by only using the previously encoded and reconstructed blocks. Each predicted residual block is represented as a 2D graph with self-loops in each vertex and unit weight edges following a 4-connected pattern. The GBT-L is then computed based on this graph and used to transform the actual residual block. By following such a prediction strategy, it is possible to recover the residual block at the decoder without signalling any additional information, as the exact same prediction can be performed at the decoder [6] (see Fig. 4.4(b)). Specifically, based on the residual block predicted by the encoder, the same 2D graph can be computed to obtain the inverse GBT-L.

We propose two different template-based strategies to predict residual blocks as follows: template matching and weighted template pooling.

4.2.2 Template matching

Template matching searches for the most similar blocks to the target block based on the similarity of their templates, where the template of a block is the area surrounding the block to the left and above [96, 97]. Fig. 4.5 depicts a sample target template, denoted by $\mathbf{x}$, and the corresponding target block to be predicted, denoted by $\mathbf{P}$. The target template is estimated by using the $k = 5$ most similar candidate templates, $\mathbf{t}_1, \mathbf{t}_2, ...\mathbf{t}_k$. We use the sum of absolute differences (SAD) between the target template and a candidate template as the criterion to select these k templates. $\mathbf{P}$ is then predicted as a weighted average of the candidate blocks $\mathbf{P}_1, ...\mathbf{P}_k$, one for each of the k-most similar

candidate templates.

4.2.3 Weighted template pooling

Weighted template pooling uses a weighted average of all the previously encoded and reconstructed blocks to predict the target block. The weights used to average these blocks are computed based on the similarity of their templates with the target template, in terms of the SAD. We use templates of 4 rows and 4 columns, which results in 72 samples surrounding a block to the left and above (see Fig. 4.5(b)). The higher the similarity among the target and the candidate templates used for prediction, the higher the prediction accuracy of the target block.

We perform the template-based strategies in **two domains**: the residual, and the pixel domain.

4.2.4 Template-based prediction in the residual domain

The prediction of the target residual block is performed by using the residual signals of previously encoded and reconstructed blocks. This is illustrated in Fig. 4.6, where one can see that all candidate templates and blocks contain residual signals. For the case of template matching, we first use optimization by least square approximation to estimate, from the k-most similar candidate templates, the target template:

$$\min_{\mathbf{w}} \|\mathbf{x} - \mathbf{T}\mathbf{w}\|_2^2 \text{ s.t.} \sum_k w_k = 1, \tag{4.2}$$

where vector $\mathbf{x}$ contains the residual values of the target template, matrix $\mathbf{T}$ contains the residual values of the k-most similar candidate templates, and $\mathbf{w} = [w_1, \cdots, w_k]$ is a weight vector. The nth target residual block, $\mathbf{P_n}$, is then predicted as as $\tilde{\mathbf{P}}_\mathbf{n}$ by using the k-most similar candidate blocks, as follows:

$$\tilde{\mathbf{P}}_\mathbf{n} = w_1\mathbf{P}_1 + w_2\mathbf{P}_2 + ... + w_k\mathbf{P}_k. \tag{4.3}$$

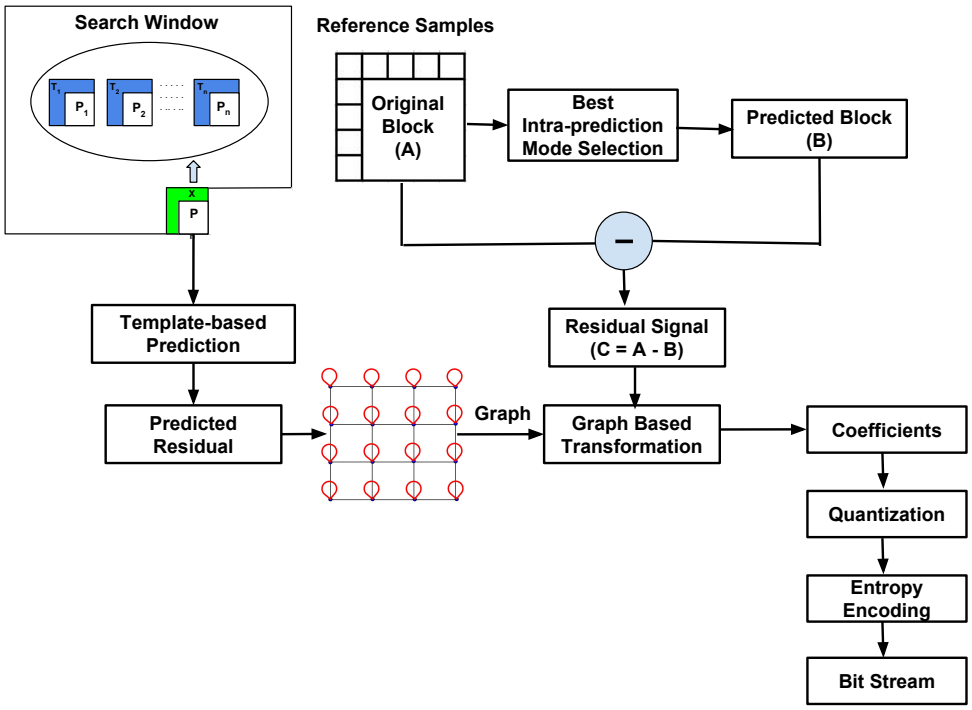
For the case of the weighted template pooling strategy, the residual signals of the $n-1$ previously encoded and reconstructed blocks are used to predict the nth target residual block, as follows:

$$\tilde{\mathbf{P}}_\mathbf{n} = w_1\mathbf{P}_1 + w_2\mathbf{P}_2 + ... + w_{n-1}\mathbf{P}_{n-1}, \tag{4.4}$$

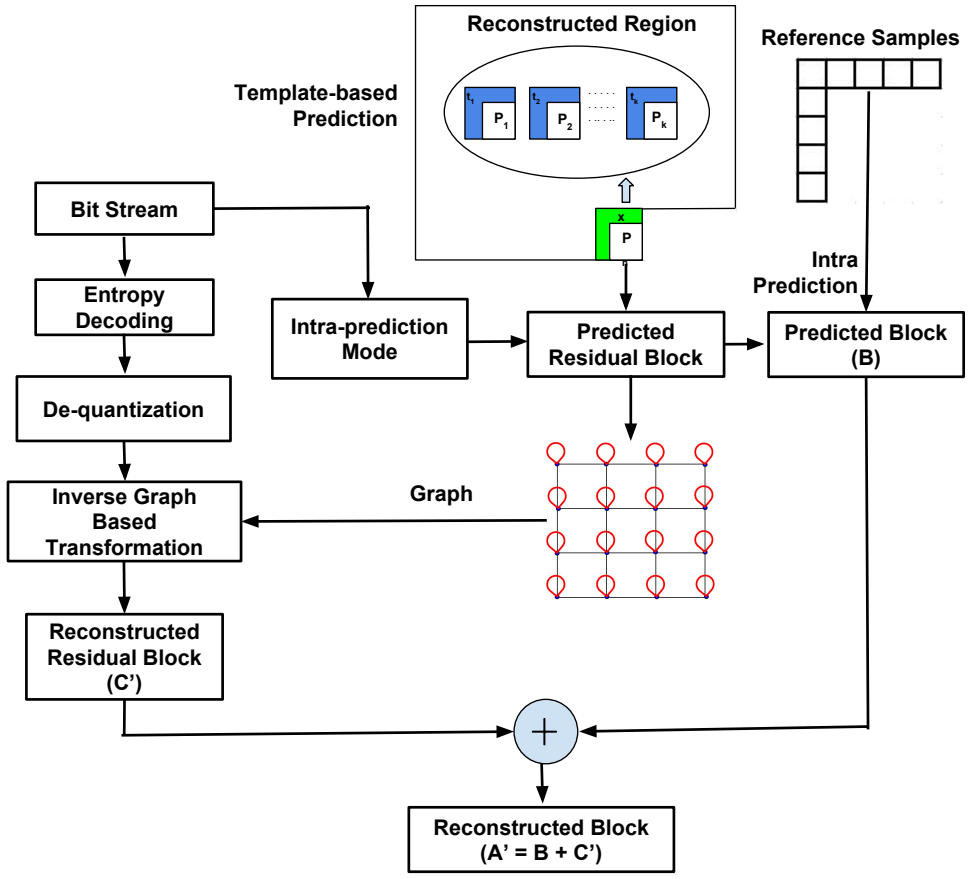
where the weight for the jth candidate block is:

$$w_j = e^{\frac{\|\mathbf{x}-\mathbf{t}_j\|_2^2}{h^2}}, \tag{4.5}$$

where $\mathbf{t}_j$ is the j^{th} candidate template and h is the average of standard deviation of the samples of the $j-1$ candidate templates.



(a)



(b)

Figure 4.4: Block diagram of proposed framework for (a) Encoding and (b) Decoding.

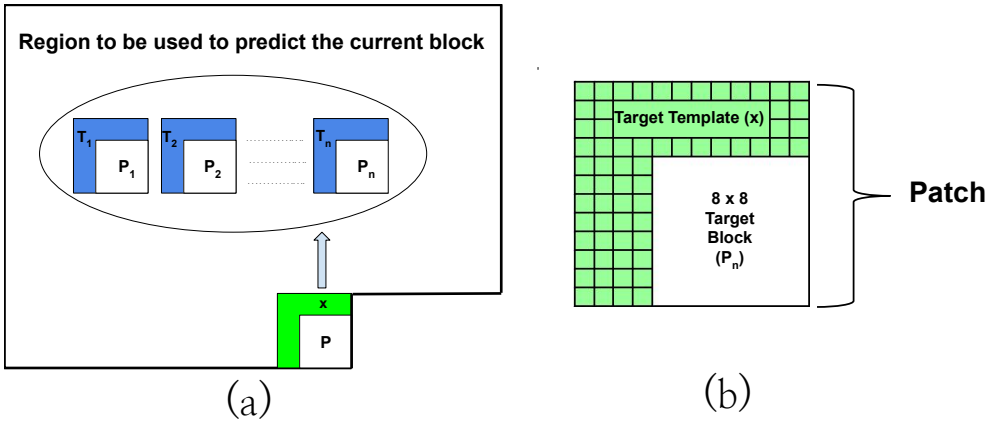


Figure 4.5: (a) Search window used to find blocks to predict the target block. (b) Sample target template and target block.

4.2.5 Template-based prediction in the pixel domain

The prediction of the target residual block is performed by using the previously encoded and reconstructed blocks. In other words, the target block is first predicted in the pixel domain. This predicted target block is then subtracted from the corresponding predicted block computed by angular intra-prediction to compute the predicted residual block. This is illustrated in Fig. 4.7. For the case of template matching, we first compute $\tilde{\mathbf{I}}_\mathbf{n}$, the predicted block for the nth target block in the pixel domain, denoted by $\mathbf{I_n}$, as follows:

$$\tilde{\mathbf{I}}_n = w_1\mathbf{I}_1 + w_2\mathbf{I}_2 + ... + w_k\mathbf{I}_k, \tag{4.6}$$

where $\mathbf{I}_1$,$\mathbf{I}_2$,...,$\mathbf{I}_k$ are the k-most similar candidate blocks in the pixel domain, and w_1,w_2,...,w_k are the weights as computed by Eq. (4.2). Note that in the pixel domain, Eq. (4.2) uses templates comprising pixel values instead of residual values.

For the case of weighted template pooling, we predict $\mathbf{I_n}$ using the $n-1$ previously coded and reconstructed blocks, as follows:

$$\tilde{\mathbf{I}}_n = w_1\mathbf{I}_1 + w_2\mathbf{I}_2 + ... + w_{n-1}\mathbf{I}_{n-1}, \tag{4.7}$$

where weights w_1,w_2,...,w_{n-1} are computed by Eq. (4.5) with candidate templates comprising pixel values. We subtract $\tilde{\mathbf{I}}_n$ from the corresponding predicted block computed by angular intra-prediction to produce the predicted residual block, $\tilde{\mathbf{P}}_n$.

4.3 Performance evaluation

The proposed GBT-L and our coding framework are evaluated on 30 different YUV frames of standard test video sequences of class A, B, C, D, E and screen

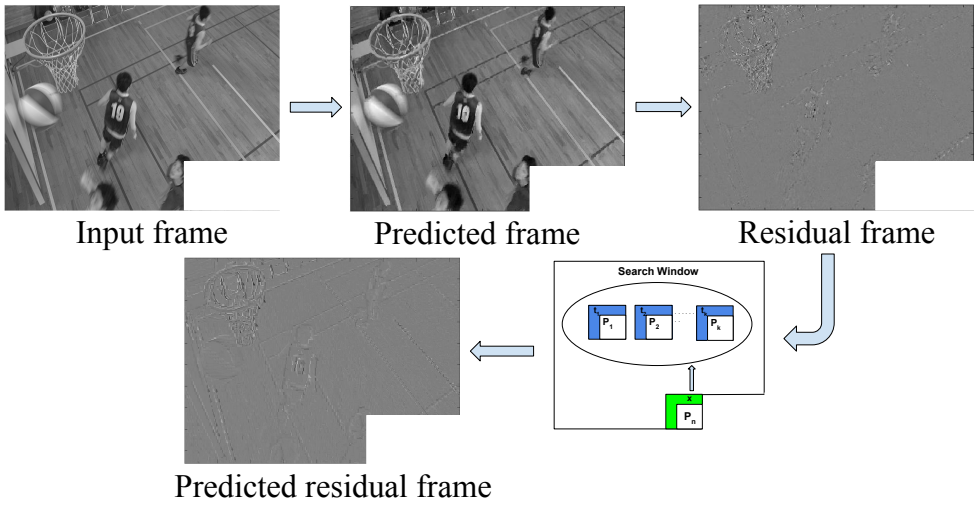


Figure 4.6: Template-based prediction in the residual domain.

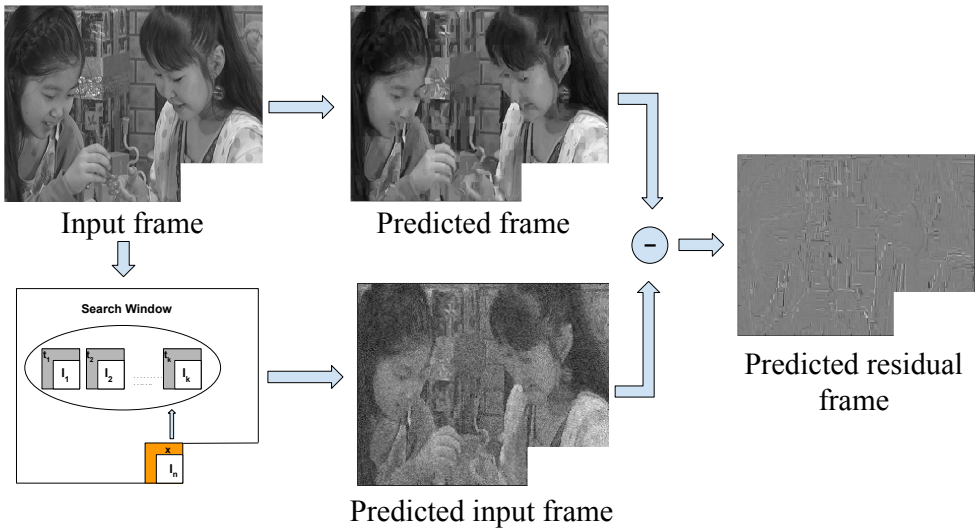


Figure 4.7: Template-based prediction in the pixel domain.

content(SC). We also use pathology images in RGB format from the Center for Biomedical Informatics and Information Technology of the US National Cancer Institute in the evaluation. We use the 35 intra-predicton modes of HEVC to compute the residual blocks. We use blocks of 8×8 pixels on the Y and G components of the video frames and pathology images, respectively.

We compare the performance of the GBT-L using template matching in the residual (GBT-L$_{T_{res}}$), and the pixel domain (GBT-L$_{T_{pix}}$), the GBT-L using weighted template pooling in the residual (GBT-L$_{W_{res}}$), and the pixel domains (GBT-L$_{W_{pix}}$), the GBT-L using the PI modelling (GBT-L$_{PI}$), the DCT, and the DST as implemented in HEVC. The performance of all transforms is measured in terms of the percentage of preserved energy (PE) by reconstructing the image using a sub-set of the largest coefficients, and the corresponding MSE. No quantization is used to clearly understand the advantages of each transform in terms of energy compaction and reconstruction error using the largest coefficients. 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. Consequently, we do not follow any conventional scanning pattern as commonly done in modern codecs. This strategy allows selecting the largest coefficients, regardless of their frequency type.

Our evaluations also include three baseline transforms: the KLT, the GBT-L$_A$ when the graphs are computed using the actual residual blocks (GBT-L$_A$), and a GBST with self-loops in every vertex, whose weights are computed by Eq. 4.1 using actual residual values. Note that these baselines require the signaling of additional information to compute the corresponding inverse transforms. Evaluation of the GBT-L$_A$, however, allows confirming the advantages of using 2D graphs with self-loops in every vertex and unit edge weights. Evaluation of the GBST allows confirming the advantages of using 2D graphs to design the transform.

Table 4.1 presents the PE (%) and MSE values for the evaluated data using a small percentage of coefficients of all the proposed transforms using weighted self-loops. Table 4.2 consists of the the PE (%) and MSE values of popular transforms we set as baseline of our experiments. We compare both the tables to evaluate the performances. As expected, the KLT provides the best performance. Compared to the GBST, the GBT-L$_A$ attains higher PE and lower MSE values. This confirms the advantages of constructing GBTs using 2D graphs. For example, by using only 5% of the largest coefficients , the GBT-L$_A$ can preserve 4.87% more energy than the GBST. The GBT-L$_A$ also outperforms the DCT and DST. The GBT-L$_A$ can preserve 19.83% and 17.47% more energy than the DST and DCT, respectively, by using only 5%

of the largest coefficients. This confirms the advantages of using self-loops in every vertex of the 2D graph.

On average, the GBT-L$_{W_{pix}}$ attains the best performance among the transforms that require signaling no additional information to the decoder. The GBT-L$_{W_{pix}}$ preserves 3.11% and 1.08% more energy than the DST and DCT if only 5% of the coefficients are used. Note that the GBT-L, when paired with template-based prediction in the residual domain, tends to perform poorly compared to using template-based prediction in the pixel domain. Predicting residual values is more challenging than predicting pixel values, as residual signals involve signed values [131]. Consequently, the template-based strategies in the residual domain are expected to attain less accurate predictions, hindering the performance of the GBT-L. In other words, the performance of the GBT-L is expected to improve as the prediction accuracy of the residual blocks improves. We can see that for class A, B, and C frames and pathology images, the GBT-L$_{W_{pix}}$ attains the best performance among those that require no extra signalling information. For class D and E, the DCT is the best transform for some cases. Frames of these two classes depict several smooth regions. The DCT is then well-suited for this content, as it approximates the KLT basis functions of natural images. We summarize the average results reported in Table 4.1 and Table 4.2 in the Table 4.3 for easier analysis and a more simplified as well as obvious point of view. Fig. 4.8 plots the PE (%) and MSE values vs. the percentage of coefficients used for reconstruction of a video frame, where the MSE values are normalized with respect to the maximum value attained when no coefficients are used for reconstruction. Note that the GBT-L$_A$ clearly outperforms the DCT and DST. The GBT-L$_{W_{pix}}$ indeed outperforms all other transforms that require no extra signalling information.

4.4 Summary

In this chapter, we proposed the GBT-L, a new class of GBT constructed based on a 2D graph with unit edge weights and weighted self-loops in every vertex. We showed that the first basis function of the GBT-L closely resembles the residual block to be transformed, which allows to preserve more energy by using a small percentage of the largest coefficients. We also presented a coding framework that allows employing the GBT-L on intra-predicted residual blocks without the need to signal information about the graphs to the decoder. The framework uses template-based strategies to predict the residual blocks in the residual or pixel domains. We evaluated the performance of the GBT-L in terms of the PE (%) and MSE when a small percentage of the largest coefficients are used for reconstruction. Evaluation results show that,as the prediction accuracy of the residual blocks improves, the improvements of the

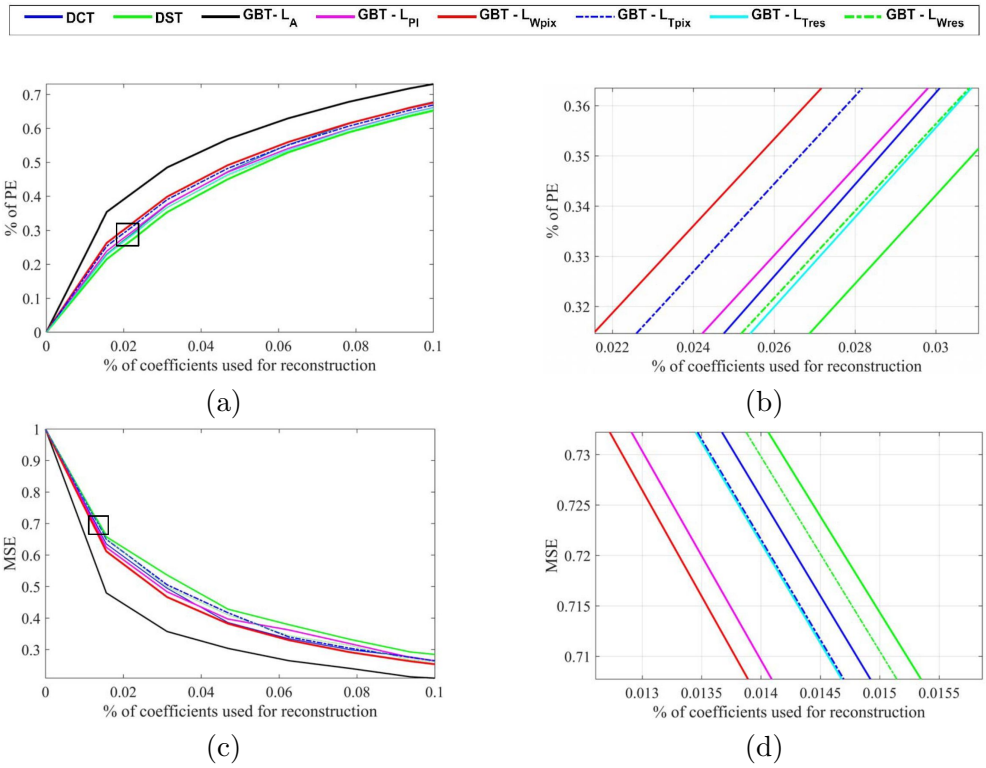


Figure 4.8: (a,b) PE (%) and (c,d) MSE vs. percentage of coefficient used for reconstruction of a frame of sequence *KristenAndSara*.

GBT-L over the DCT, DST and the GBST also increase. When only 5% of the largest coefficients are used, the GBT-L, when computed based on actual residual blocks, can preserve up to 19.83% and 17.47% more energy than the DST and DCT, respectively.

To summerize, in Chapter 3 and Chapter 4 we explored non-learning based approach to predict the data. Now a days, we are quite aware, for any ML/DL approach, it makes life easier among researchers to predict the data since it can readily process enormous amounts of data and identify patterns and relationships. Since the objective of this thesis is to predict the graph for inverse GBT to perform at the decoder, we have explored the DL approach in our following chapters. The revised strategy of using DL should produce better outcomes, as anticipated.

Table 4.1: PE (in %) and MSE using a small percentage of the largest coefficients for proposed template-based methods.

Sequence	*Percentage of coefficients used for reconstruction*															
	Template Matching								Template Pooling						PI	
	GBT-L$_{T_{res}}$				GBT-L$_{T_{pix}}$				GBT-L$_{W_{res}}$				**GBT-L$_{\mathbf{W_{pix}}}$**		GBT-L$_{PI}$	
	PE↑		MSE ↓		PE ↑		MSE ↓		PE↑		MSE ↓		PE ↑		MSE ↓		PE ↑		MSE ↓	
	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%
4:2:0 YUV sequences																
Class A																
Traffic	64.4	80.4	38	20.5	64.4	80.4	38.6	21.4	64.4	81.5	37.5	20.6	65.2	82.6	37.3	20.1	64.4	82.1	37.8	20.6
People_on_street	64.1	80.2	37.7	20.4	64.1	80.2	38.3	21.1	64.1	81.4	37.3	20.3	64.6	82.3	37	19.5	64.4	81.5	37.5	20.3
Nebuta_festival	62.7	82.7	36.3	19.1	62.4	82.4	36.6	19.4	62.4	80.1	36.1	18.6	63.5	80.6	35.6	18.4	63	80.4	36.1	18.6
Avg of Class A	63.7	81.1	37.3	20.0	63.6	81.0	37.8	20.6	63.6	81.0	37.0	19.8	64.4	81.8	36.6	19.3	63.9	81.3	37.1	19.8
Class B																
Kimono	40.6	58.3	61.8	45.2	39.0	56.8	64.0	47.4	40.6	58.3	62.0	45.4	42.7	60.5	59.3	42.8	39.1	56.9	63.3	46.7
Cactus	38.6	56.4	64.1	47.3	41.3	58.9	62.0	45.5	38.6	56.4	64.3	47.5	40.7	58.6	61.7	44.9	41.5	59.0	61.3	44.7
Park_scene	38.6	56.1	62.6	45.9	39.8	57.5	62.0	45.2	38.6	56.1	62.8	46.1	40.7	58.2	60.2	43.5	39.9	57.7	61.3	44.4
BQTerrace	40.6	58.0	64.9	48.1	42.2	59.7	64.0	47.1	40.6	58.0	65.1	48.3	42.7	60.1	62.5	45.6	42.3	59.8	63.3	46.3
Avg of Class B	39.6	57.2	63.4	46.6	40.6	58.2	63.0	46.3	39.6	57.2	63.6	46.8	41.7	59.4	60.9	44.2	40.7	58.3	62.3	45.5
Class C																
Race_horse	43.4	59.3	56.7	40.4	47.3	63.2	52.7	36.5	43.3	59.2	57.0	40.8	47.6	63.5	52.3	36.1	44.8	60.7	55.3	39.0
BQMall	45.8	61.4	54.7	38.5	45.3	61.3	55.0	38.6	45.6	61.3	55.0	38.9	45.6	61.6	54.7	38.3	47.1	62.8	53.3	37.1
Party_scene	44.2	60.1	54.7	38.2	45.3	60.9	53.5	37.2	44.1	59.9	55.0	38.5	45.6	61.2	53.2	36.9	45.6	61.4	53.3	36.8
Basketball_drill	46.6	62.2	56.7	40.1	47.3	62.8	55.9	39.4	46.5	62.1	57.0	40.4	47.6	63.2	55.5	39.0	48.0	63.5	55.3	38.7
Avg of Class C	45.0	60.7	55.7	39.3	46.3	62.0	54.3	37.9	44.9	60.6	56.0	39.7	46.6	62.4	53.9	37.6	46.4	62.1	54.3	37.9
Class D																
Race_horse_D	51.0	68.0	48.7	32.0	48.5	65.6	51.1	34.4	48.1	65.2	51.4	34.7	48.6	65.7	51.0	34.3	51.1	68.2	48.4	31.7
Blowing_bubble	49.0	66.1	51.1	34.1	50.9	67.7	49.2	32.5	50.5	67.3	49.4	32.8	51.0	67.8	49.0	32.4	49.1	66.3	50.8	33.8
BQ_square	49.0	65.7	49.5	32.8	49.3	66.3	49.2	32.1	49.0	65.9	49.4	32.4	49.5	66.5	49.0	32.0	49.2	65.9	49.3	32.5
Basketball_pass	51.0	67.7	51.9	34.9	51.7	68.4	51.2	34.0	51.3	68.1	51.4	34.3	51.8	68.6	51.0	33.9	51.2	67.8	51.6	34.6
Avg of Class D	50.0	66.9	50.3	33.4	50.1	67.0	50.2	33.3	49.7	66.6	50.4	33.6	50.2	67.1	50.0	33.2	50.1	67.0	50.0	33.1
Class E																
Kristine_and_Sara	57.9	74.1	45.0	28.1	59.9	75.6	42.0	25.6	58.3	74.1	42.9	26.4	59.7	75.5	41.9	25.5	59.0	75.2	43.5	26.7
Four_people	59.5	75.5	44.8	28.0	59.6	75.5	43.6	27.0	58.1	74.0	44.4	27.8	59.5	75.4	43.5	26.9	60.6	76.6	43.3	26.5
Jhonny	58.0	73.7	43.9	27.5	58.7	75.0	42.2	25.2	57.2	73.5	43.0	26.0	58.6	74.9	42.1	25.1	59.1	74.8	42.4	26.0
Avg of Class E	58.5	74.4	44.6	27.9	59.4	75.4	42.6	25.9	57.9	73.9	43.4	26.8	59.3	75.3	42.5	25.8	59.5	75.5	43.1	26.4
Class F/SC																
China_speed	44.3	60.4	56.3	40.7	48.3	64.5	52.3	36.6	46.7	62.8	53.9	38.3	45.4	61.5	55.2	39.6	45.1	61.2	55.4	39.7
Slide_show	46.6	62.5	54.3	38.7	46.3	62.5	54.6	38.8	44.7	60.9	56.2	40.4	47.7	63.6	53.2	37.7	47.4	63.3	53.4	37.8
Sc_Map	45.1	61.2	54.3	38.4	46.3	62.2	53.1	37.4	44.7	60.6	54.7	39.0	46.2	62.3	53.2	37.3	45.9	62.0	53.4	37.5
Sc_Programming	47.5	63.3	56.3	40.3	48.3	64.1	55.5	39.5	46.7	62.5	57.1	41.1	48.6	64.4	55.2	39.2	48.3	64.1	55.4	39.4
Avg of Class F/SC	45.9	61.9	55.3	39.5	47.3	63.3	53.9	38.1	45.7	61.7	55.5	39.7	47.0	63.0	54.2	38.5	46.7	62.7	54.4	38.6
Pathology Image																
NCI01	61.0	81.6	37.2	20.1	60.1	78.2	37.3	24.8	59.4	78.6	40.5	24.3	59.7	77.8	38.9	24.3	62.2	81.2	36.9	21.0
NCI20	59.0	79.7	39.6	22.2	62.5	80.3	35.3	22.9	61.7	80.7	38.5	22.4	62.0	80.0	36.9	22.3	60.2	79.3	39.2	23.1
GBM2	59.0	79.4	38.0	20.9	60.9	79.0	35.3	22.5	60.2	79.4	38.5	22.0	60.5	78.6	36.9	22.0	60.3	78.9	37.7	21.8
COAD1	61.0	81.3	40.4	23.0	63.3	81.1	37.3	24.4	62.6	81.5	40.5	23.9	62.9	80.7	38.9	23.9	62.3	80.8	40.0	23.9
Avg of pathology	60.0	80.5	38.8	21.5	61.7	79.7	36.3	23.7	61.0	80.1	39.5	23.2	61.3	79.3	37.9	23.1	61.3	80.1	38.5	22.4
Overall average	51.8	69.0	49.3	32.6	52.7	69.5	48.3	32.3	51.8	68.7	49.4	32.8	52.9	69.7	48.0	31.7	52.7	69.6	48.5	32.0

Table 4.2: PE (in %) and MSE using a small percentage of the largest coefficients for the methods to compare with Table 4.1.

