Learned-based Intra Coding Tools for Video Compression

Maria Claudia Santamaria Gomez

Submitted in partial fulfillment of the requirements of the Degree of Doctor of Philosophy

School of Electronic Engineering and Computer Science
Queen Mary University of London
United Kingdom

August 2021

Statement of originality

I, Maria Claudia Santamaria Gomez, confirm that the research included within this thesis is my own work or that where it has been carried out in collaboration with, or supported by others, that this is duly acknowledged below and my contribution indicated. Previously published material is also acknowledged below.

I attest that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other Intellectual Property Right, or contain any confidential material.

I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university.

Maria Claudia Santamaria Gomez
Tuesday 10th August, 2021

Details of collaborations and publications:

All papers published while working on this thesis are mentioned in Chapter 1. Any publication produced in collaboration with others is clearly mentioned.

A mi familia

Abstract

The increase in demand for video rendering in 4K and beyond displays, as well as immersive video formats, requires the use of efficient compression techniques. In this thesis novel methods for enhancing the efficiency of current and next generation video codecs are investigated. Several aspects that influence the way conventional video coding methods work are considered. The methods proposed in this thesis utilise Neural Networks (NNs) trained for regression tasks in order to predict data. In particular, Convolutional Neural Networks (CNNs) are used to predict Rate-Distortion (RD) data for intra-coded frames. Moreover, a novel intra-prediction methods are proposed with the aim of providing new ways to exploit redundancies overlooked by traditional intra-prediction tools. Additionally, it is shown how such methods can be simplified in order to derive less resource-demanding tools.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisors, Prof. Ebroul Izquierdo and Prof. Marta Mrak. I am very grateful to Ebroul, who offered me a position in his group to pursue a PhD; his guidance, suggestions and encouragement have been essential to the completion of this thesis. I would also like to thank Marta, who gave me the opportunity to be part of her team; for her constant support, insightful criticism and the lessons she has taught me along the way. My thanks also go to Dr. Qianni Zhang for her time and feedback, and to Dr. Maria Trujillo for her constant encouragement and support.

I gratefully acknowledge the funding for my research received from the Engineering and Physical Science Research Council (EPSRC) and the British Broadcasting Corporation (BBC), through an Industrial Cooperative Awards in Science & Technology (iCASE) studentship.

I would like to offer my sincerest gratitude to Dr. Saverio Blasi, who has worked closely with me on some topics of this thesis; for his patience and help. I would like to thank Dr. Krishna Chandramouli for all his reliable advice. I am also deeply indebted to all members of the Multimedia and Vision (MMV) research group at Queen Mary University of London, as well as the Visual Data Analytics team at BBC Research and Development. From who I received timely support throughout my studies. In addition, an incredible journey with lots of learning and fun.

Last but not least, I am extremely thankful to mum, dad and Alejandro for their continuous support through this period. I love you all tremendously. No amount of words will be ever enough to express my gratitude.

Contents

List of figures

List of tables

List of acronyms

AI	All Intra
AOMedia	Alliance for Open Media
AVC	Advanced Video Coding
AV1	AOmedia Video 1
bpp	bits per pixel
BD	Bjøntegaard Delta
CABAC	Context-Adaptive Binary Arithmetic Coding
CB	Coding Block
CNN	Convolutional Neural Network
CTB	Coding Tree Block
CTC	Common Test Conditions
CTU	Coding Tree Unit
CU	Coding Unit
DCT	Discrete Cosine Transform
DCTIF	Discrete Cosine Transform based Interpolation Filter
DIV2K	DIVerse 2K
DST	Discrete Sine Transform
ELU	Exponential Linear Unit
FC	Fully Connected
FCN	Fully Connected Neural Network
FHD	Full High Definition
FPS	Frames Per Second

GAN	Generative Adversarial Network
HD	High Definition
HDR	High Dynamic Range
HEVC	High Efficiency Video Coding
HFR	High Frame Rate
HM	HEVC test Model
HR	High Resolution
HVS	Human Visual System
IEC	International Electrotechnical Commission
IPFCN	Intra Prediction Fully Connected Network
ISO	International Organization for Standardization
ITS	Institute for Telecommunication Sciences
ITU-T	International Telecommunication Union - Telecommunication
JCT-VC	Joint Collaborative Team on Video Coding
JVET	Joint Video Experts Team
LD	Low Delay
LR	Low Resolution
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