Sequence	*Percentage of coefficients used for reconstruction*																			
	Baseline												Popular HEVC transforms							
	KLT				GBST				GBT-L_A				DCT				DCT/DST			
	PE↑		MSE ↓		PE ↑		MSE ↓		PE↑		MSE ↓		PE ↑		MSE ↓		PE ↑		MSE ↓	
	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%
4:2:0 YUV sequences																				
Class A																				
Traffic	89.1	93.1	13.9	12.1	68.6	86.3	32.9	18.1	73.7	84.5	24.3	14.6	65.2	82.3	38.9	21.2	63	80	40.1	24.9
People_on_street	88.8	92.4	12.1	11.4	67.5	86	31.3	17.2	74.7	85.1	25.3	15.3	63.1	82.4	38.1	20.3	62.2	78.8	39	23.4
Nebuta_festival	87.2	90.9	13.6	10.8	66.8	83.4	30	16.8	75.8	86.7	26.5	16.8	63.2	78.5	34.4	20.1	58.5	75.1	39.3	21.4
Avg of Class A	88.4	92.2	13.2	11.5	67.7	85.2	31.4	17.4	74.8	85.4	25.4	15.5	63.8	81.1	37.2	20.6	61.3	78	39.5	23.2
Class B																				
Kimono	93.2	98.3	14.8	12.4	46.4	65.9	58.2	42.5	49.7	63.2	54.1	41.1	41.6	60.0	63.7	46.4	41.4	58.2	63.1	47.4
Cactus	92.6	97.4	13.1	12.2	45.8	64.7	56.2	42.1	49.1	62.0	53.5	39.9	40.4	58.1	62.6	46.0	40.3	57.1	62.4	46.0
Park_scene	92.1	94.8	13.5	10.9	45.5	63.7	56.0	40.4	49.2	62.6	53.5	40.6	40.1	58.1	62.0	45.1	40.4	57.4	61.7	45.9
BQTerrace	93.0	95.4	11.6	10.5	46.3	63.2	53.1	39.5	49.8	62.1	54.2	40.0	41.5	56.4	59.9	44.2	39.9	56.3	61.9	44.8
Avg of Class B	92.7	96.5	13.3	11.5	46.0	64.4	55.9	41.1	49.5	62.2	53.8	40.2	40.9	58.1	62.0	45.4	40.5	57.3	62.3	46.0
Class C																				
Race_horse	92.0	95.9	9.5	7.9	54.3	67.3	50.9	30.0	53.0	71.4	45.2	34.8	46.3	63.2	57.0	39.1	46.3	62.2	58.2	42.2
BQMall	89.8	93.3	9.7	7.1	52.8	65.0	50.2	29.3	52.4	70.2	44.6	33.6	46.0	62.8	55.9	38.8	45.1	61.0	55.1	41.7
Party_scene	90.3	94.2	8.9	6.9	52.6	65.1	48.9	28.8	52.4	70.9	44.7	34.2	45.0	62.1	55.0	38.2	44.1	60.3	56.0	40.0
Basketball_drill	94.5	98.1	8.7	7.7	53.0	66.4	47.8	30.0	53.1	70.3	45.3	33.7	43.1	61.4	53.8	39.2	41.1	60.1	57.6	38.0
Avg of Class C	91.6	95.4	9.1	7.4	53.2	65.9	49.5	29.5	52.7	70.5	45.0	33.8	45.1	62.4	55.4	38.8	44.2	60.9	56.7	40.5
Class D																				
Race_horse_D	92.0	95.0	11.6	9.6	59.3	72.5	44.9	26.8	61.2	72.4	37.4	30.1	51.6	68.7	50.4	33.7	50.6	67.1	51.3	35.6
Blowing_bubble	90.7	94.5	10.0	8.4	57.3	68.4	45.6	26.5	60.6	71.2	36.8	28.9	50.2	67.4	49.6	33.2	49.5	66.2	50.8	34.3
BQ_square	90.2	94.2	9.8	8.1	57.0	68.2	45.2	26.2	60.6	71.9	36.8	29.6	50.1	67.3	49.3	33.0	49.1	66.0	50.1	34.1
Basketball_pass	89.8	94.0	9.3	7.6	57.0	68.4	44.9	25.5	61.3	71.5	37.5	29.0	50.5	67.9	49.4	32.4	49.2	66.1	51.4	34.7
Avg of Class D	90.7	94.4	10.2	8.4	57.7	69.4	45.1	26.3	60.9	71.5	37.1	29.2	50.6	67.8	49.7	33.1	49.6	66.4	50.9	34.7
Class E																				
Kristine_and_Sara	87.8	91.7	14.8	13.1	65.2	82.6	34.3	21.1	70.1	81.0	30.6	20.6	58.9	76.0	42.0	25.4	58.2	75.0	43.2	27.0
Four_people	87.6	91.5	14.4	13.0	65.1	82.4	35.8	20.9	69.7	80.3	30.2	20.0	58.8	75.9	41.9	25.3	58.1	74.6	43.1	26.9
Jhonny	88.5	91.9	15.8	13.6	65.7	82.6	35.4	22.3	69.9	81.8	30.4	21.5	59.4	75.9	42.7	27.2	58.7	75.7	43.7	27.3
Avg of Class E	87.9	91.7	15.0	13.2	65.3	82.5	35.2	21.4	69.9	81.0	30.4	20.7	59.0	76.3	42.2	25.6	58.3	75.1	43.3	27.1
Class F/SC																				
China_speed	89.0	92.4	14.0	12.1	54.7	70.6	48.3	31.0	55.3	68.9	43.1	31.7	46.4	63.6	54.4	37.9	50.1	63.1	55.3	39.0
Slide_show	88.3	91.9	13.5	11.4	54.3	70.3	47.8	30.9	54.7	67.6	42.5	30.4	46.1	63.4	54.2	37.7	50.0	62.9	55.0	38.9
Sc_Map	87.9	90.9	12.3	10.9	54.1	69.8	46.9	30.5	54.7	68.3	42.5	31.1	45.9	63.2	54.0	37.3	49.9	62.7	54.8	38.5
Sc_Programming	87.2	92.2	12.6	11.1	54.7	70.3	46.8	30.4	55.4	67.7	43.2	30.5	45.8	63.1	53.7	37.1	49.8	62.2	54.7	38.5
Avg of Class F/SC	88.1	91.9	13.1	11.3	54.5	70.3	47.4	30.7	55.0	67.9	42.8	30.7	46.1	63.3	54.1	37.5	50.0	62.7	55.0	38.7
Pathology Image																				
NCI01	88.2	92.0	11.4	9.6	66.3	83.7	31.2	19.2	71.1	83.3	21.7	17.4	60.9	78.1	38.6	22.0	59.4	76.2	40.7	24.3
NCI20	88.1	91.9	11.2	9.3	66.2	83.6	30.9	18.9	70.9	82.1	21.5	17.3	60.8	78.0	38.3	21.8	59.3	76.0	40.5	24.2
GBM2	88.0	91.7	11.0	9.2	66.0	83.5	29.7	18.7	71.5	82.7	22.1	18.0	60.6	77.9	38.2	21.5	59.2	76.0	40.4	24.0
COAD1	89.1	92.9	12.2	10.6	67.1	84.6	32.2	18.6	71.4	82.2	22.0	17.4	61.8	79.0	39.5	22.9	60.1	76.9	41.6	25.7
Avg of pathology	88.4	92.1	11.4	9.7	66.4	83.9	31.0	18.8	71.2	82.3	21.8	17.6	61.0	78.3	38.7	22.1	59.5	76.3	40.9	24.6
Overall average	89.7	93.4	12.2	10.4	58.7	74.5	42.2	26.5	62.0	74.4	36.6	26.8	52.4	69.6	48.5	31.9	51.9	68.1	49.8	33.5

Table 4.3: 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	89.73	12.22	93.43	10.43
GBST	18.56	81.92	58.71	42.24	74.54	26.49
GBT-L$_A$	**25.15**	**74.72**	**62.04**	**36.61**	**74.39**	**26.76**
DCT	17.49	82.27	52.41	48.48	69.58	31.88
DCT/DST	16.94	82.74	51.89	49.81	68.14	33.49
GBT-L$_{T_{res}}$	17.11	82.58	51.78	49.32	69.01	32.56
GBT-L$_{T_{pix}}$	17.67	81.98	52.69	48.34	69.46	32.33
GBT-L$_{W_{res}}$	17.20	82.54	51.78	49.34	68.67	32.79
GBT-L$_{W_{pix}}$	**17.67**	**81.97**	**52.88**	**48.04**	**69.68**	**31.74**
GBT-L$_{PI}$	17.56	82.17	52.67	48.45	69.61	32.00

Chapter 5

Graph-Based Transform based on Neural Networks for Intra-Prediction of Imaging data

5.1 Introduction

This chapter 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.

When the GBT is used in block-based PTC, in order to reconstruct the block at decoder, 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 chapter proposes a GBT based on a neural network approach (GBT-NN) to avoid sending such extra information. 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 this end, since templates are already available to the decoder this framework does not require sending additional information to perform the inverse transform. The novelty of this work states, it learns a graph Laplacian for various set of intra-prediction modes in both encoder and decoder, where only need to send information about prediction mode and quantization parameter as overhead. 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.

5.2 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 the original block [46]. $\mathbf{S}$ 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 eigendecomposition 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 [133]. The connectivity and the edge weights of the graph are generally inferred from the data (see Fig. 2.8).

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:

$$\mathbf{A}^b \approx f(\mathbf{A}^p), \tag{5.1}$$

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. (5.1) is based on an encoding-decoding NN, as illustrated in Fig. 5.1 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

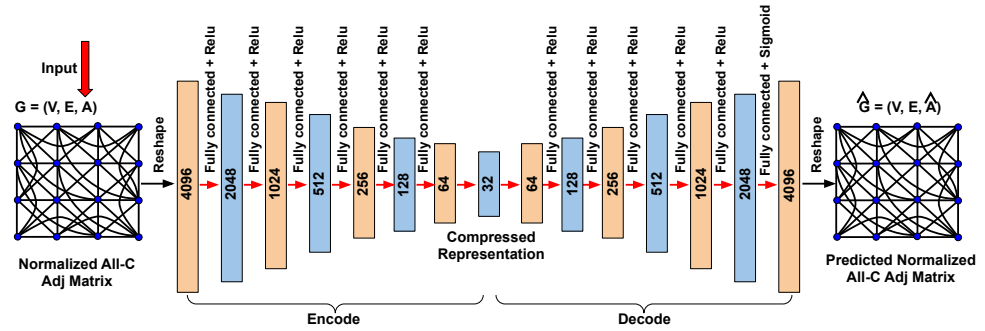


Figure 5.1: Architecture of the proposed GBT-NN for 8×8 blocks and a normalized all-connected (All-C) symmetric adjacency matrix.

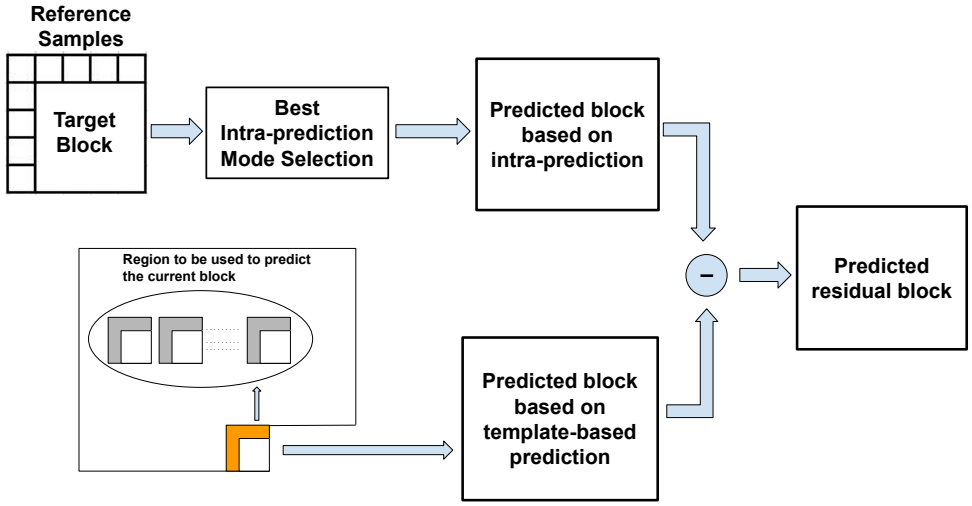


Figure 5.2: Template-based prediction (TBP).

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. 5.1 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 $[0, 1]$.

5.2.1 Prediction strategy

The matrix used as input to the network is generated from a residual block predicted by the template-based prediction strategy as in Chapter 4. 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. 4.5 (a)). The weight assigned as Eq. 4.5 in Chapter 4 to the reconstructed block. We use the same templates of 4 rows and 4 columns, which results in 72 samples surrounding an 8×8 block to the left and above (see Fig. 4.5(b)). The predicted current block is subtracted from the corresponding predicted block computed by intra-prediction to compute a predicted residual block (see Fig. 5.2). 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:

$$\mathbf{h}^{(l_e)} = ReLU(\mathbf{W}^{(l_e)}\mathbf{h}^{(l_e-1)}), \tag{5.2}$$

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:

$$\hat{\mathbf{a}}^b = Sigmoid(\mathbf{W}^{(l_d)}\mathbf{h}^{(l_d-1)}), \tag{5.3}$$

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. 5.1 differs from that of an autoencoder, as our NN does not reconstruct the same input.

5.2.2 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:

$$L = L_{recon} + \alpha L_{sym} + \lambda \parallel \mathbf{W}(:) \parallel_1, \tag{5.4}$$

where $\parallel . \parallel$ 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:

$$L_{recon} = \parallel \hat{\mathbf{a}}^b - \mathbf{a}^b \parallel_2^2, \tag{5.5}$$

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:

$$L_{sym} = \frac{\parallel \hat{\mathbf{a}}^{b(anti)} \parallel_1}{\parallel \hat{\mathbf{a}}^{b(sym)} \parallel_1 + \parallel \hat{\mathbf{a}}^{b(anti)} \parallel_1}, \tag{5.6}$$

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.

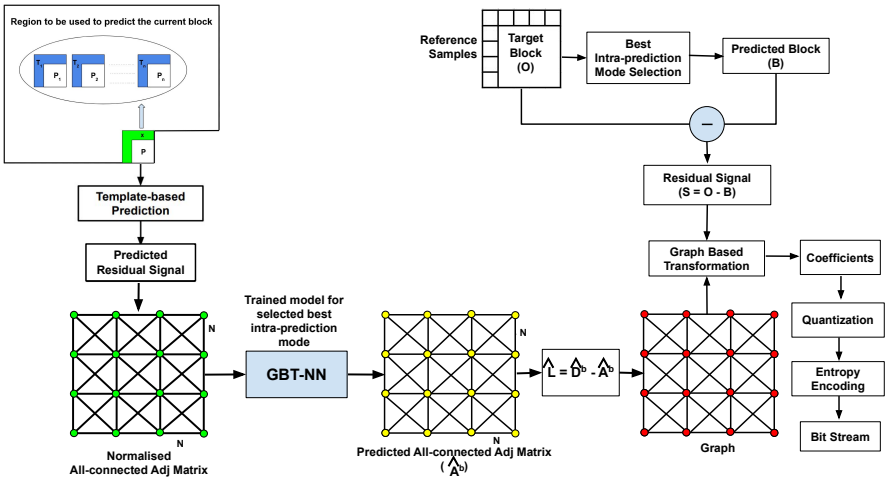


Figure 5.3: Block diagram of the proposed framework for encoding.

5.2.3 Coding/decoding framework based on the proposed GBT-NN

Our framework for coding/decoding is depicted in Fig. 5.3 and Fig. 5.4, respectively. Our framework for coding/decoding avoids signalling overhead.

To reconstruct the current block, the same graph used to compute the GBT should be used to compute the inverse GBT. 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. 5.5). 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. 5.6 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.

5.3 Performance evaluation

5.3.1 Experimental setup

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

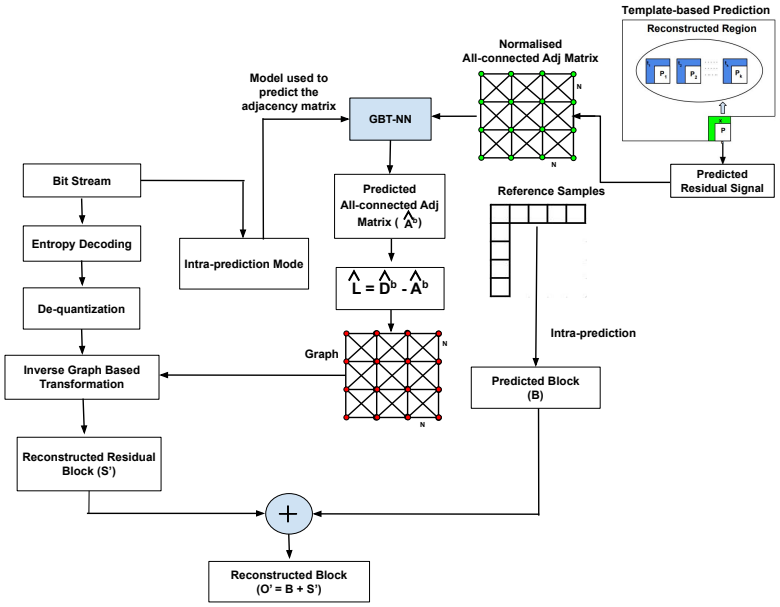


Figure 5.4: Block diagram of the proposed framework for decoding.

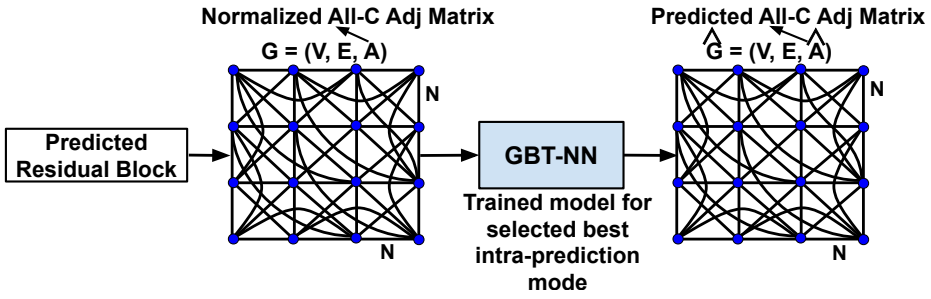


Figure 5.5: GBT-NN used to produce a predicted symmetric adjacency matrix for the current residual block.

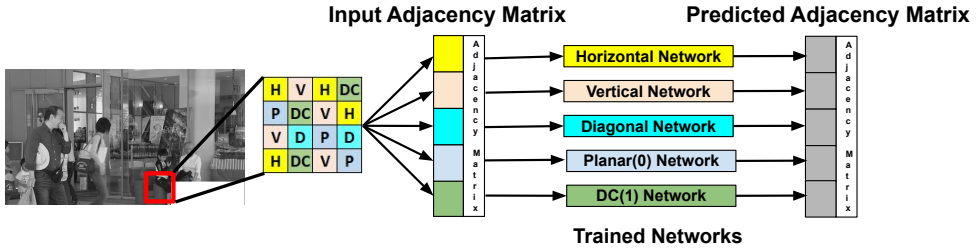


Figure 5.6: A specific trained network is selected based on the intra-prediction mode used for each block. The figure shows a section of a frame predicted by several prediction modes.

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. 5.4). 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

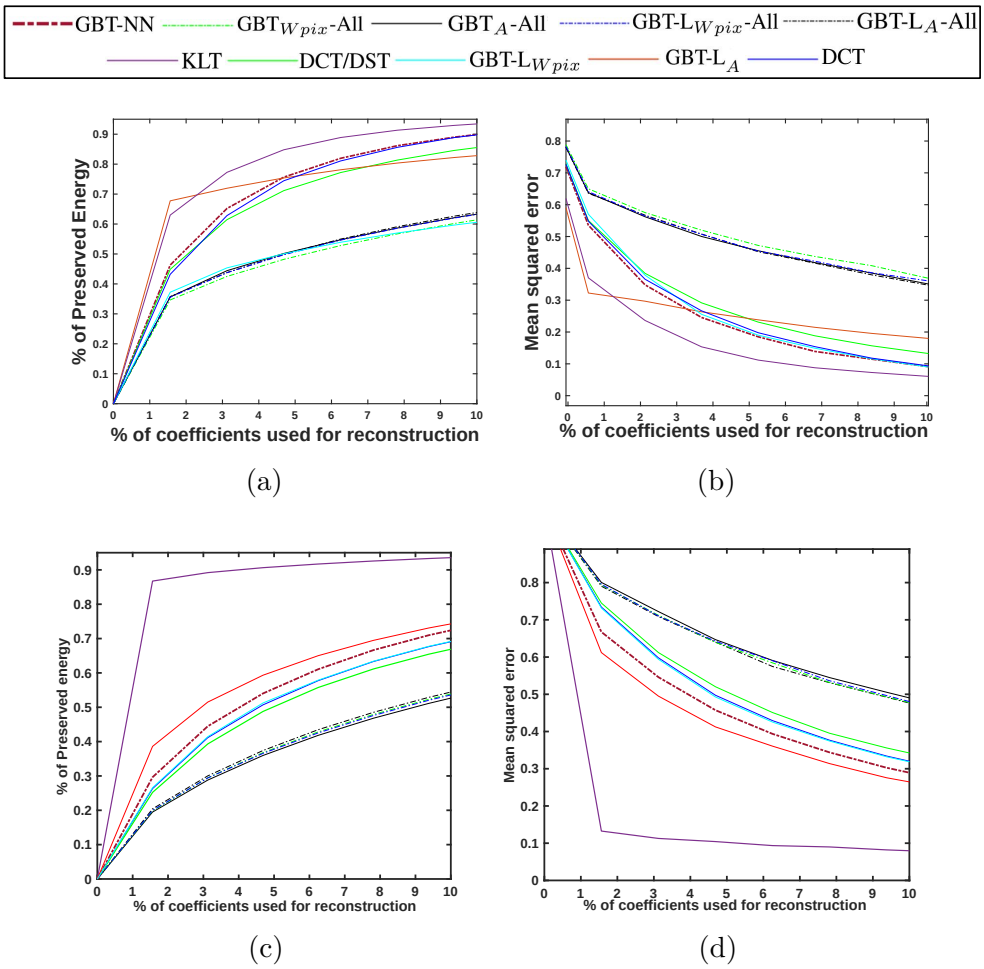


Figure 5.7: (a,c) PE (%) and (b,d) MSE vs. percentage of coefficients used for reconstruction of (1^{st} row) a frame of sequence *PeopleOnStreet* (Class A) and (2^{nd} row) a frame of sequence *BQTerrace* (Class B).

Screen Content, which are video sequences widely used to test modern video codecs [19]. 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 [128]. 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 5.1 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.

Table 5.1: Types of graph used to construct the GBT.

Approach	Explanation
	All-Connected Topology
GBT_A-All	Edge weights defined by a Gaussian kernel using the residual (no self-loops).
GBT-L_A-All	Unit edge weights and normalised self loop weights using the actual residual [130].
GBT_{Wpix}-All	Edge weights defined by a Gaussian kernel using the residual predicted by weighted template pooling.
GBT-NN	**Our proposed approach (unit edge weights but no self-loops).**
GBT-L_{Wpix}-All	Unit edge weights and normalised self loop weights using the residual predicted by weighted template pooling.
	4-Connected Topology
GBT-L_A	Unit edge weights and normalised self loop weights using the actual residual [130].
GBT-L_{Wpix}	Unit edge weights and normalised self loop weights using the residual predicted by weighted template pooling.

Table 5.2: Performance evaluation of the model on test data for all the networks.

	Horizontal		**Vertical**		**Diagonal**		**DC**		**Planar**	
Metric	L	L_{rcon}	L	L_{rcon}	L	L_{rcon}	L	L_{rcon}	L	L_{rcon}
MSE	165.94	271.10	170.4	258.59	169.65	234.98	184.28	284.16	156.52	258.5
MAE	4.66	5.29	4.35	6.43	5.44	6.91	8.40	15.4	5.21	6.99
Ψ	0.99	0.94	0.97	0.93	0.92	0.82	0.95	0.88	0.98	0.93

5.3.2 Model evaluation

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 [134, 135]:

$$\Psi = \frac{|\hat{\mathbf{a}}^{b(sym)} \|_1 - \| \hat{\mathbf{a}}^{b(anti)} \|_1}{\| \hat{\mathbf{a}}^{b(sym)} \|_1 + \| \hat{\mathbf{a}}^{b(anti)} \|_1} \in [-1, 1], \tag{5.7}$$

where a value of 1 means perfect symmetry.

Table 5.2 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.

5.3.3 Results

Table 5.3 presents the average PE (%) and MSE values for all evaluated data using a small percentage of coefficients. As expected, the KLT (Table. 4.2) 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} [7] (Proposed in Chapter 4), 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. 5.8 plots the PE

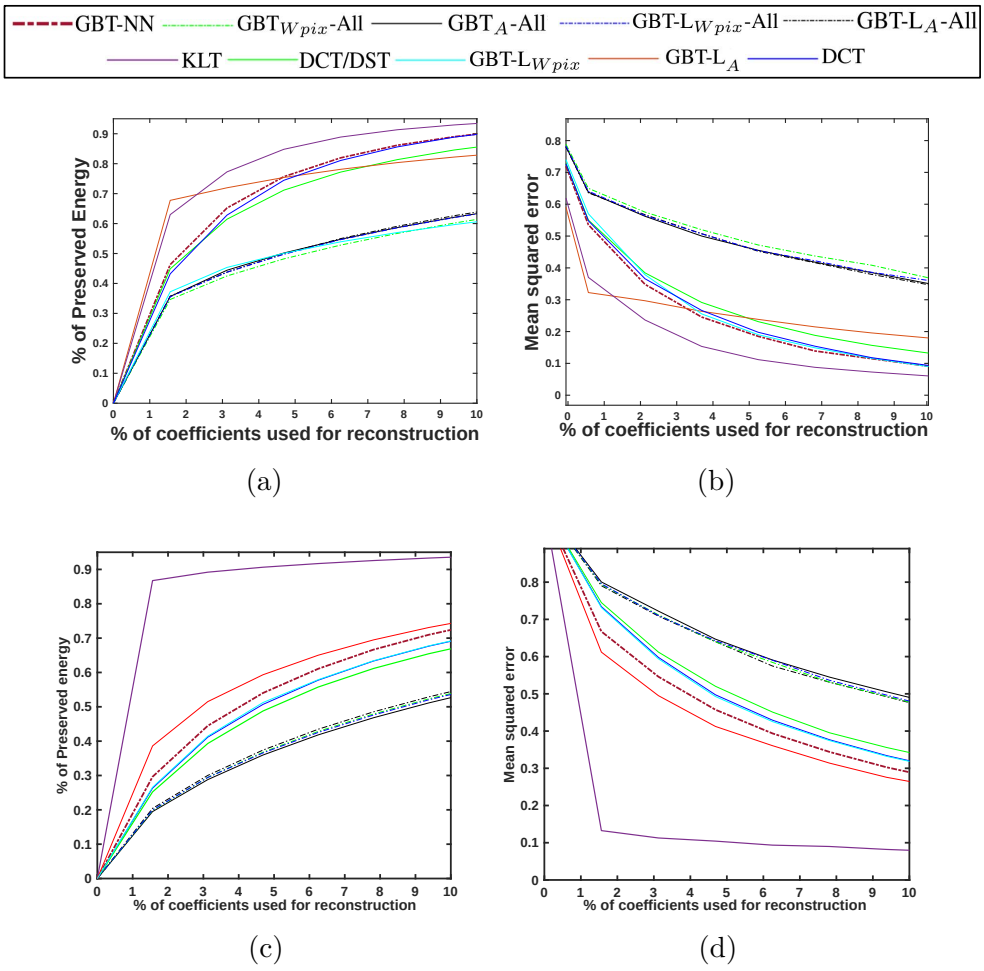


Figure 5.8: (a,c) PE (%) and (b,d) MSE vs. percentage of coefficients used for reconstruction of (1^{st} row) a frame of sequence *PeopleOnStreet* (Class A) and (2^{nd} row) a frame of sequence *BQTerrace* (Class B).

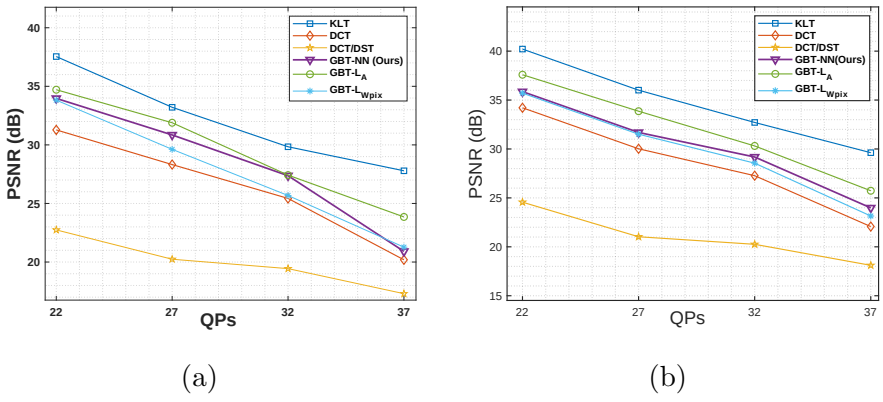
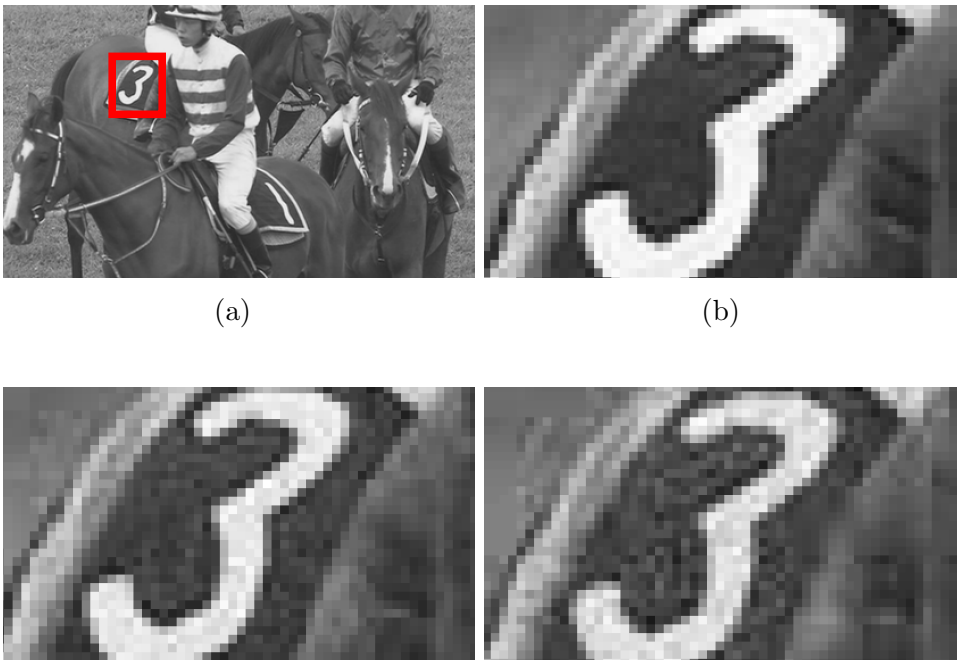


Figure 5.9: PSNR vs. QP for a frame of (a) sequence *ChinaSpeed* (Class SC) and (b) sequence *BlowingBubbles* (Class D).



(a) (b)

(c) (d)

Figure 5.10: (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.

(%) and MSE values vs. the percentage of coefficients used for reconstruction of a frame of sequence *BQTerrace* (Class B) and *PeopleOnStreet* (Class A). Table 5.4 tabulates the PSNR values for the evaluated frames/images when 4 different QPs are applied to the transform coefficients for the popular transforms, whereas; Table 5.5 tabulates the PSNR values for the evaluated frames/images for the proposed transforms. Note that the proposed GBT-NN outperforms both the DCT and DCT/DST. Table 5.6 and Table 5.7 provides a more comprehensive summary of the findings for simpler interpretation and easier analysis for PE-MSE and PSNR respectively. Fig. 5.9 plots the PSNR values for a frame of the *ChinaSpeed* (Class Screen Content) and *BlowingBubbles* (Class D) sequences. Fig. 5.10 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.

5.3.4 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.

5.4 Summary

In this chapter, 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 re- construction, 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 outper- forms DCT and DCT/DST, which are widely used by modern video codecs.

Since the architecture we have used here is a fully connected layer, the number of parameters are are learning here are quite huge in number. Our

objective is to improve the complexity by learning less number of parameters and to adopt a more advanced architecture that is more effective at feature extraction. In our next chapter we aim to use smarter architecture to predict the graph.

Table 5.3: PE (in %) and MSE using a small percentage of the largest coefficients.

Sequence	*Percentage of coefficients used for reconstruction*																			
	All-Connected														4-Connected					
	GBT_A-All				$GBT-L_A$-All				GBT_{Wpix}-All				**GBT-NN**				$GBT-L_{Wpix}$-All				$GBT-L_A$				$GBT-L_{Wpix}$			
	PE↑		MSE ↓		PE ↑		MSE ↓		PE↑		MSE ↓		PE ↑		MSE ↓		PE ↑		MSE ↓		PE ↑		MSE ↓		PE ↑		MSE ↓	
	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%
4:2:0 YUV sequences																												
Class A																												
Traffic	46.9	65.0	57.7	39.7	48.4	66.7	44.7	24.9	47.8	66.2	57.0	38.5	66.2	83.9	38.3	20.9	47.6	65.9	57.3	38.8	73.7	84.5	24.3	14.6	65.2	82.6	37.3	20.1
People_on_street	45.0	64.4	55.8	39.0	46.5	66.1	42.8	24.2	45.9	65.6	55.1	37.8	64.3	83.3	36.4	20.2	45.7	65.3	55.4	38.1	74.7	85.1	25.3	15.3	64.6	82.3	37	19.5
Nebuta_festival	44.0	61.3	54.8	36.0	45.5	63.0	41.8	21.2	44.9	62.5	54.1	34.8	63.3	80.2	35.4	17.2	44.7	62.2	54.4	35.1	75.8	86.7	26.5	16.8	63.5	80.6	35.6	18.4
Avg of Class A	45.3	63.6	56.1	38.2	46.8	65.3	43.1	23.4	46.2	64.8	55.4	37.0	64.6	82.5	36.7	19.4	46.0	64.5	55.7	37.3	74.8	85.4	25.4	15.5	64.4	81.8	36.6	19.3
Class B																												
Kimono	26.3	41.8	78.8	63.6	27.8	43.5	65.8	48.8	27.2	43.0	78.1	62.4	45.6	60.7	59.4	44.8	27.0	42.7	78.4	62.7	49.7	63.2	54.1	41.1	42.7	60.5	59.3	42.8
Cactus	25.7	40.6	78.2	62.4	27.2	42.3	65.2	47.6	26.6	41.8	77.5	61.2	45.0	59.5	58.8	43.6	26.4	41.5	77.8	61.5	49.1	62.0	53.5	39.9	40.7	58.6	61.7	44.9
Park_scene	25.7	41.2	77.5	63.1	27.2	42.9	64.5	48.3	26.6	42.4	76.8	61.9	45.0	60.1	58.1	44.3	26.4	42.1	77.1	62.2	49.2	62.6	53.5	40.6	40.7	58.2	60.2	43.5
BQTerrace	26.4	40.7	78.9	62.5	27.9	42.4	65.9	47.7	27.3	41.9	78.2	61.3	45.7	59.6	59.5	43.7	27.1	41.6	78.5	61.6	49.8	62.1	54.2	40.0	42.7	60.1	62.5	45.6
Avg of Class B	26.0	41.1	78.3	62.9	27.5	42.8	65.3	48.1	26.9	42.3	77.6	61.7	45.3	59.7	59.1	43.9	26.7	42.0	77.9	62.0	49.5	62.2	53.8	40.2	41.7	59.4	60.9	44.2
Class C																												
Race_horse	29.8	48.3	70.6	56.9	31.3	50.0	57.6	42.1	30.7	49.5	69.9	55.7	49.1	67.2	51.2	38.1	30.5	49.2	70.2	56.0	53.0	71.4	45.2	34.8	47.6	63.5	52.3	36.1
BQMall	29.2	47.1	70.0	55.6	30.7	48.8	57.0	40.8	30.1	48.3	69.3	54.4	48.5	66.0	50.6	36.8	29.9	48.0	69.6	54.7	52.4	70.2	44.6	33.6	45.6	61.6	54.7	38.3
Party_scene	29.2	47.8	70.1	56.3	30.7	49.5	57.1	41.5	30.1	49.0	69.4	55.1	48.5	66.7	50.7	37.5	29.9	48.7	69.7	55.4	52.4	70.9	44.7	34.2	45.6	61.2	53.2	36.9
Basketball_drill	29.9	47.2	70.7	55.8	31.4	48.9	57.7	41.0	30.8	48.4	70.0	54.6	49.2	66.1	51.3	37.0	30.6	48.1	70.3	54.9	53.1	70.3	45.3	33.7	47.6	63.2	55.5	39.0
Avg of Class C	29.5	47.6	70.3	56.2	31.0	49.3	57.3	41.4	30.4	48.8	69.6	55.0	48.8	66.3	51.0	37.1	30.2	48.5	69.9	55.3	52.7	70.5	45.0	33.8	46.6	62.4	53.9	37.6
Class D																												
Race_horse_D	35.8	51.5	64.3	51.4	37.3	53.2	51.3	36.6	36.7	52.7	63.6	50.2	55.1	70.4	44.9	32.6	36.5	52.4	63.9	50.5	61.2	72.4	37.4	30.1	48.6	65.7	51.0	34.3
Blowing_bubble	35.2	50.3	63.7	50.5	36.7	52.0	50.7	35.7	36.1	51.5	63.0	49.3	54.5	69.2	44.3	31.7	35.9	51.2	63.3	49.6	60.6	71.2	36.8	28.9	51.0	67.8	49.0	32.4
BQ_square	35.2	51.0	63.7	51.2	36.7	52.7	50.7	36.4	36.1	52.2	63.0	50.0	54.5	69.9	44.3	32.4	35.9	51.9	63.3	50.3	60.6	71.9	36.8	29.6	49.5	66.5	49.0	32.0
Basketball_pass	35.9	50.4	64.4	50.6	37.4	52.1	51.4	35.8	36.8	51.6	63.7	49.4	55.2	69.3	45.0	31.8	36.6	51.3	64.0	49.7	61.3	71.5	37.5	29.0	51.8	68.6	51.0	33.9
Avg of Class D	35.5	50.8	64.0	50.9	37.0	52.5	51.0	36.1	36.4	52.0	63.3	49.7	54.8	69.5	44.6	32.0	36.2	51.7	63.6	50.0	60.9	71.5	37.1	29.2	50.2	67.1	50.0	33.2
Class E																												
Kristine_and_Sara	46.6	60.9	58.9	44.2	48.1	62.6	45.9	29.4	47.5	62.1	58.2	43.0	65.9	79.8	39.5	25.4	47.3	61.8	58.5	43.3	70.1	81.0	30.6	20.6	59.7	75.5	41.9	25.5
Four_people	44.6	60.2	56.9	43.5	46.1	61.9	43.9	28.7	45.5	61.4	56.2	42.3	63.9	79.1	37.5	24.7	45.3	61.1	56.5	42.6	69.7	80.3	30.2	20.0	59.5	75.4	43.5	26.9
Jhonny	43.7	57.2	56.0	40.5	45.2	58.9	43.0	25.7	44.6	58.4	55.3	39.3	63.0	76.1	36.6	21.7	44.4	58.1	55.6	39.6	69.9	81.8	30.4	21.5	58.6	74.9	42.1	25.1
Avg of Class E	45.0	59.4	57.2	42.7	46.5	61.1	44.2	27.9	45.9	60.6	56.5	41.5	64.3	78.3	37.9	23.9	45.7	60.3	56.8	41.8	69.9	81.0	30.4	20.7	59.3	75.3	42.5	25.8
Class F/SC																												
China_speed	31.9	46.9	69.0	54.7	33.4	48.6	56.0	39.9	32.8	48.1	68.3	53.5	51.2	65.8	49.6	35.9	32.6	47.8	68.6	53.8	55.3	68.9	43.1	31.7	45.4	61.5	55.2	39.6
Slide_show	31.3	45.7	68.4	53.5	32.8	47.4	55.4	38.7	32.2	46.9	67.7	52.3	50.6	64.6	49.0	34.7	32.0	46.6	68.0	52.6	54.7	67.6	42.5	30.4	47.7	63.6	53.2	37.7
Sc_Map	31.3	46.3	68.4	54.0	32.8	48.0	55.4	39.2	32.2	47.5	67.7	52.8	50.6	65.2	49.0	35.2	32.0	47.2	68.0	53.1	54.7	68.3	42.5	31.1	46.2	62.3	53.2	37.3
Sc_Programming	32.0	45.8	69.1	53.6	33.5	47.5	56.1	38.8	32.9	47.0	68.4	52.4	51.3	64.7	49.7	34.8	32.7	46.7	68.7	52.7	55.4	67.7	43.2	30.5	48.6	64.4	55.2	39.2
Avg of Class F/SC	31.6	46.2	68.7	54.0	33.1	47.9	55.7	39.2	32.5	47.4	68.0	52.8	50.9	64.8	49.3	34.9	32.3	47.1	68.3	53.1	55.0	67.9	42.8	30.7	47.0	63.0	54.2	38.5
Pathology Image																												
NCI01	46.7	62.4	50.5	38.3	48.2	64.1	37.5	23.5	47.6	63.6	49.8	37.1	66.0	81.3	31.1	20.0	47.4	63.3	50.1	37.4	71.1	83.3	21.7	17.4	59.7	77.8	38.9	24.3
NCI20	46.5	61.1	50.3	38.1	48.0	62.8	37.3	23.3	47.4	62.3	49.6	36.9	65.8	80.0	30.9	19.8	47.2	62.0	49.9	37.2	70.9	82.1	21.5	17.3	60.0	80.0	36.9	22.3
GBM2	47.1	61.8	50.9	38.8	48.6	63.5	37.9	24.0	48.0	63.0	50.2	37.6	66.4	80.7	31.5	20.5	47.8	62.7	50.5	37.9	71.5	82.7	22.1	18.0	60.5	78.6	36.9	22.0
COAD1	46.9	61.3	50.7	38.3	48.4	63.0	37.7	23.5	47.8	62.5	50.0	37.1	66.2	80.2	31.3	20.0	47.6	62.2	50.3	37.4	71.4	82.2	22.0	17.4	62.9	80.7	38.9	23.9
Avg of pathology	46.8	61.7	50.6	38.4	48.3	63.4	37.6	23.6	47.7	62.9	49.9	37.2	66.1	80.3	31.2	20.1	47.5	62.6	50.2	37.5	71.2	82.3	21.8	17.6	61.3	79.3	37.9	23.1
Overall average	37.1	52.7	63.6	49.0	38.6	54.4	50.6	34.2	38.0	53.9	62.9	47.8	56.4	71.6	44.2	30.2	37.8	53.6	63.2	48.1	62.0	74.4	36.6	26.8	52.9	69.7	48.0	31.7

Table 5.4: PSNR (dB) values when using quantization on the transform coefficients for popular transforms.

		Quantization parameters															
Sequence	Resolution	Baseline				SOTA				Popular HEVC transforms							
		KLT				GL-GBT				DCT				DCT/DST			
	QP	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37
Class A																	
Traffic	2560×1600	37.5	33.2	29.8	27.8	34.0	33.0	30.6	22.1	33.3	31.3	27.5	21.2	21.0	20.1	19.5	16.1
People_on_street		38.7	34.4	31.0	29.0	35.2	34.2	31.8	23.3	34.5	32.5	28.7	22.4	22.0	19.3	18.6	17.3
Nebuta_festival		39.5	35.2	31.8	29.8	36.0	35.0	32.6	24.1	35.3	33.3	29.5	23.2	21.7	20.2	19.5	17.0
Class B																	
Kimono	1920×1080	37.4	35.4	30.9	29.4	34.5	32.6	28.4	24.5	32.8	30.9	28.1	21.5	19.2	16.9	16.2	15.4
Cactus		38.1	36.1	31.7	30.0	35.1	33.3	29.2	25.3	32.9	31.1	28.3	22.1	20.3	17.9	16.9	16.5
Park_scene		36.0	34.0	29.5	28.0	33.1	31.2	27.0	23.1	31.4	29.9	27.1	20.1	20.1	19.7	18.6	15.5
BQTerrace		38.8	36.7	32.3	30.7	35.8	33.4	29.8	25.9	32.9	31.1	28.4	22.8	21.6	20.9	20.4	17.7
Class C																	
Race_horse	832×480	38.8	36.0	30.7	23.8	34.8	32.9	28.7	24.8	33.1	30.0	28.1	21.8	20.5	16.8	16.2	15.6
BQMall		39.2	36.4	31.1	24.3	35.2	33.3	29.2	25.3	33.5	30.0	28.1	22.2	19.9	19.0	17.7	15.9
Party_scene		37.1	34.3	29.0	22.1	33.1	31.2	27.0	23.1	31.4	29.9	27.1	20.1	17.9	15.8	15.1	14.0
Basketball_drill		39.7	36.9	31.7	24.8	35.7	33.8	29.7	25.9	33.5	30.1	28.2	22.8	20.9	19.9	18.0	17.7
Class D																	
Race_horse_D	416×240	40.5	38.4	34.1	27.0	39.7	37.9	32.6	29.7	34.5	31.6	28.0	24.4	22.2	21.4	20.1	19.3
Blowing_bubble		41.9	40.0	35.4	28.4	41.1	39.2	34.0	31.1	35.9	23.6	29.0	25.7	23.6	22.4	21.1	20.6
BQ_square		39.1	37.0	32.7	25.6	38.3	36.5	31.2	28.3	33.1	31.2	27.0	23.0	20.8	20.0	19.0	17.9
Basketball_pass		42.2	40.3	35.7	28.7	41.4	39.5	34.3	31.3	36.0	32.9	29.3	26.0	23.7	22.7	21.3	20.9
Class E																	
Kristine_and_Sara	1280×720	40.6	39.3	35.5	30.7	36.7	33.7	31.3	26.7	36.3	34.0	31.2	26.3	24.0	22.3	21.2	21.1
Four_people		39.0	37.7	33.9	29.1	35.1	32.1	29.7	25.1	34.7	33.0	29.6	24.7	22.4	21.8	20.9	19.6
Jhonny		41.8	40.5	36.7	31.9	38.0	34.9	32.5	28.0	37.5	35.0	32.2	27.5	24.0	21.3	20.2	17.7
Class F/SC																	
China_speed	1024×768	40.2	37.0	33.9	33.1	36.9	34.0	30.5	25.1	34.9	32.8	29.6	26.0	20.4	18.7	18.1	17.9
Slide_show	1280×720	42.1	35.1	32.0	31.2	35.0	32.1	28.6	23.2	33.9	32.3	29.1	24.1	20.4	18.4	18.1	18.0
Sc_Map		40.9	37.6	34.6	33.8	37.5	34.7	31.2	25.8	35.5	32.8	30.3	26.7	23.0	19.4	18.8	18.6
Sc_Programming		41.2	37.9	34.9	34.1	37.9	35.0	31.6	26.1	35.6	32.8	30.3	27.0	23.3	22.7	22.3	21.9
Overall average		39.6	36.8	32.7	28.8	36.4	34.3	30.5	25.8	34.2	31.9	28.8	23.7	20.6	19.0	18.2	17.0

Table 5.5: PSNR (dB) values when using quantization on the transform coefficients.

Quantization parameters																											
Sequence	Resolution	All-connected																				4-connected					
		GBT_A-All				$GBT-L_A$-All				GBT_{Wpix}-All				**GBT-NN**				$GBT-L_{Wpix}$-All				$GBT-L_A$				$GBT-L_{Wpix}$			
	QP	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37
Class A																													
Traffic	2560×1600	7.6	6.0	6.9	8.3	15.4	13.8	15.0	14.0	9.1	8.1	9.4	9.9	33.2	30.2	29.2	25.1	10.3	9.8	9.5	10.2	33.6	32.7	30.4	22.0	33.4	30.5	28.7	19.5
People_on_street		8.9	7.3	8.3	9.6	16.7	15.1	16.4	15.3	10.4	9.4	10.8	11.2	34.5	31.5	30.6	26.4	11.6	11.1	10.9	11.5	34.9	33.9	31.7	23.3	34.7	31.7	30.0	20.8
Nebuta_festival		9.7	8.1	9.1	10.4	17.5	15.9	17.2	16.1	11.2	10.2	11.6	12.0	35.3	32.3	31.4	27.2	12.4	11.9	11.7	12.3	35.6	34.7	32.4	24.0	35.4	32.5	30.7	21.5
Class B																													
Kimono	1920×1080	4.8	4.1	4.0	5.3	12.6	11.9	12.1	11.0	6.3	6.2	6.5	6.9	30.4	28.3	26.3	22.1	7.5	7.9	6.6	7.2	33.8	31.9	28.1	24.4	33.6	29.7	26.4	21.9
Cactus		5.6	4.7	4.7	6.0	13.4	12.5	12.8	11.7	7.1	6.8	7.2	7.6	31.2	28.9	27.0	22.8	8.3	8.5	7.3	7.9	34.5	32.6	28.9	25.2	34.3	30.4	27.2	22.7
Park_scene		3.4	2.7	2.6	3.9	11.2	10.5	10.7	9.6	4.9	4.8	5.1	5.5	29.0	26.9	24.9	20.7	6.1	6.5	5.2	5.8	32.5	30.5	26.8	23.0	32.3	28.3	25.1	20.5
BQTerrace		9.4	7.4	8.1	6.6	17.2	15.2	16.2	12.3	10.9	9.5	10.6	8.2	35.0	31.6	30.4	23.4	12.1	11.2	10.7	8.5	35.1	33.3	29.6	25.9	34.9	31.1	27.9	23.4
Class C																													
Race_horse	832×480	5.0	4.3	4.2	5.5	12.8	12.1	12.3	11.2	6.5	6.4	6.7	7.1	30.6	28.5	26.5	22.3	7.7	8.1	6.8	7.4	34.5	32.6	28.4	24.6	34.3	30.4	26.7	22.1
BQMall		6.1	5.0	4.7	6.0	13.9	12.8	12.8	11.7	7.6	7.1	7.2	7.6	31.7	29.2	27.0	22.8	8.8	8.8	7.3	7.9	34.9	33.0	28.9	25.2	34.7	30.8	27.2	22.7
Party_scene		3.4	2.7	2.6	3.9	11.2	10.5	10.7	9.6	4.9	4.8	5.1	5.5	29.0	26.9	24.9	20.7	6.1	6.5	5.2	5.8	32.9	30.9	26.7	23.0	32.7	28.7	25.0	20.5
Basketball_drill		8.9	7.4	6.4	6.6	16.7	15.2	14.5	12.3	10.4	9.5	8.9	8.2	34.5	31.6	28.7	23.4	11.6	11.2	9.0	8.5	35.4	33.5	29.4	25.7	35.2	31.3	27.7	23.2
Class D																													
Race_horse_D	416×240	8.9	8.8	7.5	6.5	16.7	16.6	15.6	12.2	10.4	10.9	10.0	8.1	34.5	33.0	29.8	23.3	11.6	12.6	10.1	8.4	39.5	37.7	32.4	29.6	39.3	35.5	30.7	27.1
Blowing_bubble		12.3	11.2	9.9	7.8	20.1	19.0	18.0	13.5	13.8	13.3	12.4	9.4	37.9	35.4	32.2	24.6	15.0	15.0	12.5	9.7	40.9	39.0	33.8	31.0	35.8	31.3	28.5	23.0
BQ_square		7.5	7.4	6.1	5.1	15.3	15.2	14.2	10.8	9.0	9.5	8.6	6.7	33.1	31.6	28.4	21.9	10.2	11.2	8.7	7.0	38.1	36.3	31.0	28.2	37.9	34.1	29.3	25.7
Basketball_pass		12.3	12.5	10.2	8.1	20.1	20.3	18.3	13.8	13.8	14.6	12.7	9.7	37.9	36.7	32.5	24.9	15.0	16.3	12.8	10.0	41.1	39.3	34.1	31.2	40.9	37.1	32.4	28.7
Class E																													
Kristine_and_Sara	1280×720	10.7	10.3	9.4	8.1	18.5	18.1	17.5	13.8	12.2	12.4	11.9	9.7	36.3	34.5	31.7	24.9	13.4	14.1	12.0	10.0	36.0	33.0	31.1	26.5	35.8	30.8	29.4	24.0
Four_people		9.1	8.7	7.9	6.5	16.9	16.5	16.0	12.2	10.6	10.8	10.4	8.1	34.7	32.9	30.2	23.3	11.8	12.5	10.5	8.4	34.5	31.4	29.5	25.0	34.3	29.2	27.8	22.5
Jhonny		12.0	11.5	10.7	9.4	19.8	19.3	18.8	15.1	13.5	13.6	13.2	11.0	37.6	35.7	33.0	26.2	14.7	15.3	13.3	11.3	37.3	34.3	32.3	27.9	37.1	32.1	30.6	25.4
Class F/SC																													
China_speed	1024×768	10.4	9.4	4.9	7.7	18.2	17.2	13.0	13.4	11.9	11.5	7.4	9.3	36.0	33.6	27.2	24.5	13.1	13.2	7.5	9.6	36.3	33.3	30.3	25.0	33.8	29.7	26.8	21.2
Slide_show	1280×720	8.6	7.5	3.0	5.8	16.4	15.3	11.1	11.5	10.1	9.6	5.5	7.4	34.2	31.7	25.3	22.6	11.3	11.3	5.6	7.7	34.4	31.4	28.4	23.1	36.5	31.7	28.0	22.4
Sc_Map		11.1	10.0	5.6	8.3	18.9	17.8	13.7	14.0	12.6	12.1	8.1	9.9	36.7	34.2	27.9	25.1	13.8	13.8	8.2	10.2	36.9	34.0	31.0	25.7	39.2	33.8	30.8	25.2
Sc_Programming		14.5	12.3	8.9	8.6	22.3	20.1	17.0	14.3	16.0	14.4	11.4	10.2	40.1	36.5	31.2	25.4	17.2	16.1	11.5	10.5	37.2	34.3	31.4	26.0	39.5	34.1	31.2	25.5
Overall average		8.6	7.7	6.6	7.0	16.4	15.5	14.7	12.7	10.1	9.8	9.1	8.6	34.2	31.9	28.9	23.8	11.3	11.5	9.2	8.9	35.9	33.8	30.3	25.7	35.7	31.6	28.6	23.2

Table 5.6: 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	89.73	12.22	93.43	10.43
DCT	17.49	82.27	52.41	48.48	69.58	31.88
DCT/DST	16.94	82.74	51.89	49.81	68.14	33.49
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
$GBT_{W_{pix}}$-All	12.71	86.78	38.00	62.85	53.86	47.79
GBT-NN (ours)	**18.97**	**78.72**	**56.43**	**44.24**	**72.40**	**28.94**
GBT-$L_{W_{pix}}$-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-$L_{W_{pix}}$	17.01	82.82	52.58	47.93	69.18	31.86

Table 5.7: 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

Chapter 6

Graph Based Transform based on 3D Convolutional Neural Network for Intra-Prediction of Imaging Data

6.1 Introduction

This chapter presents a novel class of Graph-based Transform based on 3D convolutional neural networks (GBT-CNN) within the context of block-based predictive transform coding of imaging data. The proposed GBT-CNN uses a 3D convolutional neural network (3D-CNN) to predict the graph information needed to compute the transform and its inverse, thus reducing the signalling cost to reconstruct the data after transformation. The GBT-CNN outperforms the DCT and DCT/DST, which are commonly employed in current video codecs, in terms of the percentage of energy preserved by a subset of transform coefficients, the mean squared error of the reconstructed data, and the transform coding gain according to evaluations on several video frames and medical images.

When the GBT is used in block-based PTC, the graph used to compute the GBT of each block at the encoder should be available to compute the inverse GBT during reconstruction at the decoder. This additional data should then be signalled into the bitstream, increasing the overhead. Our previous works in [8, 9] in Chapter 5 show an attractive solution for learning a mapping function to design a GBT without requiring to signal additional information. To make the same graph available to the decoder for reconstruction, a template-

based prediction strategy is used to predict the residual followed by a neural network (NN) that estimates the graph to perform the GBT. Specifically, this approach involves two prediction methods: predicting the residuals and using the predicted residuals as an input to the NN to predict the graph. This approach, unfortunately, tends to degrade the quality of the reconstructed residual at the decoder. To address this issue, this chapter introduces a novel class of GBT based on a 3D CNN (GBT-CNN), which uses the 3 reconstructed blocks surrounding the block to be encoded as input. We use a 3D CNN because the 3D convolution allows exploiting the relationship between these surrounding blocks, which are expected to be similar to the one to be encoded, by treating them as a single volume. These features are used to predict the adjacency matrix of the block to be encoded. More specifically, our proposed method maps these 3 surrounding blocks in the pixel domain to the graph representing the residual block to be encoded by using an encoding-decoding architecture. These 3 surrounding blocks are used to compute the same graph at the decoder, thus allowing to perform the inverse GBT. Our approach then avoids signalling extra information into the bitstream. To the best of our knowledge, no approach for learning a graph using 3D CNNs within the context of block-based PTC and GBTs has been proposed before. In terms of PE, MSE, PSNR, and the transform coding gain, our evaluations on several video frames and medical images show that the proposed GBT-CNN outperforms the DCT/DST, DCT, and other similar GBTs [6–9].

6.2 Proposed GBT-CNN

Let us denote a (square) residual block as $\mathbf{S} \in \mathbb{R}^{\sqrt{N}\times\sqrt{N}}$, with a total of N residual values. $\mathbf{S}$ can be represented as an undirected weighted graph, $G = (V, E, \mathbf{A})$, where $V = \{v_n\}_{n=1}^{N}$ is the set of N nodes, E is the set of edges, and $\mathbf{A} \in \mathbb{R}^{N\times N}$ is the normalized symmetric adjacency matrix. The matrix $\mathbf{A}$ of a weighted graph stores the edge weights. The GBT for $\mathbf{S}$ can be computed by the eigendecomposition of the graph Laplacian, $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D}$ is the diagonal degree matrix. The eigendecomposition 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 [133].

As the graph Laplacian requires the computation of the matrix $\mathbf{A}$, our objective is to develop a mapping between the 3 reconstructed blocks surrounding the block to be encoded and the matrix $\mathbf{A}$ of the residual block to be encoded. To this end, we aim to learn a mapping function of the form:

$$\mathbf{A_B} \approx f(\mathbf{B}_{[\mathbf{I},\mathbf{J},\mathbf{K}]}), \tag{6.1}$$

where $\mathbf{B}_{[\mathbf{I},\mathbf{J},\mathbf{K}]}$ represents a matrix with the 3 reconstructed gray scale blocks

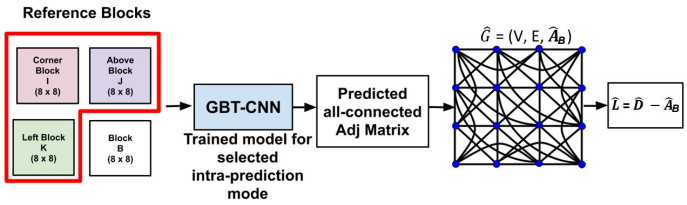


Figure 6.1: GBT-CNN used to predict matrix $\mathbf{A}$ for the current residual block. In this work, we use an all-connected topology for the graphs.

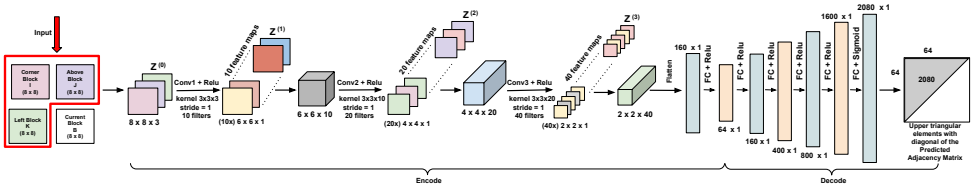


Figure 6.2: Architecture of the proposed GBT-CNN for 8×8 blocks.

surrounding the block $\mathbf{B}$ to be encoded and $\mathbf{A_B}$ is the adjacency matrix of the graph of its residual block (see Fig. 6.1). Our solution to learn the mapping function in Eq. 7.2 is based on an encoding-decoding 3D CNN, as depicted in Fig. 6.2 for the case of 8×8 blocks. In this architecture, the convolution takes place over 3 layers of the encoder to extract feature maps $\mathbf{Z}^{(l_e)}$ where $\mathbf{Z}^{(0)} = \mathbf{B}_{[\mathbf{I},\mathbf{J},\mathbf{K}]}$ is the input and $l_e \in [1, 3]$ denotes the layer number. After the convolutional layers, the feature maps are vectorized as an input to the decoder part of the architecture. Specifically, $\mathbf{Z}^{(l_e=3)}$ is transformed back to a reconstructed vector $\hat{\mathbf{a}}_{u,\mathbf{B}}$ by the decoder over a number of fully-connected (FC) layers:

$$\hat{\mathbf{a}}_{u,\mathbf{B}} = h(\mathbf{W}^{(l_d)}\mathbf{Z}^{(l_d-1)}), \tag{6.2}$$

where $\hat{\mathbf{a}}_{u,\mathbf{B}}$ is the prediction of the vectorized upper triangular matrix of $\mathbf{A_B}$, $h(\cdot)$ denotes an activation function, $\mathbf{W}^{(l_d)}$ is a weight matrix for the decoder layer l_d, and $\mathbf{Z}^{(l_d-1)}$ is the hidden representation produced by the decoder layer $(l_d - 1)$. For each FC layer, we apply the *ReLu* activation function, while the *Sigmoid* activation function is applied to the last layer of the encoder. The decoder consists of 6 FC layers. Note that the network only predicts the upper triangular elements and the diagonal of matrix $\mathbf{A_B}$. To obtain a complete predicted matrix $\hat{\mathbf{A}}_\mathbf{B}$, we mirror the elements of the upper diagonal to the lower diagonal:

$$\hat{\mathbf{A}}_\mathbf{B} = \hat{\mathbf{A}}_{u,\mathbf{B}} + (\hat{\mathbf{A}}_{u,\mathbf{B}})^T - Diag(\hat{\mathbf{A}}_{u,\mathbf{B}}), \tag{6.3}$$

where $\hat{\mathbf{A}}_{u,\mathbf{B}}$ is the matrix form of $\hat{\mathbf{a}}_{u,\mathbf{B}}$, $Diag(\hat{\mathbf{A}}_{u,\mathbf{B}})$ is the diagonal elements of $\hat{\mathbf{A}}_{u,\mathbf{B}}$ and $(\hat{\mathbf{A}}_{u,\mathbf{B}})^T - Diag(\hat{\mathbf{A}}_{u,\mathbf{B}})$ denotes the lower triangular matrix. We optimize the GBT-CNN by minimizing the following loss function:

$$L = \| \hat{\mathbf{a}}_\mathbf{B} - \mathbf{a}_\mathbf{B} \|_2^2 + \lambda \| \mathbf{W}(:) \|_2, \tag{6.4}$$

where $\hat{\mathbf{a}}_\mathbf{B}$ is the complete predicted matrix $\hat{\mathbf{A}}_{u,\mathbf{B}}$ (see Eq. 6.3) in vectorized

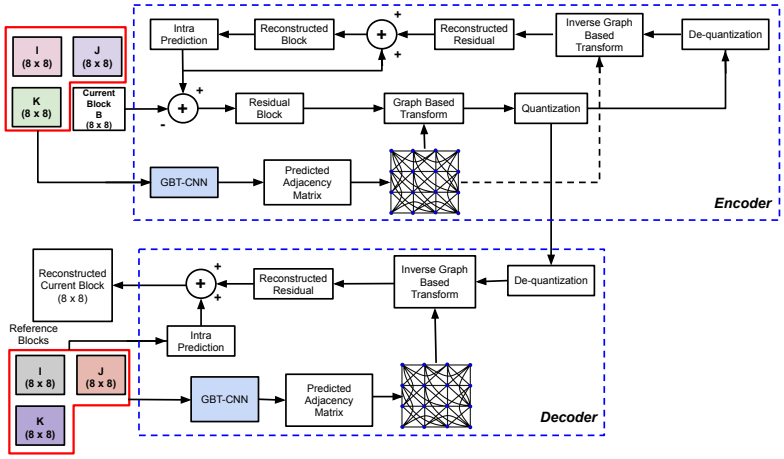


Figure 6.3: Block diagram of the proposed framework for block-based PTC.

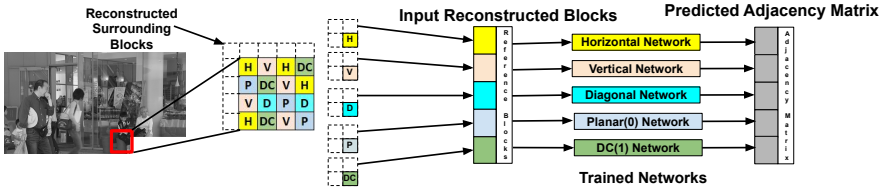


Figure 6.4: The intra-prediction mode used for each block determines the trained network to use. The figure depicts a section of a frame that has been predicted using several prediction modes.

form, $\mathbf{a_B}$ is the vectorized form of the ground truth matrix $\mathbf{A_B}$, $\| \, . \, \|$ is the L_2 norm, $\mathbf{W}(:)$ represents the learnable parameters in vector form, and λ controls the amount of L_2 regularization on the learnable parameters. The graph used to compute the GBT for the current residual block is then $\hat{G} = (V, E, \hat{\mathbf{A}}_\mathbf{B})$. To reconstruct the current block, the same graph used to compute the GBT should be used to compute the inverse GBT at the decoder. To this end, the same reconstructed blocks used as input are available at the decoder to predict matrix $\mathbf{A_B}$ by the trained GBT-CNN. Fig. 6.3 illustrates the complete compression framework assuming the trained GBT-CNN is common knowledge between encoder and decoder. As a result, our solution does not require signalling any additional data in the compressed bit-stream.

Table 6.1: GBTs used in the evaluation.

Approach	Explanation
	All-C Topology
GBT-NN	Train a NN to predict matrix $\mathbf{A_B}$. The graph for GBT has UE weights but no self-loops.
GBT-CNN	**(ours) Train a 3D CNN to predict matrix $\mathbf{A_B}$. The graph for GBT has UE weights but no self-loops.**
GL-GBT	Uses covariance matrices from several training examples to estimate the graph Laplacian.
	4-Connected Topology
GBT-L$_A$	Use actual residual to compute a graph with UE weights and normalized self-loop weights.

Table 6.2: Performance evaluation of the model on test data for all the networks.

	Networks				
Metric	H	V	D	DC	P
MSE	0.0076	0.0162	0.0134	0.0185	0.0384

6.3 Performance evaluation

6.3.1 Experimental setup

Based on the 35 HEVC intra-prediction modes, we train 5 distinct networks: one for horizontal (H) modes (modes 3 - 17), one for vertical (V) modes (modes 9 - 13), one for diagonal (D) modes (modes 2, 18 and 34), one for the DC mode, and one for the planar (P) mode (see Fig. 6.1 and Fig. 6.4). We use 8×8 blocks and graphs with unit edge (UE) weights and an all-connected (All-C) topology with no self-loops. We use 40 distinct grey level YUV frames from Class A, B, C, D, E, and Screen Content, which are popular video sequences for testing video codecs [19]. We also employ the green (G) component of 10 colour pathology images from the US National Cancer Institute's Center for Biomedical Informatics and Information Technology [128, 129]. We use $64,320$ samples in total for the five networks. Each sample comprises the following values: $\{\mathbf{B}_{[\mathbf{I},\mathbf{J},\mathbf{K}]}, \mathbf{A}_{\mathbf{B}}\}$, where $\mathbf{A}_{\mathbf{B}}$ is the ground truth. 80% of the data is used for training and 20% is used for testing. The training and testing sets do not overlap. We use the Adam optimizer to train each network for 150 epochs with a learning rate of 0.0001 and $\lambda = 0.001$ (see Eq. 6.4). As mentioned before, the surrounding reconstructed blocks $\mathbf{I}$, $\mathbf{J}$, and $\mathbf{K}$, which are available at the decoder, are used as inputs to a specific trained GBT-CNN according to the mode used by the encoder (see Fig. 6.4). We use 10, 20, and 40 3D filters respectively in each 3D-CNN layer. For each convolution operation, we apply 1 stride, which leads to feature maps with dimensions of $6 \times 6 \times 10$, $4 \times 4 \times 20$, and $2 \times 2 \times 40$, respectively. At the end of the convolutions, the feature maps are flattened to a vector of dimensions of 160×1. The output layer has 2080 neurons, which matches the upper triangular elements plus those in the diagonal of the matrix $\mathbf{A}_{\mathbf{B}}$.

6.3.2 Evaluation of energy compaction for unquantized coefficients

We compare our proposed method with several GBTs as summarized in Table 6.1. Specifically, the table tabulates the topology of the graph, the edge weights, and how the graph is obtained to compute the GBT. The KLT, DCT, and DCT/DST as used in the HEVC and VVC standards are also evaluated, with the DCT/DST being employed as separable transforms for rows and columns

Table 6.3: PE (in %) and MSE using a small percentage of the largest coefficients.

Sequence	*Percentage of coefficients used for reconstruction*																			
	Baseline				Offline training								No offline training							
	KLT				**GBT-CNN**				GBT-NN				GL-GBT				GBT-L$_A$			
	PE↑		MSE ↓		PE ↑		MSE ↓		PE↑		MSE ↓		PE ↑		MSE ↓		PE ↑		MSE ↓	
	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%
4:2:0 YUV sequences																				
Class A																				
Traffic	89.1	93.1	13.9	12.1	67.1	85.0	37.8	19.9	66.2	83.9	38.3	20.9	90.1	93.9	14.6	11.8	73.7	84.5	24.3	14.6
People_on_street	88.8	92.4	12.1	11.4	65.3	84.0	36.9	18.5	64.3	83.3	36.4	20.2	89.1	93.1	12.8	10.4	74.7	85.1	25.3	15.3
Nebuta_festival	87.2	90.9	13.6	10.8	63.8	82.8	33.6	16.5	63.3	80.2	35.4	17.2	87.4	90.2	11.4	10.5	75.8	86.7	26.5	16.8
Avg of Class A	88.4	92.2	13.2	11.5	65.4	83.9	36.2	18.3	64.6	82.5	36.7	19.4	88.9	92.4	13.0	10.9	74.8	85.4	25.4	15.5
Class B																				
Kimono	93.2	98.3	14.8	12.4	49.1	61.0	55.9	41.9	45.6	60.7	59.4	44.8	96.5	99.9	08.6	06.9	49.7	63.2	54.1	41.1
Cactus	92.6	97.4	13.1	12.2	48.5	60.2	55.1	41.2	45.0	59.5	58.8	43.6	96.	99.9	08.0	06.5	49.1	62.0	53.5	39.9
Park_scene	92.1	94.8	13.5	10.9	48.2	60.0	54.9	40.9	45.0	60.1	58.1	44.3	95.8	99.9	07.9	06.4	49.2	62.6	53.5	40.6
BQTerrace	93.0	95.4	11.6	10.5	49.0	60.1	55.0	41.0	45.7	59.6	59.5	43.7	96.0	99.9	08.0	06.7	49.8	62.1	54.2	40.0
Avg of Class B	92.7	96.5	13.3	11.5	48.7	60.3	55.2	41.3	45.3	59.7	59.1	43.9	96.1	99.9	8.1	06.7	49.5	62.2	53.8	40.2
Class C																				
Race_horse	92.0	95.9	09.5	07.9	52.9	70.3	46.8	35.6	49.1	67.2	51.2	38.1	93.9	94.5	06.1	06.0	53.0	71.4	45.2	34.8
BQMall	89.8	93.3	09.7	07.1	52.2	70.0	46.3	35.2	48.5	66.0	50.6	36.8	93.5	93.9	06.1	06.1	52.4	70.2	44.6	33.6
Party_scene	90.3	94.2	08.9	06.9	52.0	69.8	46.1	35.1	48.5	66.7	50.7	37.5	93.4	93.6	06.1	06.1	52.4	70.9	44.7	34.2
Basketball_drill	94.5	98.1	08.7	07.7	53.0	70.1	46.7	35.7	49.2	66.1	51.3	37.0	93.8	94.4	06.1	05.9	53.1	70.3	45.3	33.7
Avg of Class C	91.6	95.4	09.1	07.4	52.5	70.1	46.5	35.4	48.8	66.3	51.0	37.1	93.7	94.1	06.1	06.0	52.7	70.5	45.0	33.8
Class D																				
Race_horse_D	92.0	95.0	11.6	09.6	57.9	69.9	38.2	29.6	55.1	70.4	44.9	32.9	91.8	95.8	06.0	06.0	61.2	72.4	37.4	30.1
Blowing_bubble	90.7	94.5	10.0	08.4	58.4	70.3	38.8	30.0	54.5	69.2	44.3	31.7	92.1	97.1	06.2	06.2	60.6	71.2	36.8	28.9
BQ_square	90.2	94.2	09.8	08.1	58.0	70.1	38.5	29.9	54.5	69.9	44.3	32.4	92.0	95.9	06.1	06.1	60.6	71.9	36.8	29.6
Basketball_pass	89.8	94.0	09.3	07.6	57.8	70.0	38.3	29.7	55.2	69.3	45.0	31.8	92.0	94.6	06.0	06.0	61.3	71.5	37.5	29.0
Avg of Class D	90.7	94.4	10.2	08.4	58.0	70.1	38.5	29.8	54.8	69.5	44.6	32.0	92.0	95.9	06.1	06.1	60.9	71.5	37.1	29.2
Class E																				
Kristine_and_Sara	87.8	91.7	14.8	13.1	70.0	80.8	33.7	22.9	65.9	79.8	39.5	25.4	93.9	96.8	07.8	07.0	70.1	81.0	30.6	20.6
Four_people	87.6	91.5	14.4	13.0	69.3	79.9	33.1	21.9	63.9	79.1	37.5	24.7	93.1	95.9	07.3	06.5	69.7	80.3	30.2	20.0
Jhonny	88.5	91.9	15.8	13.6	69.3	80.1	33.6	21.8	63.0	76.1	36.6	21.7	93.4	96.1	07.9	06.7	69.9	81.8	30.4	21.5
Avg of Class E	87.9	91.7	15.0	13.2	69.5	80.3	33.5	22.2	64.3	78.3	37.9	23.9	93.4	96.3	07.7	06.7	69.9	81.0	30.4	20.7
Class F/SC																				
China_speed	89.0	92.4	14.0	12.1	55.1	66.1	44.2	31.9	51.2	65.8	49.6	35.9	92.7	96.9	06.0	04.8	55.3	68.9	43.1	31.7
Slide_show	88.3	91.9	13.5	11.4	54.3	65.0	43.0	30.9	50.6	64.6	49.0	34.7	92.2	96.2	05.8	04.3	54.7	67.6	42.5	30.4
Sc_Map	87.9	90.9	12.3	10.9	54.6	65.2	43.4	31.2	50.6	65.2	49.0	35.2	92.5	96.4	05.9	04.5	54.7	68.3	42.5	31.1
Sc_Programming	87.2	92.2	12.6	11.1	54.6	64.9	43.3	31.3	51.3	64.7	49.7	34.8	92.9	96.7	06.0	04.7	55.4	67.7	43.2	30.5
Avg of Class F/SC	88.1	91.9	13.1	11.3	54.7	65.3	43.5	31.3	50.9	64.8	49.3	34.9	92.6	96.5	05.9	04.6	55.0	67.9	42.8	30.7
Pathology Image																				
NCI01	88.2	92.0	11.4	09.6	71.0	82.0	23.4	17.9	66.0	81.3	31.1	20.0	89.7	95.2	06.7	06.6	71.1	83.3	21.7	17.4
NCI20	88.1	91.9	11.2	09.3	70.8	79.8	23.1	17.7	65.8	80.0	30.9	19.8	89.4	95.1	06.5	06.3	70.9	82.1	21.5	17.3
GBM2	88.0	91.7	11.0	09.2	71.4	82.8	24.3	18.4	66.4	80.7	31.5	20.5	90.5	96.1	07.1	06.9	71.5	82.7	22.1	18.0
COAD1	89.1	92.9	12.2	10.6	71.3	84.5	24.0	18.3	66.2	80.2	31.3	20.0	90.2	95.9	07.1	06.5	71.4	82.2	22.0	17.4
Avg of pathology	88.4	92.1	11.4	09.7	71.2	82.3	23.7	18.1	66.1	80.3	31.2	20.1	90.0	95.5	06.9	06.6	71.2	82.3	21.8	17.6
Overall average	89.7	93.4	12.2	10.4	60.0	73.2	39.6	28.1	56.4	71.6	44.2	30.2	92.4	95.9	07.7	06.8	62.0	74.4	36.6	26.8

Sequence	*Percentage of coefficients used for reconstruction*																			
	No offline training																			
	DCT								DCT/DST											
	PE ↑				MSE ↓				PE ↑				MSE ↓							
	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%
4:2:0 YUV sequences																				
Class A																				
Traffic	65.2	82.3	38.9	21.2	63.0	80.0	40.1	24.9												
People_on_street	63.1	82.4	38.1	20.3	62.2	78.8	39.0	23.4												
Nebuta_festival	63.2	78.5	34.4	20.1	58.5	75.1	39.3	21.4												
Avg of Class A	63.8	81.1	37.2	20.6	61.3	78.0	39.5	23.2												
Class B																				
Kimono	41.6	60.0	63.7	46.4	41.4	58.2	63.1	47.4												
Cactus	40.4	58.1	62.6	46.0	40.3	57.1	62.4	46.0												
Park_scene	40.1	58.1	62.0	45.1	40.4	57.4	61.7	45.9												
BQTerrace	41.5	56.4	59.9	44.2	39.9	56.3	61.9	44.8												
Avg of Class B	40.9	58.1	62.0	45.4	40.5	57.3	62.3	46.0												
Class C																				
Race_horse	46.3	63.2	57.0	39.1	46.3	62.2	58.2	42.2												
BQMall	46.0	62.8	55.9	38.8	45.1	61.0	55.1	41.7												
Party_scene	45.0	62.1	55.0	38.2	44.1	60.3	56.0	40.0												
Basketball_drill	43.1	61.4	53.8	39.2	41.1	60.1	57.6	38.0												
Avg of Class C	45.1	62.4	55.4	38.8	44.2	60.9	56.7	40.5												
Class D																				
Race_horse_D	51.6	68.7	50.4	33.7	50.6	67.1	51.3	35.6												
Blowing_bubble	50.2	67.4	49.6	33.2	49.5	66.2	50.8	34.3												
BQ_square	50.1	67.3	49.3	33.0	49.1	66.0	50.1	34.1												
Basketball_pass	50.5	67.9	49.4	32.4	49.2	66.1	51.4	34.7												
Avg of Class D	50.6	67.8	49.7	33.1	49.6	66.4	50.9	34.7												
Class E																				
Kristine_and_Sara	58.9	76.0	42.0	25.4	58.2	75.0	43.2	27.0												
Four_people	58.8	75.9	41.9	25.3	58.1	74.6	43.1	26.9												
Jhonny	59.4	75.9	42.7	27.2	58.7	75.7	43.7	27.3												
Avg of Class E	59.0	76.3	42.2	25.6	58.3	75.1	43.3	27.1												
Class F/SC																				
China_speed	46.4	63.6	54.4	37.9	50.1	63.1	55.3	39.0												
Slide_show	46.1	63.4	54.2	37.7	50.0	62.9	55.0	38.9												
Sc_Map	45.9	63.2	54.0	37.3	49.9	62.7	54.8	38.5												
Sc_Programming	45.8	63.1	53.7	37.1	49.8	62.2	54.7	38.5												
Avg of Class F/SC	46.1	63.3	54.1	37.5	50.0	62.7	55.0	38.7												
Pathology Image																				
NCI01	60.9	78.1	38.6	22.0	59.4	76.2	40.7	24.3												
NCI20	60.8	78.0	38.3	21.8	59.3	76.0	40.5	24.2												
GBM2	60.6	77.9	38.2	21.5	59.2	76.0	40.4	24.0												
COAD1	61.8	79.0	39.5	22.9	60.1	76.9	41.6	25.7												
Avg of pathology	61.0	78.3	38.7	22.1	59.5	76.3	40.9	24.6												
Overall average	52.4	69.6	48.5	31.9	51.9	68.1	49.8	33.5												

Table 6.4: PSNR (dB) values when using quantization on the transform coefficients.

Quantization parameters																											
Sequence	Resolution	Baseline				Offline training												No offline training									
		KLT				**GBT-CNN**				GBT-NN				GL-GBT				GBT-L$_A$				DCT				DCT/DST			
	QP	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37
Class A																													
Traffic	2560×1600	37.5	33.2	29.8	27.8	33.3	30.3	29.4	25.2	33.2	30.2	29.2	25.1	34.0	33.0	30.6	22.1	33.6	32.7	30.4	22.0	33.3	31.3	27.5	21.2	21.0	20.1	19.5	16.1
People_on_street		38.7	34.4	31.0	29.0	34.5	31.5	30.6	26.4	34.5	31.5	30.6	26.4	35.2	34.2	31.8	23.3	34.8	33.9	31.7	23.2	34.5	32.5	28.7	22.4	22.0	19.3	18.6	17.3
Nebuta_festival		39.5	35.2	31.8	29.8	35.3	32.3	31.4	27.2	35.3	32.2	31.4	27.2	36.0	35.0	32.6	24.1	35.6	34.7	32.4	24.0	35.3	33.3	29.5	23.2	21.7	20.2	19.5	17.0
Class B																													
Kimono	1920×1080	37.4	35.4	30.9	29.4	30.5	28.4	26.4	22.2	30.4	28.3	26.3	22.1	34.5	32.6	28.4	24.5	33.8	31.9	28.1	24.4	32.8	30.9	28.1	21.5	19.2	16.9	16.2	15.4
Cactus		38.1	36.1	31.7	30.0	31.2	29.0	27.	22.9	31.2	28.9	27.0	22.8	35.1	33.3	29.2	25.3	34.5	32.6	28.9	25.2	32.9	31.1	28.3	22.1	20.3	17.9	16.9	16.5
Park_scene		36.0	34.0	29.5	28.0	29.1	27.0	25.0	20.8	29.0	26.9	24.9	20.7	33.1	31.2	27.0	23.1	32.4	30.5	26.8	23.0	31.4	29.9	27.1	20.1	20.1	19.7	18.6	15.5
BQTerrace		38.8	36.7	32.3	30.7	35.1	31.7	30.5	23.5	35.0	31.6	30.4	23.4	35.8	33.4	29.8	25.9	35.1	33.3	29.6	25.9	32.9	31.1	28.4	22.8	21.6	20.9	20.4	17.7
Class C																													
Race_horse	832×480	38.8	36.0	30.7	23.8	30.8	28.6	26.7	22.5	30.6	28.5	26.5	22.3	34.8	32.9	28.7	24.8	34.5	32.6	28.4	24.6	33.1	30.0	28.1	21.8	20.5	16.8	16.2	15.6
BQMall		39.2	36.4	31.1	24.3	31.2	29.2	27.1	23.0	31.7	29.2	27.1	22.8	35.2	33.3	29.2	25.3	34.9	33.0	28.9	25.2	33.5	30.0	28.1	22.2	19.9	19.0	17.7	15.9
Party_scene		37.1	34.3	29.0	22.1	29.1	27.0	25.0	20.8	29.0	26.9	24.9	20.7	33.1	31.2	27.0	23.1	32.8	30.9	26.7	23.0	31.4	29.9	27.1	20.1	17.9	15.8	15.1	14.0
Basketball_drill		39.7	36.9	31.7	24.8	34.6	31.7	28.7	23.5	34.5	31.7	28.7	23.5	35.7	33.8	29.7	25.9	35.4	33.5	29.4	25.7	33.5	30.1	28.2	22.8	20.9	19.9	18.0	17.7
Class D																													
Race_horse_D	416×240	40.5	38.4	34.1	27.0	34.5	33.1	30.0	23.4	34.5	33.0	29.8	23.3	39.7	37.9	32.6	29.7	39.4	37.7	32.4	29.6	34.5	31.6	28.0	25.5	22.2	21.4	20.1	19.3
Blowing_bubble		41.9	40.0	35.4	28.4	37.9	35.6	32.4	24.7	37.9	35.4	32.2	24.6	41.1	39.2	34.0	31.1	40.9	39.0	33.8	31.0	35.9	23.6	29.0	22.4	23.6	22.4	21.1	20.6
BQ_square		39.1	37.0	32.7	25.6	33.1	31.7	28.6	22.0	33.1	31.6	28.4	21.9	38.3	36.5	31.2	28.3	38.0	36.2	31.0	28.2	33.1	31.2	27.0	24.1	20.8	20.0	19.0	17.9
Basketball_pass		42.2	40.3	35.7	28.7	37.9	36.8	32.7	25.0	37.9	36.7	32.5	24.9	41.4	39.5	34.3	31.3	41.1	39.2	34.1	31.2	36.0	32.9	29.3	23.8	23.7	22.7	21.3	20.9
Class E																													
Kristine_and_Sara	1280×720	40.6	39.3	35.5	30.7	36.4	34.6	31.9	25.0	36.3	34.5	31.7	24.9	36.7	33.7	31.3	26.7	36.0	32.9	31.0	28.2	36.3	34.0	31.2	26.3	24.0	22.3	21.2	21.1
Four_people		39.0	37.7	33.9	29.1	34.8	33.0	30.3	23.4	34.7	32.9	30.2	23.3	35.1	32.1	29.7	25.1	34.5	31.4	29.5	25.0	34.7	33.0	29.6	24.7	22.4	21.8	20.9	19.6
Jhonny		41.8	40.5	36.7	31.9	37.6	35.8	33.1	26.3	37.6	35.7	33.0	26.2	38.0	34.9	32.5	28.0	37.3	34.2	32.3	27.8	37.5	35.0	32.2	27.5	24.0	21.3	20.2	17.7
Class F/SC																													
China_speed	1024×768	40.2	37.0	33.9	33.1	36.1	33.7	27.4	24.6	36.0	33.6	27.2	24.5	36.9	34.0	30.5	25.1	36.3	33.3	30.3	25.0	34.9	32.8	29.6	26.0	20.4	18.7	18.1	17.9
Slide_show	1280×720	42.1	35.1	32.0	31.2	34.2	31.8	25.5	22.7	34.2	31.7	25.3	22.6	35.0	32.1	28.6	23.2	34.4	31.4	28.4	23.1	33.9	32.3	29.1	24.1	20.4	18.4	18.1	18.0
Sc_Map		40.9	37.6	34.6	33.8	36.7	34.3	28.1	25.2	36.7	34.2	27.9	25.1	37.5	34.7	31.2	25.8	36.9	34.0	31.0	25.7	35.5	32.8	30.3	26.7	23.0	19.4	18.8	18.6
Sc_Programming		41.2	37.9	34.9	34.1	40.1	36.6	31.4	25.5	40.1	36.5	31.2	25.4	37.9	35.0	31.6	26.1	37.2	34.3	31.4	26.0	35.6	32.8	30.3	27.0	23.3	22.7	22.3	21.9
Overall average		39.6	36.8	32.7	28.8	34.3	32.0	29.1	23.9	34.2	31.9	28.9	23.8	36.4	34.3	30.5	25.8	35.9	33.8	30.3	25.7	34.2	31.9	28.8	23.7	20.6	19.0	18.2	17.0

Table 6.5: Coding gain when using quantization on the transform coefficients.

Quantization parameters																											
Sequence	Resolution	Baseline				Offline training												No offline training									
		KLT				**GBT-CNN**				GBT-NN				GL-GBT				GBT-L$_A$				DCT				DCT/DST			
	QP	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37
Class A																													
Traffic	2560×1600	7.0	8.0	8.7	7.8	2.5	3.0	3.1	3.3	1.6	2.8	3.2	3.6	6.5	8.6	9.0	8.4	3.1	5.0	6.9	8.3	-1.0	1.7	1.8	1.9	-1.6	0.6	1.2	2.1
People_on_street		6.8	8.1	8.8	8.2	2.0	2.6	2.7	2.8	1.1	2.3	3.0	3.2	6.0	8.2	8.3	7.6	2.7	4.5	7.0	7.8	-1.8	1.3	1.4	1.5	-1.9	0.8	1.0	1.7
Nebuta_festival		6.6	7.6	9.2	8.3	2.4	3.1	3.2	3.5	1.2	2.7	3.4	3.7	6.1	8.1	8.5	7.7	3.2	4.6	6.5	8.2	-2.0	1.2	1.3	1.4	-2.8	0.4	0.8	1.9
Class B																													
Kimono	1920×1080	7.7	8.4	8.6	8.7	1.9	3.0	3.4	4.0	1.0	2.4	3.2	3.5	6.1	8.3	9.3	8.5	2.7	4.5	6.3	7.5	0.1	1.0	1.2	1.3	0.2	0.5	-0.4	0.9
Cactus		7.5	8.0	8.3	8.4	2.3	2.9	3.0	3.1	1.2	2.3	3.3	3.4	6.2	8.0	9.2	7.9	2.1	3.9	6.0	7.1	-1.8	0.8	1.1	1.2	-2.8	0.9	-0.7	0.8
Park_scene		7.4	8.3	8.9	8.2	2.1	2.6	3.1	3.3	0.8	2.2	2.9	3.2	5.3	7.4	8.6	7.3	2.3	4.2	6.2	7.4	-2.0	0.4	0.6	0.7	-2.1	0.4	-0.3	1.2
BQTerrace		7.0	8.1	8.2	8.3	1.7	2.7	3.3	3.6	0.6	1.9	3.0	3.1	6.0	7.9	8.9	7.5	2.5	3.8	5.9	7.2	-2.3	0.2	0.3	0.4	-1.3	0.6	-0.6	1.5
Class C																													
Race_horse	832×480	6.8	7.9	8.0	8.1	2.0	2.4	2.9	3.2	1.4	2.6	2.9	3.1	6.2	7.6	8.7	7.2	2.5	4.8	6.3	7.9	-1.4	1.3	1.4	1.5	-1.9	0.5	1.5	2.3
BQMall		7.1	8.0	8.3	8.4	1.6	2.0	2.2	2.5	1.0	2.1	2.3	2.7	5.4	7.4	8.3	7.3	2.7	5.5	6.7	8.2	-1.1	0.9	1.0	1.1	-0.8	0.2	1.1	1.9
Party_scene		7.2	8.3	8.5	7.8	1.9	2.6	3.0	3.4	1.1	2.3	2.5	2.6	5.5	7.0	8.0	6.9	3.0	4.5	6.5	8.4	-1.5	0.7	0.8	0.9	-1.2	0.3	1.4	2.2
Basketball_drill		6.9	8.2	8.4	8.5	1.7	2.2	2.7	2.9	1.3	3.0	3.1	3.2	5.3	7.2	8.6	7.0	2.2	5.2	6.9	8.7	-1.3	1.1	1.2	1.3	-2.1	0.2	1.2	2.0
Class D																													
Race_horse_D	416×240	6.5	7.0	8.7	8.5	2.0	2.4	3.1	3.3	0.7	2.0	2.2	3.0	5.9	8.0	9.5	7.4	3.3	3.9	6.6	6.8	-2.2	1.1	1.2	1.3	-1.2	0.6	-0.1	1.8
Blowing_bubble		6.2	7.3	8.5	7.8	2.4	3.0	3.4	3.7	0.9	2.6	3.0	3.6	6.1	7.7	8.9	7.0	2.9	3.8	7.0	7.1	-1.6	0.8	0.9	1.0	-1.4	0.3	-0.3	1.4
BQ_square		6.1	7.6	8.4	7.5	1.8	2.6	3.0	3.2	0.5	2.1	2.2	3.3	5.8	7.4	9.1	7.3	3.0	4.2	7.2	7.5	-1.8	1.0	1.1	1.2	-2.4	0.1	-0.2	1.5
Basketball_pass		6.4	7.7	8.8	7.8	2.2	2.8	3.3	3.4	1.1	2.5	2.6	3.7	6.2	8.1	9.3	7.1	3.2	4.1	6.8	7.4	-2.0	0.7	0.8	0.9	-2.2	0.2	-0.2	1.7
Class E																													
Kristine_and_Sara	1280×720	6.4	7.5	7.9	7.0	1.6	2.0	2.4	2.5	1.2	2.4	2.7	2.8	6.0	7.6	9.2	7.2	2.3	4.2	6.3	7.9	-2.0	0.5	0.6	0.7	-1.0	0.3	0.4	1.3
Four_people		6.8	7.9	8.3	7.5	1.5	2.4	2.9	3.0	1.0	2.8	3.2	3.3	5.6	7.2	8.5	7.1	1.8	3.4	5.6	7.4	-1.5	0.2	0.3	0.4	-0.6	0.2	0.3	0.8
Jhonny		6.3	7.4	7.8	7.1	2.0	3.0	3.1	3.2	1.3	2.9	3.1	3.2	5.5	7.1	9.0	7.6	2.2	4.1	5.8	7.5	-1.9	0.1	0.2	0.4	-0.8	0.1	0.2	0.9
Class F/SC																													
China_speed	1024×768	6.0	7.2	8.2	8.3	2.0	2.5	2.6	3.6	0.9	2.3	2.5	3.1	5.6	7.9	8.3	8.0	2.4	4.9	6.3	6.8	-1.3	0.8	0.9	1.0	-2.9	0.5	-0.3	1.5
Slide_show	1280×720	5.5	6.8	7.8	7.9	1.1	1.9	2.0	2.4	0.6	2.0	2.1	2.8	5.2	7.4	8.0	7.6	1.8	3.6	5.9	6.4	-1.6	0.3	0.4	0.5	-2.3	0.4	-0.9	1.2
Sc_Map		5.9	7.1	8.4	8.5	1.6	2.8	2.9	3.7	0.8	2.2	2.4	2.7	5.5	7.8	8.2	8.0	2.1	4.5	6.0	6.7	-1.7	0.7	0.8	0.9	-2.5	0.6	-0.2	1.4
Sc_Programming		5.8	6.9	8	8.1	1.3	2.4	2.5	3.1	0.5	1.9	2.2	3.4	5.3	7.3	7.9	7.2	1.7	3.4	6.2	6.1	-1.4	0.2	0.3	0.4	-2.7	0.1	-1.4	1.1
Overall average		6.6	7.7	8.4	8.0	1.9	2.6	2.9	3.2	1.0	2.4	2.8	3.2	5.8	7.7	8.7	7.5	2.5	4.3	6.4	7.5	-1.6	0.8	0.9	1.0	-1.8	0.4	0.1	1.5

Table 6.6: 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
GL-GBT [136]	53.23	45.18	**92.37**	**07.66**	**95.92**	**06.81**
KLT	55.51	44.49	89.73	12.22	93.43	10.43
DCT	17.49	82.27	52.41	48.48	69.58	31.88
DCT/DST	16.94	82.74	51.89	49.81	68.14	33.49
GBT-NN [9]	18.97	78.72	55.43	44.46	72.40	28.94
GBT-CNN (ours)	**21.45**	**76.36**	**59.92**	**39.12**	**73.16**	**27.9**
GBT-L_A [8]	24.71	75.17	60.47	40.21	74.28	26.47
GBT-L_W [8]	17.01	82.82	52.58	47.93	69.18	31.86

Table 6.7: Average PSNR and coding gain when using quantization on the transform coefficients.

	Quantization Parameters							
	QP=22		QP=27		QP=32		QP=37	
	PSNR	Gain	PSNR	Gain	PSNR	Gain	PSNR	Gain
GL-GBT	39.63	5.80	**36.92**	**7.68**	**33.05**	**8.66**	28.45	7.50
KLT	**40.22**	**6.58**	36.05	7.66	32.71	8.43	**29.62**	**8.03**
DCT	35.21	-1.63	31.02	0.77	28.29	0.95	23.07	0.99
DCT/DST	20.56	-1.83	19.02	0.45	18.28	0.13	17.10	1.48
GBT-NN	35.86	1.03	31.69	2.43	29.18	2.80	23.93	3.22
GBT-CNN (ours)	**36.13**	**1.95**	**32.73**	**2.65**	**29.90**	**2.91**	**25.02**	**3.25**
GBT-L_A	36.69	2.46	33.84	4.32	30.31	6.42	25.73	7.58
GBT-L_W	35.71	-0.46	31.58	-0.72	28.55	0.16	23.16	1.54

of the residual block depending on the prediction mode used. Note that our approach differs from GL-GBT since that method does not use any deep learning. We use the MSE to assess how efficiently the normalized symmetric adjacency matrix is predicted in comparison to the ground truth. Table 6.2 tabulates the performance of the five trained GBT-CNNs on the test data in terms of the MSE.

We first 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, since the efficiency of a transform is measured by its de-correlating properties and the maximum energy it concentrates in only a few transform coefficients. We set a threshold that indicates the minimum absolute value of the coefficients to be used for the reconstruction. This strategy gradually includes the largest coefficients in a subset by gradually lowering an initial large threshold as discussed in Chapter 3 in Fig. 3.5 [8]. Table 6.3 presents the PE (%) and MSE values for all evaluated data using a small percentage of the largest coefficients. The GBT-CNN preserves 19.41% and 14.98% more energy than the DCT/DST and the DCT, respectively, if only 5% of the largest coefficients are used. We find that the GBT-L_A outperforms the GBT-CNN; however, since the GBT-L_A requires graph information to compute the inverse transform, this transform is not practical as it significantly increases

the overhead. Note that the GL-GBT outperforms all other transforms. Plots in Fig. 6.5 (a, b) show the PE (%) and MSE values vs. the percentage of coefficients used for reconstruction of several transforms for the sequence *BlowingBubble* (Class D).

6.3.3 Objective quality evaluation

We also compute the reconstruction quality attained by the evaluated transforms in terms of the PSNR when quantization is used. Specifically, we employ four quantization parameters (QPs) used by the HEVC and VVC standards: $QP = \{22, 27, 32, 37\}$. Table 6.4 tabulates PSNR values for the evaluated frames/ images when these QPs are applied to the transform coefficients. Note that the proposed GBT-CNN outperforms both the DCT/DST and DCT by 7.92 dB and 1.95 dB, respectively, when QP=37. Again, the GBT-L_A outperforms our method by 0.71 db since this method uses actual residuals for reconstruction. Fig. 6.5 (c) plots the PSNR values for *People_on_Street* of Class A.

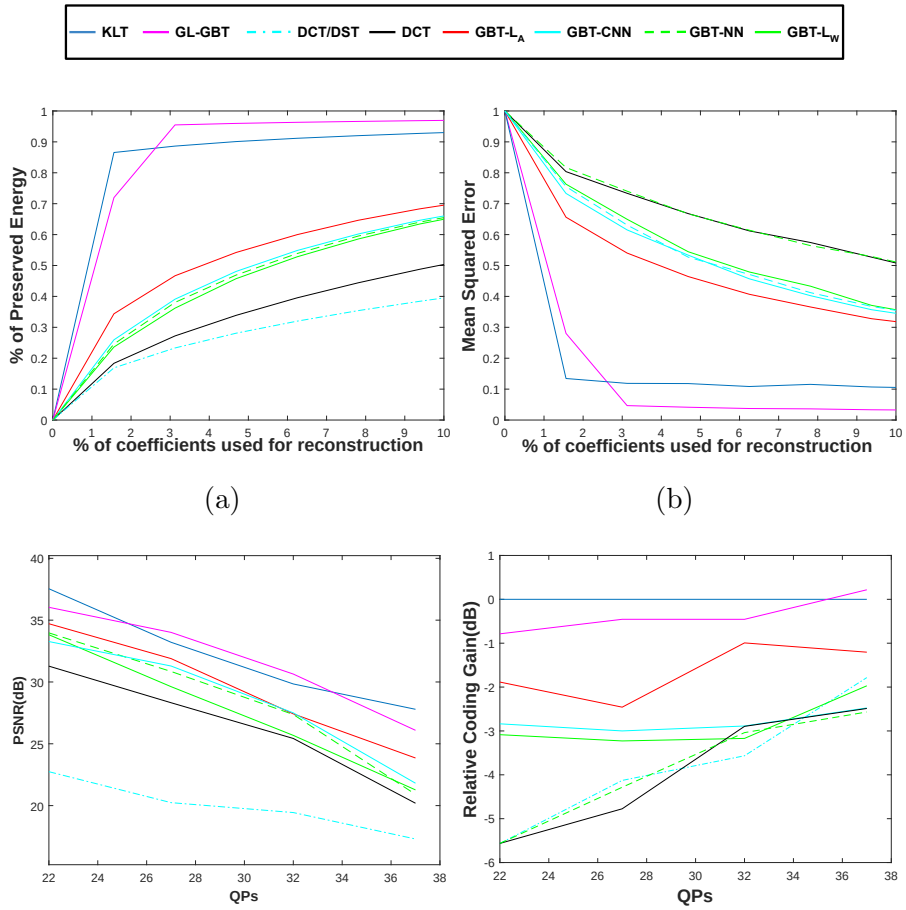
6.3.4 Coding gain

To demonstrate the rate-distortion trade off among the evaluated transforms, we also compute the transform coding gain in decibels (see Table 6.5), as the ratio of the distortion incurred between the uncoded and the coded frames [137]:

$$G_T(dB) = 10log_{10}(\frac{D_U}{D_T}), \tag{6.5}$$

where D_U is the distortion caused by applying direct quantization to the residuals and then dequantizing them to reconstruct the frames, while D_T is the distortion caused by quantization of the transformed coefficients of transform T and then reconstructing the frames after dequantization and inverse transformation. The distortion is measured in terms of the MSE. Table 6.5 shows that the GBT-CNN outperforms the DST/DCT and the DCT by 3.78 dB and 3.58 dB respectively, when QP=22. Plots in Fig. 6.5 (d) show the coding gain of several transforms for the sequence *People_on_Street* (Class A) relative to the KLT, computed as $G_T - G_{KLT}$, where G_T, is obtained is the coding gain for transform T and G_{KLT} is the coding gain for KLT . Fig. 6.6 shows a reconstructed frame of the sequence *BlowingBubble* (Class D) after transformation by several transforms and quantization with QP= 37. As depicted, the GBT-CNN achieves a higher visual reconstruction quality than the DCT. The GL-GBT achieves a visual quality very close to that achieved by the KLT.

For easier interpretation and analysis of the findings for PE-MSE, PSNR, and Coding-gain, Tables 6.6 and Table 6.7 provide a more comprehensive



(a) (b)

(c) (d)

Figure 6.5: (a) Preserved Energy (b) Mean-squared Error, when used up to 10% of largest coefficients for reconstruction, (c) PSNR for the sequence, and (d) Relative coding gain *BlowingBubble* of Class D.

overview of the results.

Any GBT requires eigendecomposition of the Laplacian graph. However, the eigendecomposition used by the KLT tends to be more complex as it uses a dense matrix. On the other hand, the sparsity in the graph Laplacian for the GBT can be controlled by the graph topology, which can lead to a lower computational complexity. Unfortunately, the GBT is just as computationally expensive as the KLT for the case of the All-C topology. In terms of learnable parameters, the network used by the GBT-CNN requires 4723510 parameters. The architecture used by the GBT-NN [9] requires 22368256 learnable parameters.

6.4 Summary

In this chapter, we proposed the GBT-CNN, a new class of GBTs that performs efficiently in block-based PTC with intra-prediction. The GBT-CNN is based on a 3D-CNN that learns a mapping function to approximate a symmetric

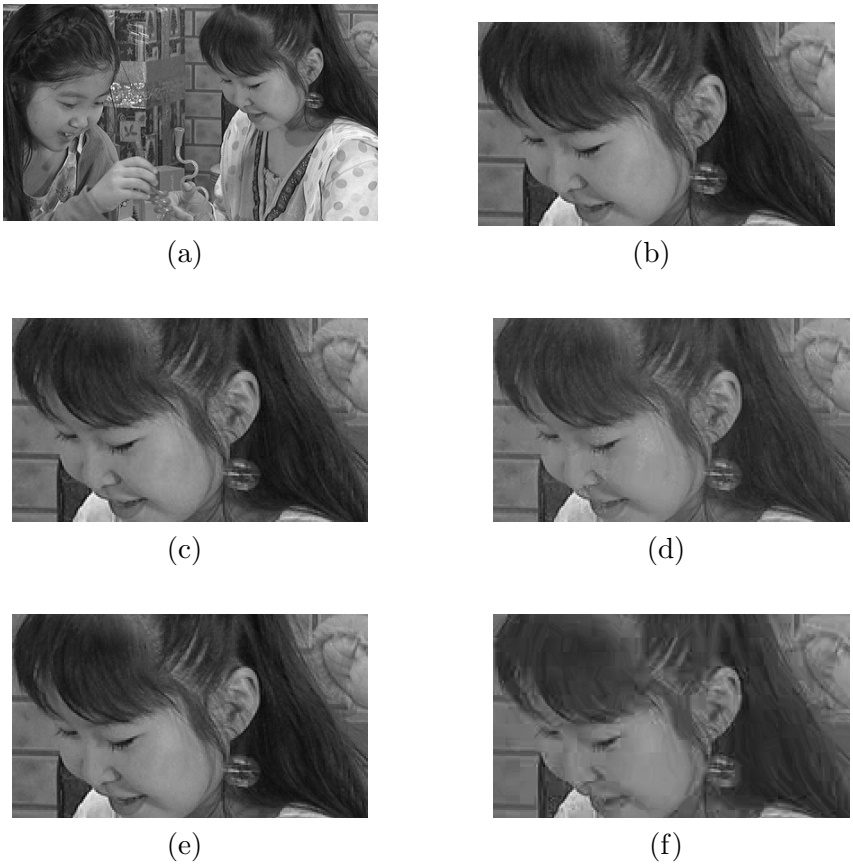


Figure 6.6: (a) An original frame of sequence *BlowingBubble* (Class D). (b) An area reconstructed after using the KLT (c) the proposed GBT-CNN , (d) the GBT-NN (e) the GL-GBT , and (f) the DCT . In all cases, QP=37.

adjacency matrix associated with the graph of the residual block to be encoded. We evaluated the performance of the GBT-CNN 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. We also compared the coding gain of the evaluated transforms. The evaluation results show that the proposed GBT-CNN outperforms the DCT and the DCT/DST, while the GL-GBT achieves the best performance, surpassing the KLT.

In both Chapter 5 and Chapter 6, we propose a framework for learning GBTs using DL, in which the encoder and decoder of a video codec that use block-based PTC are considered to share knowledge of a trained NN. As a result of this, our technique avoids the need to indicate any extra information in the compressed bit-stream. However, it does necessitate training a NN offline using the proper training data first. However, the volume, caliber, and relevance of the training data are what determine how well these ML-based algorithms perform. As a result, online optimization of the ML model has become a viable alternative to the offline learning procedure. We proposed online training to predict the graph in our next chapter, which relates to developing ML models while observing data and avoiding the use of pre-trained models.

Chapter 7

Online Graph-based Transforms for Intra-Predicted Imaging Data

7.1 Introduction

In this chapter, we leverage online training to learn GBTs without requiring of any training data or offline training processes. We specifically propose an online GBT, hereinafter called GBT-ONL, for block based PTC in the context of intra-prediction. The GBT-ONL predicts the graph Laplacian needed to compute the GBT of each block by using an over-fitted NN that is optimized online. The GBT-ONL has two main contributions. First, it introduces an online learning framework where the model is optimized as blocks are being encoded. This allows the model to adapt to each block to accurately predict the graph needed to compute the GBT. Second, since the training is performed online, it can be replicated at the decoder, thus avoiding the need to signal extra information to compute the inverse GBT for reconstruction.

7.2 Proposed GBT-ONL

Online optimization aims to learn a mapping function based on a sequence of samples as the samples are observed by the model. Such a mapping function is expected to perform a specific task based on the observed samples, for example, classification or regression. In the case of online optimization of an FC-NN, the mapping function is learned by defining the parameters of the network as samples are being observed. Those parameters are expected to perform the task very well for the current sample. Specifically, the parameters are usually initialized to random values and are updated sequentially using only the data being processed. To improve performance, the parameters learned for

the current sample can be used as the initial set of parameters to be updated for the next sample.

Our work concentrates on learning GBTs online within the context of block-based PTC. We focus on intra-prediction as currently performed by the HEVC and VVC standards; however, this work is codec-agnostic and can be used with any video and image codec that uses block-based PTC with intra-prediction. More specifically, our GBT-ONL framework relies on a shallow FC-NN trained over several iterations of gradient descent (GD) to predict the graph Laplacian required to compute the GBT for the current residual block. Once the optimization of the current residual block is complete, the FC-NN is optimized to predict the graph Laplacian of the subsequent residual block using as the initial set of parameters those optimized for the previous block. This process is repeated until all residual blocks are processed.

The process of defining the initial set of parameters to be used for the current block can be expressed mathematically for any two consecutive blocks with indices k and $k+1$ as follows:

$$\mathbf{W}_{k+1}^{0} \leftarrow \tilde{\mathbf{W}}_{k}, \tag{7.1}$$

where $\mathbf{W}_{k}^{0}$ and $\tilde{\mathbf{W}}_{k}$ denote the initial and final set of parameters for block $k+1$ and block k, respectively. For the first residual block of a frame, our shallow FC-NN uses parameters initialized to known values, i.e., $\mathbf{W}_{k=0}^{0}$ is known for block $k=0$. Moreover, the FC-NN uses information obtained from the blocks that have been already processed by block-based PTC as the input. Consequently, no additional side information needs to be stored to repeat the same training process during the reconstruction of the blocks since the same input is available when blocks are reconstructed sequentially and in the same order used to compute their GBTs. Let us recall that such sequential encoding and decoding are common in modern video and image codecs that use block-based PTC.

By using online optimization, our main objective is to learn a mapping function, $\mathtt{f}(\cdot)$, between an *average residual block* for the current block k, denoted by $\mathbf{C}_k$, and the graph Laplacian of such an average residual block:

$$\hat{\mathbf{L}}_{k} \approx \mathtt{f}(\mathbf{C}_{k}), \tag{7.2}$$

where $\hat{\mathbf{L}}_{k}$ is the predicted graph Laplacian of the average residual block and $\mathbf{C}_{k} = \frac{1}{3}\sum_{d=1}^{3}\mathbf{M}_{d}$ is computed as the average residual of the three residual blocks surrounding the current block, k. Here, $\mathbf{M}_{d}$ represents the d^{th} surrounding residual block (see Fig. 7.1).

Note that the same three residual blocks, $\{\mathbf{M}_{d}\}$, are available when re-

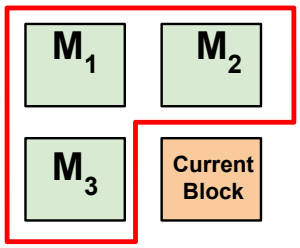


Figure 7.1: Surrounding residual blocks of the current residual block to be encoded.

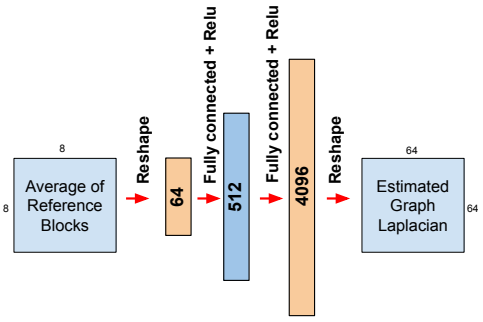


Figure 7.2: Architecture of the shallow FC-NN used by the proposed GBT-ONL framework for 8×8 blocks.

constructing a frame block-by-block. Hence, they can be used as an input to the same FC-NN for predicting the same graph Laplacian, which is needed to compute the inverse GBT for block k. The rationale behind using these three surrounding residual blocks is based on their similarities with the current residual block. These three blocks are expected to have similar characteristics to those of the residual block to be transformed. For residual blocks located in the corner or along the edges of a frame, which may not be surrounded by three residual blocks, we use a residual block with a constant value equal to the DC value of the frame, e.g., for 8 bbp images, we use a value of 128.

Our solution to learn the mapping function in Eq. 7.2 is based on an encoding-decoding shallow FC-NN, as depicted in Fig. 7.2 for the case of 8×8 blocks. This shallow FC-NN has an input layer of 64 neurons and a 512-neuron hidden layer, leading to the output layer of 4096 neurons. The average residual block, $\mathbf{C}_k$, in vector form, denoted by $\mathbf{c}_k$ is used as input to the FC-NN, which is trained to predict the graph Laplacian of $\mathbf{C}_k$, also in vector form and denoted by $\mathbf{l}_{\mathbf{c}_k}$. In other words, $\mathbf{l}_{\mathbf{c}_k}$ serves as the *ground truth* for the training process. Under the assumption that the three surrounding residual blocks $\{\mathbf{M}_d\}$ are similar to the residual block to be transformed, the graph Laplacian $\mathbf{l}_{\mathbf{c}_k}$ is then expected to be similar to that of the current residual block, denoted by $\mathbf{l}_k$; hence $\mathbf{l}_{\mathbf{c}_k} \approx \mathbf{l}_k$.

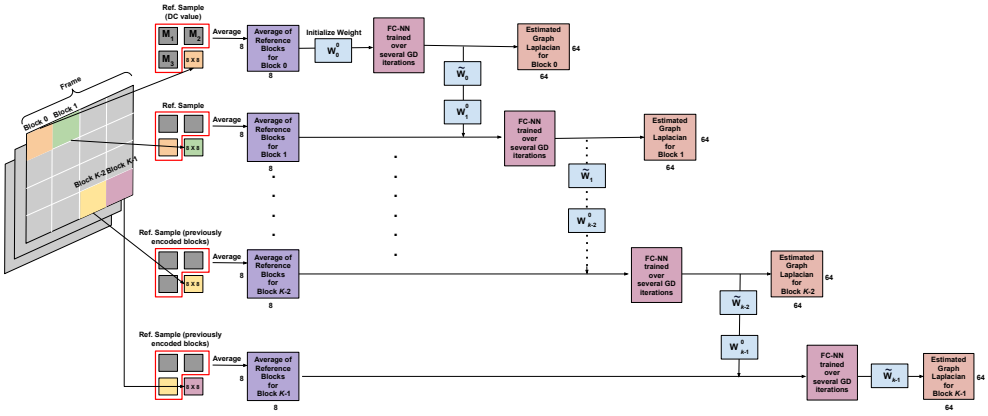


Figure 7.3: Sequential processing of blocks by the proposed GBT-ONL framework. To estimate the graph Laplacian of the current block, k, the shallow FC-NN uses the average residual block in vector form, denoted by $\mathbf{c}_k$.

Fig. 7.3 shows the overall functionality of the GBT-ONL framework to predict the graph Laplacian for each residual block k under block-based PTC using intra-prediction. Note that the shallow FC-NN is optimized for each block k over several iterations of gradient descent (GD) to accurately predict $\mathbf{l}_{\mathbf{c}_k}$. This optimization process is stopped based on threshold ξ, i.e., when the Mean Squared Error (MSE) between the predicted graph Laplacian, $\hat{\mathbf{l}}_{\mathbf{c}_k}$, and the corresponding ground truth, $\mathbf{l}_{\mathbf{c}_k}$, is less than ξ, or when enough iterations of GD have been performed. In other words, the FC-NN is overfitted on the input $\mathbf{c}_k$. Note that this overfitting process is appropriate for block-based PTC as the prediction is based on a single input. The weights found after optimizing the FC-NN on block k, i.e., $\tilde{\mathbf{W}}_k$, are used as the initial weights for block $k+1$ (see Eq. 7.1). The process is repeated for all K residual blocks in the frame.

Algorithm 1 summarizes the online optimization process used by the GBT-ONL framework, where P is the maximum number of GD iterations to be performed for the current block, α is the learning rate, $Arc(\cdot)$ denotes the architecture of the shallow FC-NN, $\{DC_{value}\}$ is a reference block in vector form with all values equal to the DC of the image, $\mathbf{m}_d$ is the d^{th} reference residual block in vector form, and $\{r, c\}$ denotes the row and column, respectively, where a block is located in the image. Line 1 of the algorithm iterates over all K residual blocks. Line 3 initializes all the parameters of the FC-NN for the first block to a value of 0.5. Line 4 calculates the average residual block for block $k = 0$ as a block with DC values. Line 6 initializes the parameters of the FC-NN to the parameters found for the previous block. Lines 7 -13 initialize the average residual block according to the position of the current block k to account for any unavailable residual block $\mathbf{m}_d$. Line 15 optimizes the FC-NN for block k over a maximum of P iterations of GD. Line 16 computes the predicted graph Laplacian and the optimized parameters of the FC-NN

Algorithm 1: Online training of the GBT-ONL framework for an image with K blocks.

Require: $\left\{\{\mathbf{m}_d\}, \mathbf{l}_{\mathbf{c}_k}\right\}$ for each block k, $Arch(\cdot), P, \alpha, \xi$, $\{DC_{value}\}$

1: **for** $k = 0 \rightarrow (K-1)$ **do**
2: **if** $k = 0$ **then**
3: $\mathbf{W}^0_{k=0} = \{0.5\}$
4: $\mathbf{c}_k \leftarrow \{DC_{value}\}$
5: **else**
6: $\mathbf{W}^0_k \leftarrow \tilde{\mathbf{W}}_{k-1}$
7: **if** $r = 0$ AND $c! = 0$ **then**
8: $\mathbf{c}_k \leftarrow \frac{1}{3}(\mathbf{m}_3 + \{DC_{value}\} + \{DC_{value}\})$
9: **elseif** $r! = 0$ AND $c = 0$ **then**
10: $\mathbf{c}_k \leftarrow \frac{1}{3}(\mathbf{m}_2 + \{DC_{value}\} + \{DC_{value}\})$
11: **else**
12: $\mathbf{c}_k \leftarrow \frac{1}{3}\sum_{d=1}^{3} \mathbf{m}_d$
13: **end**
14: **end**
15: **for** $p = 1 \rightarrow P$ **do**
16: $\{\hat{\mathbf{l}}_{\mathbf{c}_k}, \mathbf{W}^p_k\} \leftarrow Arch(\mathbf{c}_k, \mathbf{l}_{\mathbf{c}_k}, \alpha, \mathbf{W}^0_k)$
17: **if** $||\hat{\mathbf{l}}_{\mathbf{c}_k} - \mathbf{l}_{\mathbf{c}_k}||_2^2 > \xi$ **then**
18: $\mathbf{W}^0_k \leftarrow \mathbf{W}^p_k$
19: go to line 16
20: **else**
21: $\tilde{\mathbf{W}}_k \leftarrow \mathbf{W}^p_k$
22: **return** $\hat{\mathbf{l}}_{\mathbf{c}_k}$
23: **end**
24: **end**
25: **end**

for interation p of GD, denoted by $\mathbf{W}^p_k$. The optimization is based on the following loss function:

$$\mathcal{L}(\hat{\mathbf{l}}_{\mathbf{c}_k}, \mathbf{l}_{\mathbf{c}_k}) = \| \hat{\mathbf{l}}_{\mathbf{c}_k} - \mathbf{l}_{\mathbf{c}_k} \|_2^2 + \lambda \| \mathbf{W}^p_k \|_2^2, \tag{7.3}$$

where $\| \cdot \|_2$ is the L2 norm and λ is a hyperparameter to control the level of L2-regularization on $\mathbf{W}^p_k$. Line 17 computes the square of the error between the ground truth and the predicted graph Laplacian and checks if this squared error is above the threshold ξ. If this squared error is above ξ, the parameters found after iteration p of GD are used as the initial set of parameters to be further optimized in Line 16. Otherwise, Line 21 defines the final set of parameters as those found at iteration p and Line 22 returns the predicted graph Laplacian, which is to be used to compute the GBT for block k.

Fig. 7.4 shows how the proposed GBT-ONL framework can be incorporated into an encoder-decoder pipeline that uses block-based PTC for compression

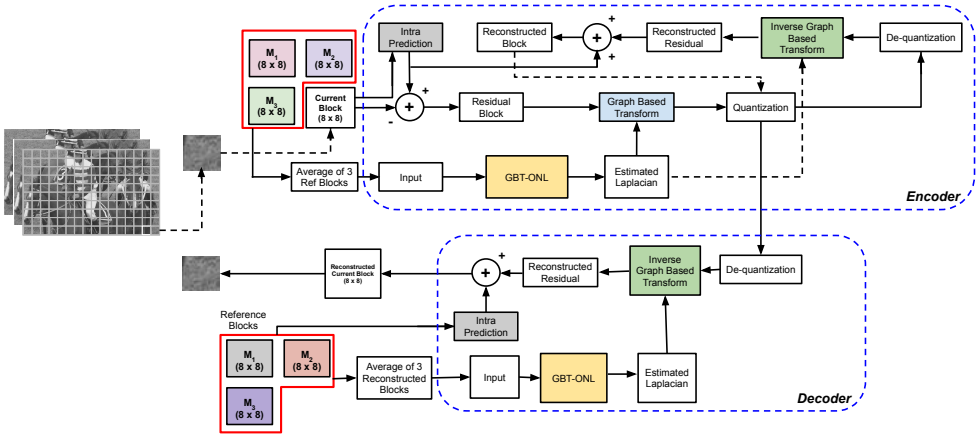


Figure 7.4: The GBT-ONL framework incorporated into an encoder-decoder system that uses block-based PTC for compression.

with intra-prediction. Our framework assumes that the initial parameters $\mathbf{W}_{k=0}^{0}$ of the shallow FC-NN for block $k = 0$ are *common knowledge* between the encoder and decoder. Note that the residual blocks $\{\mathbf{M}_d\}$ used to compute the average residual block $\mathbf{C}_k$ are those available at the encoder after the corresponding blocks are processed and reconstructed. This guarantees that even after quantization of the corresponding transform coefficients, these residual blocks are the same as those available at the decoder.

7.3 Performance evaluation

Our test dataset comprises several 4:2:0 YUV video sequences commonly used to test the performance of the HEVC and VVC standards. These sequences are organized into six classes: A, B, C, D, E, and F/SC listed in the CTC of JCT-VC [138]. They cover a wide range of characteristics in terms of length, smoothness, scene complexity, and type of content. Table 2.1 illustrates the characteristics of the videos used for the experiments. We also use 10 pathology images in RGB format from the Center for Biomedical Informatics and Information Technology of the US National Cancer Institute [128] in the evaluation. To predict the test sequences using intra-prediction and compute the residual blocks, we use blocks of 8×8 pixels and the 35 modes that are common to the HEVC and VVC standards. We use the mode that provides the lowest residual signal for block k. Note that modern video codecs usually make intra-prediction decisions on the Y and G components of Y:U:V and RGB data, respectively. For this reason, we use only these components of our test data.

We compare our proposed method with several GBTs. Specifically, we compare the GBTs, which vary in the nature of training, the topology of the

graph, the edge weights, and how the graph is obtained to compute the GBT. Apart from proposed GBT-ONL we compare other two GBTs constructed differently from our proposed GBT-ONL. For the first kind, the GBT use covariance matrices from several training examples to estimate the graph Laplacian, hereinafter called GL-GBT [136]. This approach mainly focus on offline training with traditional machine learning approach. For the second one, called GBT-CNN [10], the graph has unit edge weights with no self-loop. This method focuses on training a 3D-CNN model offline. Our experiments includes other transforms, such as, the KLT, DCT, and DCT/DST as used in the HEVC and VVC standards are also evaluated, with the DST being employed as separable transforms for rows and columns of the residual block depending on the prediction mode used. KLT is used as the baseline for optimality. Overall, we categorize the evaluated transforms in two kind: i) transform requires offline training, and ii) transform requires no offline training.

In the following subsections we summarize our evaluations and experiments performed on several transforms. In subsection 7.3.1 we evaluate the proposed model with appropriate metric. Further, subsection 7.3.2 shows the experimental results for unquantized co-efficient. Additionally, we tabulate and explain the compression and reconstruction quality, respectively, in subsections 7.3.3 and 7.3.4.

7.3.1 Model evaluation

To first evaluate the performance of the NN by using mean absolute error (MAE) metric. The FC-NN has 2 hidden layers of 64 and 512 neurons in input and 4096 neurons in output layer (see Fig. 7.2). In order to receive the same performance at the encoder and decoder we initialize the weight of 0.5 which works as a common knowledge. To overfit the network, we apply several steps of gradient descent with the stopping criteria of $\xi \leq 1e^{-8}$ or reach n cycles. Due to the online approach each block of the frame is tested sequentially while passing the updated parameter to the next block. We experienced loss minimization in every block initially starting with a comparatively poor performance. However, due to overfitting and passing the updated parameter from the previous block makes the prediction more accurate with high regressor. As an instance, Table 7.1 shows the performance of *Blowing_bubble*. Since the frame is divided into 1560 non-overlapping blocks the loss gradually decreases for the later blocks.

For all experiments, we initialize the parameters of the FC-NN used by the GBT-ONL framework to a value of 0.5 for the first block of each frame, i.e., $\mathbf{W}^0_{k=0} = \{0.5\}$. To optimize the FC-NN, we use up to $P = 100$ iterations of GD with a threshold of $\xi = 1e^{-8}$. In other words, the optimization process is stopped at block k after $P = 100$ iterations of GD steps or if the MSE

between the predicted and ground truth graph Laplacians is less than or equal to $\xi = 1e^{-8}$. As explained in Section 7.2, the blocks are processed sequentially and the parameters optimized for block k are used as the initial set of parameters for block $k+1$.

7.3.2 Evaluation of energy compaction for unquantized coefficients

We first 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, since the efficiency of a transform is measured by its de-correlating properties and the maximum energy it concentrates in only a few transform coefficients. We set a threshold that indicates the minimum absolute value of the coefficients to be used for the reconstruction. This strategy gradually includes the largest coefficients in a subset by gradually lowering an initial large threshold. Fig. 3.5 shows a toy example of the mechanism followed.

Table 7.2 presents the PE (%) and MSE values for all evaluated sequences using a small percentage of the largest coefficients. The transforms are categorised based on the training approach. In our experiments GBT-CNN and GL-GBT needs to train their model offline, where as, DCT, DCT/DST and our proposed method GBT-ONL do not use any offline training. The GBT-ONL preserves 3.13% more energy than the DCT for Class A whereas overall 3.65% more energy than DCT for all the test sequence if only 5% of the largest coefficients are used. For unquantized co-efficients GL-GBT outperforms KLT in terms of PE and MSE for natural image of Class D (see Fig. 7.5). Recall, GL-GBT uses an offline trained model whose performance completely depends on amount and relevance of training data whereas our model adapts the new pattern of data without any training sample.

7.3.3 Objective quality evaluation

We compute the reconstruction quality attained by the evaluated transforms in terms of the PSNR when quantization is used. Specifically, we employ four quantization parameters (QPs) used by the HEVC and VVC standards: $QP = \{22, 27, 32, 37\}$. Table 7.3 tabulates PSNR values for the evaluated frames/ images when these QPs are applied to the transform coefficients. Note that the proposed GBT-ONL outperforms DCT and DCT/DST by 2.1 dB and 7.2 dB respectively for *Traffic* Class A, when $QP = 37$ whereas for the same class the PSNR of DCT outperforms GBT-ONL, when $QP = 27$. For Class B, DCT outperforms GBT-ONL and GL-GBT by 0.7 dB and 0.1 dB respectively, for $QP = 32$. For Class B, GBT-ONL outperforms DCT

by 0.47 dB for $QP = 37$. On the other hand, DCT for *BQsquare* Class D sequence outperforms GBT-ONL for PSNR (see Fig. 7.5). The most challenging sequence are the screen content where there are several edges. For *SC_Programming* GBT-ONL outperforms DCT by 2.4 dB, 2.1 dB, 0.6 dB respectively, for $QP = 22$, $QP = 27$, and $QP = 32$. The GL-GBT outperforms our method by 1.8 dB for $QP = 22$ for all the test sequence since this method uses offline training for predicting graph Laplacian.

7.3.4 Bjontegaard-based (BD) metric

We evaluate the compression quality of our proposed method in terms of BD-PSNR and BD-BR (bit-rate). As a trade-off to any HM software used for compression, we use the entropy [139] of the evaluated transforms as the lower limit for the average coding length in bits per pixel for 4 different QPs. We use entropy of DCT as a reference average bit-rate to compare other transforms. Then we use different PSNR for specified QP values to compare the other transforms with respect to DCT. Table 7.4 tabulates the BD-PSNR and BD-BR for the test sequences of different classes. Our table shows better compression performance for proposed method with respect to DCT and DCT/DST.

For simplified evaluation and comprehension of the results for the PE-MSE, PSNR, and BD metrics, Tables 7.5, Table 7.6, and Table 7.7 give an expanded summary of the results respectively.

Fig. 7.6 shows a reconstructed frame of the sequence *Basketball_drill* after transformation by several transforms and quantization with $QP = 37$. As depicted, the GBT-ONL achieves a higher visual reconstruction quality than the DCT. The GL-GBT achieves a visual quality very close to that achieved by the KLT. Since our network is fully connected and performed for each block the parameters we learned ($n \times 4096 \times 512 \times 4096$) where n is the number of blocks in the sequence.

Table 7.1: Performance evaluation of training networks for a sequences

(a) *Blowing_bubble*

	Metric
	MAE
	Updated initial weight
Block 0	268.513
Block 50	151.194
Block 100	73.290
Block 300	28.671
Block 600	10.248
Block 1000	7.301
Block 1200	5.458
Block 1559	2.173

7.4 Summary

In this chapter, we proposed the GBT-ONL, a new class of GBTs that performs efficiently in block-based PTC with intra-prediction. The GBT-ONL is based on online optimization of the block to be encoded. We also present a coding/decoding framework that allows employing the GBT-ONL on residual blocks without the need to signal information about the graphs to the decoder. The framework uses reference samples while decoding. We evaluated the performance of the GBT-ONL 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. We also compared the compression efficiency using BD-PSNR and BD-BR of the evaluated transforms. The evaluation results show that the proposed GBT-ONL outperforms the DCT and the DCT/DST, while the GL-GBT achieves the best performance among offline trained methods, surpassing the KLT in some cases. Note that among any *non-trainable* approach, GBT-ONL is outperforming others. GBT-ONL is remarkably performing well in terms of computational complexity compared to the other offline trained methods since we use a shallow network. This method is advantageous in adapting new patterns in data without the use of any training data or pre-trained model. The performance of this method on unstructured graphs is still unclear, and its only drawback is that GBT-ONL is only applied to structured graphs. The future direction leads to extract the features of the video content online with convolution approach and learn the parameters which is expected to response well for blocks with similar content all over the sequences.

Table 7.2: PE (in %) and MSE using a small percentage of the largest coefficients.

Sequence	*Percentage of coefficients used for reconstruction*																							
	Baseline				Offline training								No offline training											
	KLT				GBT-CNN				GL-GBT				**GBT-ONL**				DCT				DCT/DST			
	PE↑		MSE ↓		PE ↑		MSE ↓		PE↑		MSE ↓		PE ↑		MSE ↓		PE ↑		MSE ↓		PE ↑		MSE ↓	
	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%	5%	10%
4:2:0 YUV sequences																								
Class A																								
Traffic	89.1	93.1	13.9	12.1	67.1	85.0	37.8	19.9	90.1	93.9	14.6	11.8	**67.1**	**83.5**	**38.0**	**20.4**	65.2	82.3	38.9	21.2	63.0	80.0	40.1	24.9
People_on_street	88.8	92.4	12.1	11.4	65.3	84.0	36.9	18.5	89.1	93.1	12.8	10.4	**65.2**	**83.7**	**37.1**	**19.1**	63.1	82.4	38.1	20.3	62.2	78.8	39.0	23.4
Nebuta_festival	87.2	90.9	13.6	10.8	63.8	82.8	33.6	16.5	87.4	90.2	11.4	10.5	**63.5**	**80.2**	**33.9**	**18.2**	63.2	78.5	34.4	20.1	58.5	75.1	39.3	21.4
Avg of Class A	88.4	92.2	13.2	11.5	65.4	83.9	36.2	18.3	88.9	92.4	13.0	10.9	65.3	82.5	36.3	19.2	63.8	81.1	37.2	20.6	61.3	78.0	39.5	23.2
Class B																								
Kimono	93.2	98.3	14.8	12.4	49.1	61.0	55.9	41.9	96.5	99.9	08.6	06.9	**43.5**	**60.5**	**61.6**	**46.2**	41.6	60.0	63.7	46.4	41.4	58.2	63.1	47.4
Cactus	92.6	97.4	13.1	12.2	48.5	60.2	55.1	41.2	96.	99.9	08.0	06.5	**42.4**	**59.8**	**60.7**	**44.4**	40.4	58.1	62.6	46.0	40.3	57.1	62.4	46.0
Park_scene	92.1	94.8	13.5	10.9	48.2	60.0	54.9	40.9	95.8	99.9	07.9	06.4	**42.7**	**59.6**	**59.7**	**43.7**	40.1	58.1	62.0	45.1	40.4	57.4	61.7	45.9
BQTerrace	93.0	95.4	11.6	10.5	49.0	60.1	55.0	41.0	96.0	99.9	08.0	06.7	**43.5**	**58.3**	**57.8**	**42.1**	41.5	56.4	59.9	44.2	39.9	56.3	61.9	44.8
Avg of Class B	92.7	96.5	13.3	11.5	48.7	60.3	55.2	41.3	96.1	99.9	8.1	06.7	43.0	59.6	60.0	44.1	40.9	58.1	62.0	45.4	40.5	57.3	62.3	46.0
Class C																								
Race_horse	92.0	95.9	09.5	07.9	52.9	70.3	46.8	35.6	93.9	94.5	06.1	06.0	48.1	**65.4**	**55.2**	**36.9**	46.3	63.2	57.0	39.1	46.3	62.2	58.2	42.2
BQMall	89.8	93.3	09.7	07.1	52.2	70.0	46.3	35.2	93.5	93.9	06.1	06.1	**48.0**	**64.5**	**53.9**	**37.1**	46.0	62.8	55.9	38.8	45.1	61.0	55.1	41.7
Party_scene	90.3	94.2	08.9	06.9	52.0	69.8	46.1	35.1	93.4	93.6	06.1	06.1	**47.2**	**64.1**	**52.8**	**36.2**	45.0	62.1	55.0	38.2	44.1	60.3	56.0	40.0
Basketball_drill	94.5	98.1	08.7	07.7	53.0	70.1	46.7	35.7	93.8	94.4	06.1	05.9	**47.5**	**63.3**	**49.4**	**37.3**	43.1	61.4	53.8	39.2	41.1	60.1	57.6	38.0
Avg of Class C	91.6	95.4	09.1	07.4	52.5	70.1	46.5	35.4	93.7	94.1	06.1	06.0	47.7	64.3	52.8	36.9	45.1	62.4	55.4	38.8	44.2	60.9	56.7	40.5
Class D																								
Race_horse_D	92.0	95.0	11.6	09.6	57.9	69.9	38.2	29.6	91.8	95.8	06.0	06.0	**53.4**	**69.1**	**48.6**	**33.3**	51.6	68.7	50.4	33.7	50.6	67.1	51.3	35.6
Blowing_bubble	90.7	94.5	10.0	08.4	58.4	70.3	38.8	30.0	92.1	97.1	06.2	06.2	**52.4**	**69.3**	**47.4**	**31.3**	50.2	67.4	49.6	33.2	49.5	66.2	50.8	34.3
BQ_square	90.2	94.2	09.8	08.1	58.0	70.1	38.5	29.9	92.0	95.9	06.1	06.1	**52.4**	**69.2**	**49.3**	**31.1**	50.1	67.3	47.0	33.0	49.1	66.0	50.1	34.1
Basketball_pass	89.8	94.0	09.3	07.6	57.8	70.0	38.3	29.7	92.0	94.6	06.0	06.0	**52.3**	**69.4**	**47.6**	**30.9**	50.5	67.9	49.4	32.4	49.2	66.1	51.4	34.7
Avg of Class D	90.7	94.4	10.2	08.4	58.0	70.1	38.5	29.8	92.0	95.9	06.1	06.1	52.6	69.3	47.7	31.7	50.6	67.9	49.1	33.1	49.6	66.4	50.9	34.7
Class E																								
Kristine_and_Sara	87.8	91.7	14.8	13.1	70.0	80.8	33.7	22.9	93.9	96.8	07.8	07.0	**60.3**	**78.1**	**40.6**	**23.3**	58.9	76.0	42.0	25.4	58.2	75.0	43.2	27.0
Four_people	87.6	91.5	48.4	13.0	69.3	79.9	33.1	21.9	93.1	95.9	07.3	06.5	**60.6**	**77.5**	**40.1**	**23.7**	58.8	75.9	41.9	25.3	58.1	74.6	43.1	26.9
Jhonny	88.5	91.9	15.8	13.6	69.3	80.1	33.6	21.8	93.4	96.1	07.9	06.7	**61.3**	**77.7**	**40.8**	**25.4**	59.4	75.9	42.7	27.2	58.7	75.7	43.7	27.3
Avg of Class E	87.9	91.7	15.0	13.2	69.5	80.3	33.5	22.2	93.4	96.3	07.7	06.7	60.7	77.8	40.5	24.1	59.0	76.3	42.2	25.6	58.3	75.1	43.3	27.1
Class F/SC																								
China_speed	89.0	92.4	14.0	12.1	55.1	66.1	44.2	31.9	92.7	96.9	06.0	04.8	**48.5**	**65.5**	**52.3**	**36.0**	46.4	63.6	54.4	37.9	50.1	63.1	55.3	39.0
Slide_show	88.3	91.9	13.5	11.4	54.3	65.0	43.0	30.9	92.2	96.2	05.8	04.3	**48.1**	**64.7**	**52.2**	**36.4**	46.1	63.4	54.2	37.7	50.0	62.9	55.0	38.9
Sc_Map	87.9	90.9	12.3	10.9	54.6	65.2	43.4	31.2	92.5	96.4	05.9	04.5	**47.7**	**65.1**	**52.2**	**35.4**	45.9	63.2	54.0	37.3	49.9	62.7	54.8	38.5
Sc_Programming	87.2	92.2	12.6	11.1	54.6	64.9	43.3	31.3	92.9	96.7	06.0	04.7	**47.6**	**64.1**	**51.9**	**36.1**	45.8	63.1	53.7	37.1	49.8	62.2	54.7	38.5
Avg of Class F/SC	88.1	91.9	13.1	11.3	54.7	65.3	43.5	31.3	92.6	96.5	05.9	04.6	48.0	64.9	52.2	36.0	46.1	63.8	54.1	37.5	50.0	62.7	55.0	37.3
Pathology Image																								
NCI01	88.2	92.0	11.4	09.6	71.0	82.0	23.4	17.9	89.7	95.2	06.7	06.6	**62.1**	**80.3**	**38.6**	**19.8**	60.9	78.1	37.4	22.0	59.4	76.2	40.7	24.3
NCI20	88.1	91.9	11.2	09.3	70.8	79.8	23.1	17.7	89.4	95.1	06.5	06.3	**62.2**	**79.2**	**36.5**	**20.6**	60.8	78.0	38.3	21.8	59.3	76.0	40.5	24.2
GBM2	88.0	91.7	11.0	09.2	71.4	82.8	24.3	18.4	90.5	96.1	07.1	06.9	**62.4**	**79.7**	**36.4**	**19.7**	60.6	77.9	38.2	21.5	59.2	76.0	40.4	24.0
COAD1	89.1	92.9	12.2	10.6	71.3	84.5	24.0	18.3	90.2	95.9	07.1	06.5	**63.1**	**81.3**	**38.2**	**20.6**	61.8	79.0	39.5	22.9	60.1	76.9	41.6	25.7
Avg of pathology	88.4	92.1	11.4	09.7	71.2	82.3	23.7	18.1	90.0	95.5	06.9	06.6	62.6	80.1	37.1	20.2	61.0	78.3	38.7	22.1	59.5	76.3	40.9	24.6
Overall average	89.7	93.4	12.2	10.4	60.0	73.2	39.6	28.1	92.4	95.9	07.7	06.8	**54.3**	**71.2**	**46.6**	**30.3**	52.4	69.6	48.5	31.9	51.9	68.1	49.8	33.5

Table 7.3: PSNR (dB) values when using quantization on the transform coefficients.

Quantization parameters																								
Sequence	Resolution	Baseline				Offline training								**No offline training**											
		KLT				GBT-CNN				GL-GBT				**GBT-ONL**				DCT				DCT/DST			
	QP	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37	22	27	32	37
Class A																									
Traffic	2560×1600	37.5	33.2	29.8	27.8	33.3	30.3	29.4	25.2	34.0	33.0	30.6	22.1	**33.4**	30.9	**28.5**	**23.3**	33.3	**31.3**	27.5	21.2	21.0	20.1	19.5	16.1
People_on_street		38.7	34.4	31.0	29.0	34.5	31.5	30.6	26.4	35.2	34.2	31.8	23.3	**34.6**	32.2	**29.7**	**24.4**	34.5	**32.5**	28.7	22.4	22.0	19.3	18.6	17.3
Nebuta_festival		39.5	35.2	31.8	29.8	35.3	32.3	31.4	27.2	36.0	35.0	32.6	24.1	**35.4**	32.9	**30.4**	**25.3**	35.3	**33.3**	29.5	23.2	21.7	20.2	19.5	17.0
Class B																									
Kimono	1920×1080	37.4	35.4	30.9	29.4	30.5	28.4	26.4	22.2	34.5	32.6	28.4	24.5	31.7	29.8	27.4	**22.0**	**32.8**	**30.9**	**28.1**	21.5	19.2	16.9	16.2	15.4
Cactus		38.1	36.1	31.7	30.0	31.2	29.0	27.	22.9	35.1	33.3	29.2	25.3	32.1	30.2	27.8	**22.6**	**32.9**	**31.1**	**28.3**	22.1	20.3	17.9	16.9	16.5
Park_scene		36.0	34.0	29.5	28.0	29.1	27.0	25.0	20.8	33.1	31.2	27.0	23.1	30.5	28.6	26.3	**20.6**	**31.4**	**29.9**	**27.1**	20.1	20.1	19.7	18.6	15.5
BQTerrace		38.8	36.7	32.3	30.7	35.1	31.7	30.5	23.5	35.8	33.4	29.8	25.9	**34.1**	**31.6**	**29.5**	**23.3**	32.9	31.1	28.4	22.8	21.6	20.9	20.4	17.7
Class C																									
Race_horse	832×480	38.8	36.0	30.7	23.8	30.8	28.6	26.7	22.5	34.8	32.9	28.7	24.8	32.2	29.5	27.7	**22.3**	**33.1**	**30.0**	**28.1**	21.8	20.5	16.8	16.2	15.6
BQMall		39.2	36.4	31.1	24.3	31.2	29.2	27.1	23.0	35.2	33.3	29.2	25.3	32.5	29.8	27.8	**22.8**	**33.5**	**30.0**	**28.1**	22.2	19.9	19.0	17.7	15.9
Party_scene		37.1	34.3	29.0	22.1	29.1	27.0	25.0	20.8	33.1	31.2	27.0	23.1	30.6	28.6	26.4	**20.6**	**31.4**	**29.9**	**27.1**	20.1	17.9	15.8	15.1	14.0
Basketball_drill		39.7	36.9	31.7	24.8	34.6	31.7	28.7	23.5	35.7	33.8	29.7	25.9	**34.3**	**31.1**	**28.7**	**23.3**	33.5	30.1	28.2	22.8	20.9	19.9	18.0	17.7
Class D																									
Race_horse_D	416×240	40.5	38.4	34.1	27.0	34.5	33.1	30.0	23.4	39.7	37.9	32.6	29.7	**34.5**	**32.5**	**29.0**	24.0	34.5	31.6	28.0	**24.4**	22.2	21.4	20.1	19.3
Blowing_bubble		41.9	40.0	35.4	28.4	37.9	35.6	32.4	24.7	41.1	39.2	34.0	31.1	**37.0**	**34.2**	**30.8**	25.3	35.9	23.6	29.0	**25.7**	23.6	22.4	21.1	20.6
BQ_square		39.1	37.0	32.7	25.6	33.1	31.7	28.6	22.0	38.3	36.5	31.2	28.3	**33.2**	**31.6**	**27.9**	22.6	33.1	31.2	27.0	**23.0**	20.8	20.0	19.0	17.9
Basketball_pass		42.2	40.3	35.7	28.7	37.9	36.8	32.7	25.0	41.4	39.5	34.3	31.3	**37.1**	**35.0**	**31.1**	25.6	36.0	32.9	29.3	**26.0**	23.7	22.7	21.3	20.9
Class E																									
Kristine_and_Sara	1280×720	40.6	39.3	35.5	30.7	36.4	34.6	31.9	25.0	36.7	33.7	31.3	26.7	**36.4**	**34.5**	**31.5**	25.8	36.3	34.0	31.2	**26.3**	24.0	22.3	21.2	21.1
Four_people		39.0	37.7	33.9	29.1	34.8	33.0	30.3	23.4	35.1	32.1	29.7	25.1	**34.8**	**33.2**	**30.0**	24.2	34.7	33.0	29.6	**24.7**	22.4	21.8	20.9	19.6
Jhonny		41.8	40.5	36.7	31.9	37.6	35.8	33.1	26.3	38.0	34.9	32.5	28.0	**37.6**	**35.6**	**32.6**	27.0	37.5	35.0	32.2	**27.5**	24.0	21.3	20.2	17.7
Class F/SC																									
China_speed	1024×768	40.2	37.0	33.9	33.1	36.1	33.7	27.4	24.6	36.9	34.0	30.5	25.1	**35.8**	**33.4**	28.8	25.4	34.9	32.8	**29.6**	**26.0**	20.4	18.7	18.1	17.9
Slide_show	1280×720	42.1	35.1	32.0	31.2	34.2	31.8	25.5	22.7	35.0	32.1	28.6	23.2	**34.4**	32.1	27.6	23.4	33.9	**32.3**	**29.1**	**24.1**	20.4	18.4	18.1	18.0
Sc_Map		40.9	37.6	34.6	33.8	36.7	34.3	28.1	25.2	37.5	34.7	31.2	25.8	**36.3**	**33.7**	29.4	26.1	35.5	32.8	**30.3**	**26.7**	23.0	19.4	18.8	18.6
Sc_Programming		41.2	37.9	34.9	34.1	40.1	36.6	31.4	25.5	37.9	35.0	31.6	26.1	**38.0**	**34.9**	**30.9**	26.4	35.6	32.8	30.3	**27.0**	23.3	22.7	22.3	21.9
Overall average		39.6	36.8	32.7	28.8	34.3	32.0	29.1	23.9	36.4	34.3	30.5	25.8	**34.4**	**32.1**	**29.1**	**23.9**	34.2	31.9	28.8	23.7	20.6	19.0	18.2	17.0

Table 7.4: BD-PSNR, BD-BR (with respect to DCT) values when using quantization on the transform coefficients.

	Methods									
	Baseline		*Offline training*				**No offline training**			
Sequence	KLT		GBT-CNN		GL-GBT		GBT-ONL (ours)		DCT/DST	
	BD-PSNR	BD-BR	BD-PSNR	BD-BR	BD-PSNR	BD-BR	BD-PSNR	BD-BR	BD-PSNR	BD-BR
Class A										
Traffic	08.16	-78.56	01.17	-14.87	4.69	-31.44	**01.01**	**-11.63**	-6.39	23.62
People_on_street	07.02	-80.64	00.74	-15.61	3.83	-33.14	**00.59**	**-13.47**	-7.01	21.11
Nebut_festival	07.29	-80.35	00.92	-15.02	3.96	-33.04	**00.81**	**-12.23**	-6.32	22.94
Class B										
Kimono	03.75	-67.21	0.45	-09.23	02.33	-37.76	**00.31**	**-06.44**	-4.11	19.47
Cactus	02.91	-68.23	0.67	-08.22	1.31	-38.69	**00.51**	**-06.08**	-4.54	19.93
Park_scene	03.92	-67.30	1.03	-13.22	02.24	-37.67	**00.90**	**-11.08**	-5.12	16.94
BQTerrace	03.06	-68.18	1.51	-14.79	1.36	-38.64	**1.35**	**-11.55**	-3.82	13.01
Class C										
Race_horse	10.83	-52.33	01.11	-12.43	5.17	-46.19	**00.97**	**-09.44**	-09.16	30.01
BQMall	09.99	-53.26	00.94	-14.99	4.15	-47.12	**00.78**	**-11.85**	-08.29	27.81
Party_scene	08.01	-52.24	02.26	-11.56	05.08	-46.10	**02.15**	**-08.95**	-06.77	20.76
Basketball_drill	10.13	-53.21	01.46	-12.03	04.20	-47.07	**01.30**	**-09.29**	-07.77	21.45
Class D										
Race_horse_D	10.67	-75.72	02.12	-09.32	08.50	-75.26	**01.98**	**-06.53**	-15.27	32.85
Blowing_bubble	09.65	-76.65	02.15	-09.12	-07.66	-76.28	**01.99**	**-06.98**	-14.04	31.51
BQ_square	10.58	-75.63	01.06	-12.94	08.67	-75.35	**00.95**	**-10.80**	-11.88	29.03
Basketball_pass	09.70	-76.60	01.65	-11.22	07.81	-76.23	**01.49**	**-07.98**	-14.96	32.59
Class E										
Kristine_and_Sara	08.47	-69.89	0.56	-17.34	02.52	-22.28	**00.40**	**-14.10**	-12.92	35.96
Four_people	07.33	-71.97	0.64	-17.02	01.66	-23.98	**00.49**	**-14.88**	-11.93	36.01
Jhonny	07.60	-71.68	1.01	-10.50	01.79	-23.88	**00.90**	**-07.71**	-10.72	19.34
Class F/SC										
China_speed	09.21	-69.05	0.33	-08.61	06.15	-40.27	**00.19**	**-05.82**	-14.09	35.90
Slide_show	08.52	-70.02	0.36	-07.92	05.18	-41.16	**00.20**	**-05.78**	-14.71	35.01
Sc_Map	08.37	-70.07	0.21	-06.22	05.13	-41.20	**00.08**	**-04.08**	-15.82	37.90
Sc_Programming	10.21	-69.14	0.34	-09.05	06.06	-40.19	**00.18**	**-05.81**	-14.28	36.03

Table 7.5: Average PE (in %) and MSE using a small percentage of the largest coefficients.

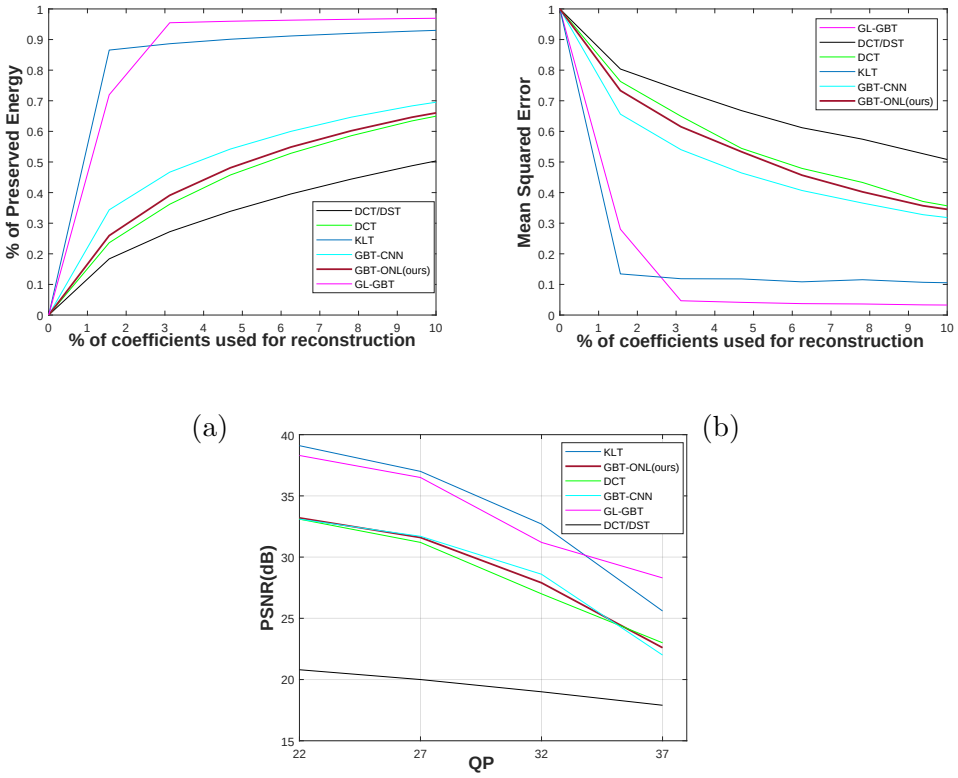
	Percentage of coefficients used					
	1%		5%		10%	
	PE	MSE	PE	MSE	PE	MSE
KLT	55.51	44.49	89.73	12.22	93.43	10.43
GBT-CNN	21.45	76.36	59.92	39.12	73.16	27.9
GL-GBT [136]	53.23	45.18	92.37	07.66	95.92	06.81
GBT-ONL(ours)	**18.56**	**78.78**	**54.33**	**46.6**	**71.23**	**30.3**
DCT	17.49	82.27	52.41	48.48	69.58	31.88
DCT/DST	16.94	82.74	51.89	49.81	68.14	33.49

Table 7.6: 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
GBT-CNN	36.13	32.73	29.90	25.02
GL-GBT [136]	39.63	36.92	33.05	28.45
GBT-ONL(ours)	**34.42**	**32.12**	**29.14**	**23.93**
DCT	35.21	31.02	28.29	23.07
DCT/DST	20.56	19.02	18.25	17.10

Table 7.7: Average BD-PSNR, BD-BR values when using quantization on the transform coefficients.

	Compression Quality	
	BD-PSNR	BD-BR
KLT	7.97	-69.00
GBT-CNN	1.03	-11.87
GL-GBT [136]	3.82	-44.22
GBT-ONL(ours)	**0.89**	**-9.20**
DCT/DST	-10.00	27.24



(c)

Figure 7.5: (a) Preserved Energy (b) Mean-squared Error, when used up to 10% of largest coefficients for reconstruction, and (c) PSNR for the sequence *BQsquare* of Class D.

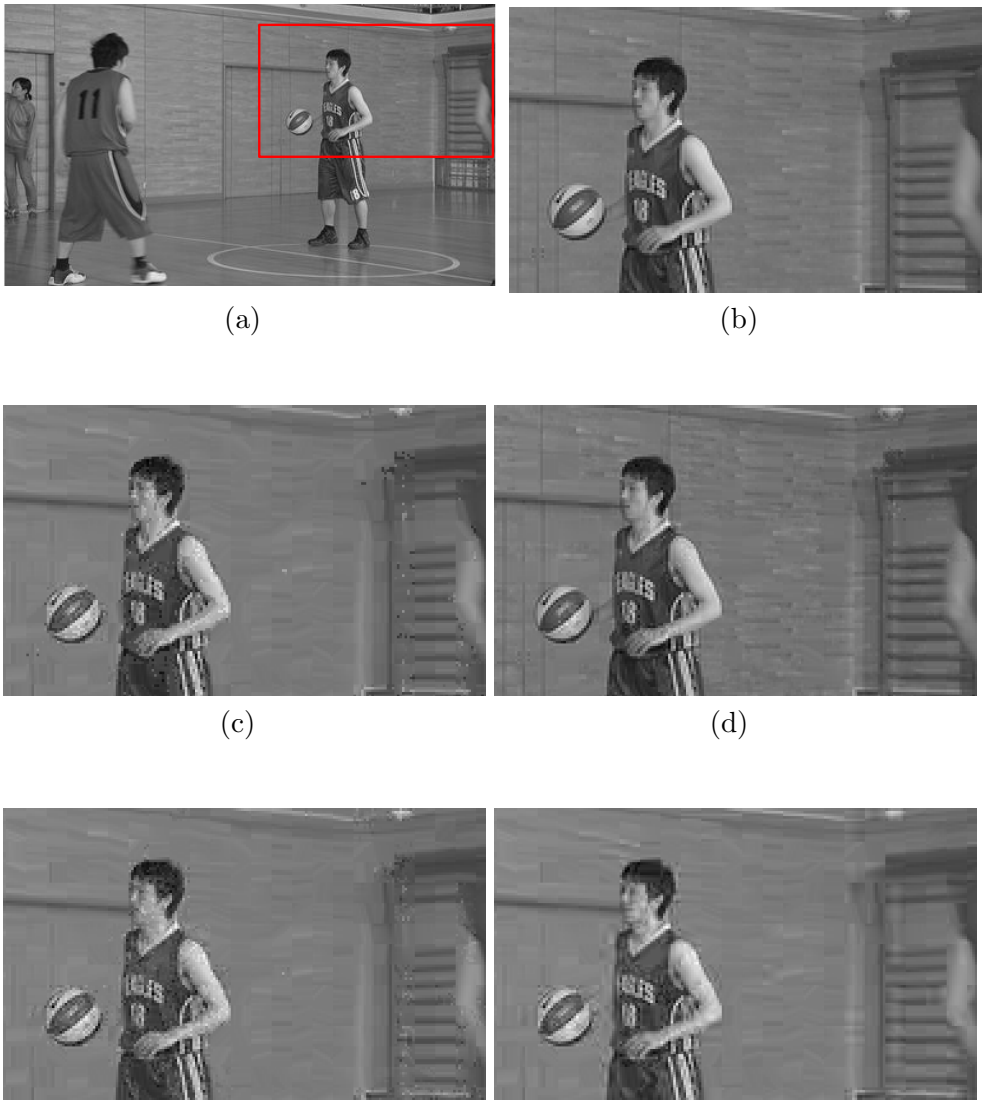


Figure 7.6: (a) An original frame of sequence *Basketball_pass* (Class D). (b) An area reconstructed after using the GL-GBT (PSNR = 31.3 dB), (c) the DCT (PSNR = 26.0 dB), (d) the KLT (PSNR = 28.7 dB), (e) the GBT-CNN (PSNR = 25.0 dB), and (f) the proposed GBT-ONL (PSNR = 25.6 dB). In all cases, $QP = 37$.

Chapter 8

Conclusions and Future Work

In this thesis, we have proposed a set of strategies for graph-based transforms for image and video coding. In this chapter, we summarise our contributions, discuss applications, limitations and consider future work.

8.1 Summary of Contributions

In Chapter 3 we have introduced the concept of reducing signaling overhead by proposing a framework that allows employing the GBT to transform intra-predicted residual signals of the multigiga-pixel WSI images images without the need to signal information about graphs to the decoder. This framework is only using reference samples to perform the IGBT for reconstruction. In Chapter 4 we have introduced a framework which uses a 2D graph with unit edge weights and weighted self-loops in every vertex while constructing the graphs for GBT. Similar to previous chapter we took care of the fact to reduce additional signaling overhead to the decoder. To accomplish this we have used template-based prediction techniques to predict the same graph at the decoder to perform IGBT. In Chapter 5 we have introduced NN architecture to predict the graph by learning the adjacency matrix. Our method assumes the trained GBT-NN is common knowledge between the compression and reconstruction processes. Hence we avoided any overhead signaling. We used the template-based prediction strategy from Chapter 4 to predict the residual at the decoder as an input to the GBT-NN. In Chapter 6.1 we exploited 3D CNN architecture to predict the graph at the decoder. Similar to Chapter 5 trained GBT-CNN is assumed to be the common knowledge between encoder and decoder. However, the novelty in this chapter is, we overcome the issue of applying two consecutive prediction techniques, (i) template matching for predicting residuals, (ii) GBT-NN to predict the graph with the help of 3D-CNN. In Chapter 7 we introduced online learning of graphs by using a shallow over-fitted NN. For its online learning strategy this framework allows to adapt

to each block to accurately predict the graph for GBT. The same online training is performed for IGBT, thus, avoids the overhead extra signaling.

8.2 Application

The methods proposed in this thesis have a range of applications which are yet to be applied in real life. We identify two main application domains.

- **Medical Imaging**:
  - **Internationally standard diagnosis**:
    International medical specialists make diagnoses based on how serious and critical a medical image is. An image can only lose a little amount of information in order to retain consistency in diagnosis. Our techniques keep the majority of the energy, thus the redundant information in the images and films is only slightly lost.
  - **Preserve**:
    It is well acknowledged that the availability of earlier imaging studies frequently has a significant impact on how a new study is interpreted since it enables the detection of changes in the findings and an estimation of the rate of any such change.

    Furthermore, there is a chance that in the future, new methods will allow for the use of data that has already been collected and stored in ways that aren't currently feasible. Unavoidably, new methods, settings, and insights will emerge. Then, information that was not immediately obvious during the initial examination may be deduced.

    Since our approach relies on an exact reflection of pixel correlation for graph structures, it is extremely adaptable to any new approaches that emerge over time. Our methods demonstrate a higher compression by evaluating the entropy encoding of the methods in maximising the utility of lossy compression and to limit the chance of compression to become orphan data.
  - **Transmission**:
    Bandwidth issues hinder transmission of an uncompressed or lossless compressed image. In these cases, the medical practitioners use of appropriate levels of lossy image compression to speed the initial arrival of the images, with the patient's interests foremost in their consideration. Our method allow more cost-effective utilization of network bandwidth and storage capacity for medical images.
- **Streaming Media**:

– **Live Streaming**:
Live streaming requires a form of source media (e.g. a video camera, an audio interface, screen capture software), an encoder to digitize the content, a media publisher, and a content delivery network to distribute and deliver the content. For this application our method can be adopted.

– **Media Upload and Download**: Compression of multimedia data is quite necessary now a days for upload and download purpose. In the era of COVID-19, the over-the-top (OTT) platforms that provide television and film content over the internet at the request and to suit the requirements of the individual consumer are quite active. Our methods can be adapted to accomplish this purpose.

8.3 Limitation

In our thesis we found the limitation as stated below:

- The proposed frameworks are only applied to HEVC standard videos. Unfortunately, due to some protocols of access issues by JCT-VC team we had to limit our experiments in HEVC videos.

- The evaluation results show that our proposed methods outperforms the most popular transforms DCT and the DCT/DST used in HEVC, while the GL-GBT, the SOTA achieves the best performance among offline trained methods, surpassing the KLT in some cases. Note that among any *non-trainable* approach, GBT-ONL in Chapter 7 is outperforming others to a great extent.

8.4 Future Work

Graph-based signal processing is a promising concept in the signal and image processing field. The potential of the compression abilities of this techniques provide solutions to problems of high definition videos. In this section, we describe the future work of our proposed frameworks as follows:

- The future direction leads to extend our research ideas in Chapter 7. In our existing framework 7.3 we use a shallow NN to predict the estimated Laplacian where sample covariance matrix is the input to the network. In future our plan is to extract the features of the video content online with convolution approach by involving the surrounding blocks of the block to be encoded. Instead of averaging the 3 surrounded blocks of the block to be encoded, we could use 3D convolution neural network

to extract the features to learn the parameters of those content. It is expected to response well to estimate Laplacian for blocks with similar content all over the sequences.

- Another most SOTA future direction leads to extend our research ideas in Chapter 5 and Chapter 6.1 is to introduce Graph Neural Network (GNN) to predict the graphs for transforms. This could specifically help to involve unstructured graph structures, such as, MRI images of brain where the focus is on blood flow direction. The graph representation could indeed encode the complex structure of the brain to indicate either physical or functional connectivity across different brain regions, and blood flow through veins are an example of a graph signal.

Bibliography

[1] P. Chaudhary, R. Gupta, A. Singh, P. Majumder, and A. Pandey, "Joint image compression and encryption using a novel column-wise scanning and optimization algorithm," *Procedia Computer Science*, vol. 167, pp. 244–253, 01 2020.

[2] "Lossless compression," https://www.pcmag.com/encyclopedia/term/lossless-compression/, [Online].

[3] V. Sanchez and J. Bartrina-Rapesta, "Lossless compression of medical images based on hevc intra coding," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 6622–6626.

[4] X. Xu and S. Liu, "Recent advances in video coding beyond the hevc standard," *APSIPA Transactions on Signal and Information Processing*, vol. 8, p. e18, 2019.

[5] "Dct transform," https://users.cs.cf.ac.uk/Dave.Marshall/Multimedia/node231.html/, [Online].

[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] 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.

[8] 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.

[9] ——, "Graph-based transform based on neural networks for intra-prediction of imaging data," in *2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP)*, 2021, pp. 1–6.

[10] ——, "Graph-based transform based on 3d convolutional neural network for intra-prediction of imaging data," in *2022 Data Compression Conference (DCC)*, 2022, pp. 212–221.

[11] J. Lainema, F. Bossen, W. 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.

[12] F. Zou, O. C. Au, C. Pang, and J. Dai, "Rate distortion optimized transform for intra block coding for hevc," in *Visual Communications and Image Processing (VCIP), 2011 IEEE*. IEEE, 2011, pp. 1–4.

[13] A. Sandryhaila and J. M. Moura, "Discrete signal processing on graphs," *IEEE transactions on signal processing*, vol. 61, no. 7, pp. 1644–1656, 2013.

[14] 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.

[15] H. E. Egilmez, A. Said, Y. H. Chao, and A. Ortega, "Graph-based transforms for inter predicted video coding," in *Image Processing, 2015 IEEE International Conference on*. IEEE, 2015, pp. 3992–3996.

[16] H. E. Egilmez, Y.-H. Chao, A. Ortega, B. Lee, and S. Yea, "Gbst: Separable transforms based on line graphs for predictive video coding," in *IEEE Int. Conf. Image Processing*. IEEE, 2016, pp. 2375–2379.

[17] A. Rosenfeld, "Picture processing by computer," *ACM Computing Surveys (CSUR)*, vol. 1, no. 3, pp. 147–176, 1969.

[18] 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.

[19] J. 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.

[20] B. Bross, J. Chen, J. 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.

[21] M. Rabbani and P. W. Jones, *Digital image compression techniques*. SPIE press, 1991, vol. 7.

[22] K. R. Castleman, *Digital image processing*. Prentice Hall Press, 1996.

[23] "Lossless compression," https://www.thebroadcastbridge.com/content/entry/16741/lossless-compression-rewriting-data-in-a-more-efficient-way/, [Online].

[24] A. Moffat, "Huffman coding," *ACM Computing Surveys (CSUR)*, vol. 52, no. 4, pp. 1–35, 2019.

[25] S. A. Aly, A. Klappenecker, and P. K. Sarvepalli, "On quantum and classical bch codes," *IEEE Transactions on Information Theory*, vol. 53, no. 3, pp. 1183–1188, 2007.

[26] I. H. Witten, R. M. Neal, and J. G. Cleary, "Arithmetic coding for data compression," *Communications of the ACM*, vol. 30, no. 6, pp. 520–540, 1987.

[27] A. E. Jacquin, "Fractal image coding: A review," *Proceedings of the IEEE*, vol. 81, no. 10, pp. 1451–1465, 1993.

[28] E. Delp and O. Mitchell, "Image compression using block truncation coding," *IEEE transactions on Communications*, vol. 27, no. 9, pp. 1335–1342, 1979.

[29] Y. Hu, W. Yang, Z. Ma, and J. Liu, "Learning end-to-end lossy image compression: A benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 8, pp. 4194–4211, 2022.

[30] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," *arXiv preprint arXiv:1703.00395*, 2017.

[31] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Deep convolutional autoencoder-based lossy image compression," in *2018 Picture Coding Symposium (PCS)*. IEEE, 2018, pp. 253–257.

[32] N. Johnston, D. Vincent, D. Minnen, M. Covell, S. Singh, T. Chinen, S. J. Hwang, J. Shor, and G. Toderici, "Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4385–4393.

[33] A. M. Rufai, G. Anbarjafari, and H. Demirel, "Lossy image compression using singular value decomposition and wavelet difference reduction," *Digital signal processing*, vol. 24, pp. 117–123, 2014.

[34] H. Nobuhara, W. Pedrycz, and K. Hirota, "Fast solving method of fuzzy relational equation and its application to lossy image compression/reconstruction," *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 3, pp. 325–334, 2000.

[35] Y. Wang, M. Xiao, C. Liu, S. Zheng, and T.-Y. Liu, "Modeling lost information in lossy image compression," 2020. [Online]. Available: https://arxiv.org/abs/2006.11999

[36] Y. L. Prasanna, Y. Tarakaram, Y. Mounika, and R. Subramani, "Comparison of different lossy image compression techniques," in *2021 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)*, 2021, pp. 1–7.

[37] A. Wenzel, E. Gotfredsen, E. Borg, and H.-G. Gröndahl, "Impact of lossy image compression on accuracy of caries detection in digital images taken with a storage phosphor system," *Oral Surgery, Oral Medicine, Oral Pathology, Oral Radiology, and Endodontology*, vol. 81, no. 3, pp. 351–355, 1996. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1079210496803362

[38] T. Mantoro and F. Alfiah, "Comparison methods of dct, dwt and fft techniques approach on lossy image compression," in *2017 International Conference on Computing, Engineering, and Design (ICCED)*, 2017, pp. 1–4.

[39] M. T. Pourazad, C. Doutre, M. Azimi, and P. Nasiopoulos, "Hevc: The new gold standard for video compression: How does hevc compare with h.264/avc?" *IEEE Consumer Electronics Magazine*, vol. 1, no. 3, pp. 36–46, 2012.

[40] J. CT-VC, "Encoder-side description of test model under consideration," in *Proc. JCT-VC Meeting*, 2010.

[41] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h.264/avc video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003.

[42] B. Bross, "High efficiency video coding (hevc) text specification draft 9 (sodis)," in *11th JCT-VC meeting, Oct. 2012*, 2012.

[43] ——, "High efficiency video coding (hevc) text specification draft 10 (for fdis & last call)," in *Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, 12th Meeting, Geneva,(Jan. 2013)*, 2013.

[44] J. Pfaff, A. Filippov, S. Liu, X. Zhao, J. Chen, S. De-Luxán-Hernández, T. Wiegand, V. Rufitskiy, A. K. Ramasubramonian, and G. Van der Auwera, "Intra prediction and mode coding in vvc," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 10, pp. 3834–3847, 2021.

[45] B. Bross, Y.-K. Wang, Y. Ye, S. Liu, J. Chen, G. J. Sullivan, and J.-R. Ohm, "Overview of the versatile video coding (vvc) standard and its applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 10, pp. 3736–3764, 2021.

[46] 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.

[47] B. Bross, K. Andersson, M. Bläser, V. Drugeon, S.-H. Kim, J. Lainema, J. Li, S. Liu, J.-R. Ohm, G. J. Sullivan *et al.*, "General video coding technology in responses to the joint call for proposals on video compression with capability beyond hevc," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 5, pp. 1226–1240, 2019.

[48] T. K. Tan, C. S. Boon, and Y. Suzuki, "Intra prediction by template matching," in *2006 International Conference on Image Processing*, 2006, pp. 1693–1696.

[49] S. He, Z. Deng, and C. Shi, "Fast decision algorithm of cu size for hevc intra-prediction based on a kernel fuzzy svm classifier," *Electronics*, vol. 11, no. 17, p. 2791, 2022.

[50] L. Tan, Y. Zeng, and W. Zhang, "Research on image compression coding technology," in *Journal of Physics: Conference Series*, vol. 1284, no. 1. IOP Publishing, 2019, p. 012069.

[51] "Lossless compression," http://www.petccufpb.com.br/wp-content/uploads/2009/06/KLT.pdf/, [Online].

[52] S. Huang, S. Quek, and K. Phoon, "Convergence study of the truncated karhunen–loeve expansion for simulation of stochastic processes," *International journal for numerical methods in engineering*, vol. 52, no. 9, pp. 1029–1043, 2001.

[53] P. K. Pandey, Y. Singh, and S. Tripathi, "Image processing using principle component analysis," *International Journal of Computer Applications*, vol. 15, no. 4, pp. 37–40, 2011.

[54] R. P. McDonald, "The theoretical foundations of principal factor analysis, canonical factor analysis, and alpha factor analysis," *British Journal of Mathematical and Statistical Psychology*, vol. 23, no. 1, pp. 1–21, 1970.

[55] H. Prasantha, H. Shashidhara, and K. B. Murthy, "Image compression using svd," in *International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007)*, vol. 3. IEEE, 2007, pp. 143–145.

[56] A. Chatterjee, "An introduction to the proper orthogonal decomposition," *Current science*, pp. 808–817, 2000.

[57] Y. Lu, T. Belytschko, and L. Gu, "A new implementation of the element free galerkin method," *Computer methods in applied mechanics and engineering*, vol. 113, no. 3-4, pp. 397–414, 1994.

[58] M. Mofarreh-Bonab and M. Mofarreh-Bonab, "A new technique for image compression using pca," *International Journal of Computer Science & Communication Networks*, vol. 2, no. 1, pp. 111–116, 2012.

[59] P. Dash, M. Nayak, and G. P. Das, "Principal component analysis using singular value decomposition for image compression," *International Journal of Computer Applications*, vol. 93, no. 9, 2014.

[60] J. K. Percus and G. J. Yevick, "Analysis of classical statistical mechanics by means of collective coordinates," *Physical Review*, vol. 110, no. 1, p. 1, 1958.

[61] I. H. Tarman, "A karhunen–loeve analysis of turbulent thermal convection," *International journal for numerical methods in fluids*, vol. 22, no. 1, pp. 67–79, 1996.

[62] I. Tumer, K. Wood, and I. Busch-Vishniac, "Monitoring of signals from manufacturing processes using the karhunen–loeve transform," *Mechanical Systems and Signal Processing*, vol. 14, no. 6, pp. 1011–1026, 2000.

[63] A. J. Newman, "Model reduction via the karhunen-loeve expansion part ii: Some elementary examples," Tech. Rep., 1996.

[64] J. Baker and P. D. Christofides, "Finite-dimensional approximation and control of non-linear parabolic pde systems," *International Journal of Control*, vol. 73, no. 5, pp. 439–456, 2000.

[65] S. Amaral, D. Allaire, and K. Willcox, "A decomposition-based approach to uncertainty analysis of feed-forward multicomponent systems," *International Journal for Numerical Methods in Engineering*, vol. 100, no. 13, pp. 982–1005, 2014.

[66] D. Shi and F. Tsung, "Modelling and diagnosis of feedback-controlled processes using dynamic pca and neural networks," *International Journal of Production Research*, vol. 41, no. 2, pp. 365–379, 2003.

[67] M. Kirby and L. Sirovich, "Application of the karhunen-loeve procedure for the characterization of human faces," *IEEE Transactions on Pattern analysis and Machine intelligence*, vol. 12, no. 1, pp. 103–108, 1990.

[68] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz, "Efficient and effective querying by image content," *Journal of intelligent information systems*, vol. 3, no. 3, pp. 231–262, 1994.

[69] A. K. Jain, "A fast karhunen-loeve transform for a class of random processes," *IEEE Transactions on Communications*, vol. 24, no. 9, pp. 1023–1029, 1976.

[70] S. Olmos, M. MillAn, J. Garcia, and P. Laguna, "Ecg data compression with the karhunen-loeve transform," in *Computers in Cardiology 1996*. IEEE, 1996, pp. 253–256.

[71] H. Abbas and M. Fahmy, "Neural model for karhunen-loeve transform with application to adaptive image compression," *IEE Proceedings I (Communications, Speech and Vision)*, vol. 140, no. 2, pp. 135–143, 1993.

[72] J.-M. Chiou, Y.-C. Zhang, W.-H. Chen, and C.-W. Chang, "A functional data approach to missing value imputation and outlier detection for traffic flow data," *Transportmetrica B: Transport Dynamics*, vol. 2, no. 2, pp. 106–129, 2014.

[73] D. Brauckhoff, K. Salamatian, and M. May, "Applying pca for traffic anomaly detection: Problems and solutions," in *IEEE INFOCOM 2009*, 2009, pp. 2866–2870.

[74] H. A. Almurib, T. N. Kumar, and F. Lombardi, "Approximate dct image compression using inexact computing," *IEEE Transactions on Computers*, vol. 67, no. 2, pp. 149–159, 2018.

[75] W. Xiao, N. Wan, A. Hong, and X. Chen, "A fast jpeg image compression algorithm based on dct," in *2020 IEEE International Conference on Smart Cloud (SmartCloud)*, 2020, pp. 106–110.

[76] S. L. Agrwal, M. Sharma, D. Kumari, and S. K. Gupta, "Improved image compression technique using iwt-dct transformation," in *2016 2nd International Conference on Next Generation Computing Technologies (NGCT)*, 2016, pp. 683–686.

[77] F. Bayer and R. Cintra, "DCT-like transform for image compression requires 14 additions only," *Electronics Letters*, vol. 48, no. 15, p. 919, 2012. [Online]. Available: https://doi.org/10.1049%2Fel.2012.1148

[78] D. Liu, H. Ma, Z. Xiong, and F. Wu, "Cnn-based dct-like transform for image compression," in *International Conference on Multimedia Modeling*. Springer, 2018, pp. 61–72.

[79] N. Ponomarenko, V. Lukin, K. Egiazarian, and J. Astola, "Dct based high quality image compression," in *Scandinavian Conference on Image Analysis*. Springer, 2005, pp. 1177–1185.

[80] A. J. Ahumada Jr and H. A. Peterson, "Luminance-model-based dct quantization for color image compression," in *Human vision, visual processing, and digital display III*, vol. 1666. SPIE, 1992, pp. 365–374.

[81] R. T. Nageswara and K. D. Srinivasa, "Image compression using discrete cosine transform," *Computer Sciences and Telecommunications*, no. 3, pp. 35–44, 2008.

[82] D. Chikouche, R. Benzid, and M. Bentoumi, "Application of the dct and arithmetic coding to medical image compression," in *2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications*. IEEE, 2008, pp. 1–5.

[83] A. B. Watson *et al.*, "Image compression using the discrete cosine transform," *Mathematica journal*, vol. 4, no. 1, p. 81, 1994.

[84] N.-U. Kim, S.-C. Lim, J. Kang, H. Y. Kim, and Y.-L. Lee, "Transform with residual rearrangement for hevc intra coding," *Signal Processing: Image Communication*, vol. 78, pp. 322–330, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0923596519300591

[85] A. Saxena and F. C. Fernandes, "Mode dependent dct/dst for intra prediction in block-based image/video coding," in *2011 18th IEEE International Conference on Image Processing*. IEEE, 2011, pp. 1685–1688.

[86] V. Britanak and R. Rao, "Two-dimensional dct/dst universal computational structure for 2 × 2 block sizes," *Signal Processing, IEEE Transactions on*, vol. 48, pp. 3250 – 3255, 12 2000.

[87] A. Gnutti, F. Guerrini, R. Leonardi, and A. Ortega, "Coding of image intra prediction residuals using symmetric graphs," in *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019, pp. 131–135.

[88] C. Zhang and D. Florêncio, "Analyzing the optimality of predictive transform coding using graph-based models," *IEEE Signal Processing Letters*, vol. 20, no. 1, pp. 106–109, 2012.

[89] Y. Xu, W. Hu, S. Wang, X. Zhang, S. Wang, S. Ma, Z. Guo, and W. Gao, "Predictive generalized graph fourier transform for attribute compression of dynamic point clouds," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 5, pp. 1968–1982, 2020.

[90] Y.-H. Chao, A. Ortega, and S. Yea, "Graph-based lifting transform for intra-predicted video coding," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 1140–1144.

[91] 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.

[92] K.-S. Lu and A. Ortega, "Symmetric line graph transforms for inter predictive video coding," in *2016 Picture Coding Symposium (PCS)*. IEEE, 2016, pp. 1–5.

[93] S. Bagheri, T. T. Do, G. Cheung, and A. Ortega, "Hybrid model-based / data-driven graph transform for image coding," in *2022 IEEE International Conference on Image Processing (ICIP)*, 2022, pp. 3667–3671.

[94] X. Cai and J. S. Lim, "Transforms for intra prediction residuals based on prediction inaccuracy modeling," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5505–5515, 2015.

[95] T. K. Tan, C. S. Boon, and Y. Suzuki, "Intra prediction by template matching," in *2006 International Conference on Image Processing*. IEEE, 2006, pp. 1693–1696.

[96] C. Lan, J. Xu, F. Wu, and G. Shi, "Intra frame coding with template matching prediction and adaptive transform," in *2010 IEEE International Conference on Image Processing*. IEEE, 2010, pp. 1221–1224.

[97] Y. Suzuki, C. S. Boon, and T. K. Tan, "Video encoding scheme employing intra and inter prediction based on averaged template matching

predictors," *IEICE transactions on information and systems*, vol. 91, no. 4, pp. 1127–1134, 2008.

[98] 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.

[99] 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)*. IEEE, 2015, pp. 199–203.

[100] Y.-H. Chao, H. E. Egilmez, A. Ortega, S. Yea, and B. Lee, "Edge adaptive graph-based transforms: Comparison of step/ramp edge models for video compression," in *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 1539–1543.

[101] M. Rizkallah, X. Su, T. Maugey, and C. Guillemot, "Graph-based transforms for predictive light field compression based on super-pixels," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 1718–1722.

[102] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning laplacian matrix in smooth graph signal representations," *IEEE Transactions on Signal Processing*, vol. 64, no. 23, pp. 6160–6173, 2016.

[103] D. Recoskie and R. Mann, "Learning filters for the 2d wavelet transform," in *2018 15th Conference on Computer and Robot Vision (CRV)*. IEEE, 2018, pp. 198–205.

[104] ——, "Learning sparse wavelet representations," *arXiv preprint arXiv:1802.02961*, 2018.

[105] G. Michau, G. Frusque, and O. Fink, "Fully learnable deep wavelet transform for unsupervised monitoring of high-frequency time series," *Proceedings of the National Academy of Sciences*, vol. 119, no. 8, p. e2106598119, 2022.

[106] A. Arneodo, Y. d'Aubenton Carafa, B. Audit, E. Bacry, J.-F. Muzy, and C. Thermes, "What can we learn with wavelets about dna sequences?" *Physica A: Statistical Mechanics and its Applications*, vol. 249, no. 1-4, pp. 439–448, 1998.

[107] G. Quellec, M. Lamard, P. M. Josselin, G. Cazuguel, B. Cochener, and C. Roux, "Optimal wavelet transform for the detection of microaneurysms in retina photographs," *IEEE transactions on medical imaging*, vol. 27, no. 9, pp. 1230–1241, 2008.

[108] M. Sun, X. He, S. Xiong, C. Ren, and X. Li, "Reduction of jpeg compression artifacts based on dct coefficients prediction," *Neurocomputing*, vol. 384, pp. 335–345, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231219317175

[109] L. Gueguen, A. Sergeev, B. Kadlec, R. Liu, and J. Yosinski, "Faster neural networks straight from jpeg," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018. [Online]. Available: https://proceedings.neurips.cc/paper/2018/file/7af6266cc52234b5aa339b16695f7fc4-Paper.pdf

[110] J. Jiang, "Image compression with neural networks – a survey," *Signal Processing: Image Communication*, vol. 14, no. 9, pp. 737 – 760, 1999. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0923596598000411

[111] I. Garg, S. S. Chowdhury, and K. Roy, "Dct-snn: Using dct to distribute spatial information over time for learning low-latency spiking neural networks," *arXiv preprint arXiv:2010.01795*, 2020.

[112] M.-J. Kwon, I.-J. Yu, S.-H. Nam, and H.-K. Lee, "Cat-net: Compression artifact tracing network for detection and localization of image splicing," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, January 2021, pp. 375–384.

[113] A. Arrufat, P. Philippe, and O. Déforges, "Non-separable mode dependent transforms for intra coding in hevc," in *2014 IEEE Visual Communications and Image Processing Conference*, 2014, pp. 61–64.

[114] K. Fan, R. Wang, W. Lin, L.-Y. Duan, and W. Gao, "Signal-independent separable klt by offline training for video coding," *IEEE Access*, vol. 7, pp. 33 087–33 093, 2019.

[115] E. Pavez, A. Ortega, and D. Mukherjee, "Learning separable transforms by inverse covariance estimation," in *2017 IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 285–289.

[116] H. Abbas and M. Fahmy, "A neural model for adaptive karhunen loeve transformation (klt)," in *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks*, vol. 2, 1992, pp. 975–980 vol.2.

[117] A. Madabhushi, "Digital pathology image analysis: opportunities and challenges," *Imaging in medicine*, vol. 1, no. 1, p. 7, 2009.

[118] M. May, "A better lens on disease," *Scientific American*, vol. 302, no. 5, pp. 74–77, 2010.

[119] V. Sanchez, M. Hernandez-Cabronero, F. Auli-Llinàs, and J. Serra-Sagristà, "Fast lossless compression of whole slide pathology images using hevc intra-prediction," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 1456–1460.

[120] V. Sanchez, F. Aulí-Llinàs, J. Bartrina-Rapesta, and J. Serra-Sagristà, "Hevc-based lossless compression of whole slide pathology images," in *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE, 2014, pp. 297–301.

[121] V. Sanchez, F. Auli-Llinas, R. Vanam, and J. Bartrina-Rapesta, "Rate control for lossless region of interest coding in hevc intra-coding with applications to digital pathology images," in *International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2015, pp. 1250–1254.

[122] M. Hernández-Cabronero, V. Sanchez, F. Auli-Llinas, and J. Serra-Sagrista, "Fast mct optimization for the compression of whole-slide images," in *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2016, pp. 2370–2374.

[123] M. Hernández-Cabronero, F. Auli-Llinas, V. Sanchez, and J. S. Sagrista, "Transform optimization for the lossy coding of pathology whole-slide images," in *2016 Data Compression Conference (DCC)*. IEEE, 2016, pp. 131–140.

[124] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The jpeg 2000 still image compression standard," *IEEE Signal processing magazine*, vol. 18, no. 5, pp. 36–58, 2001.

[125] 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.

[126] X. Cai *et al.*, "Transforms for prediction residuals based on prediction inaccuracy modeling," Ph.D. dissertation, Massachusetts Institute of Technology, 2017.

[127] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," *IEEE Transactions on circuits and systems for video technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

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

[129] V. Sanchez, F. Aulí-Llinàs, J. Bartrina-Rapesta, and J. Serra-Sagristà, "Hevc-based lossless compression of whole slide pathology images," in *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2014, pp. 297–301.

[130] E. Pavez, A. Ortega, and D. Mukherjee, "Learning separable transforms by inverse covariance estimation," in *Int. Conf. Image Processing*, 2017, pp. 285–289.

[131] V. Sanchez, F. Auli-Llinas, and J. Serra-Sagristà, "Piecewise mapping in hevc lossless intra-prediction coding," *IEEE Transactions on Image Processing*, vol. 25, no. 9, pp. 4004–4017, 2016.

[132] C. Lan, J. Xu, W. Zeng, G. Shi, and F. Wu, "Variable block-sized signal-dependent transform for video coding," *IEEE Trans. Circuits and Systems for Video Tech*, vol. 28, no. 8, pp. 1920–1933, 2018.

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

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

[135] Y. Li and Z. Zhang, "Digraph laplacian and the degree of asymmetry," *Internet Mathematics*, vol. 8, no. 4, pp. 381–401, 2012. [Online]. Available: https://doi.org/10.1080/15427951.2012.708890

[136] H. E. Egilmez, Y. H. Chao, and A. Ortega, "Graph-based transforms for video coding," *IEEE Transactions on Image Processing*, vol. 29, p. 9330–9344, 2020. [Online]. Available: http://dx.doi.org/10.1109/TIP.2020.3026627

[137] J. Han, A. Saxena, V. Melkote, and K. Rose, "Jointly optimized spatial prediction and block transform for video and image coding," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1874–1884, 2012.

[138] SO/IEC-JCT1/SC29/WG11, Common Test Conditions and Software Reference Configurations, Geneva, Switzerland, 2012.

[139] C. Thum, "Measurement of the entropy of an image with application to image focusing," *Optica Acta: International Journal of Optics*, vol. 31, no. 2, pp. 203–211, 1984. [Online]. Available: https://doi.org/10.1080/713821475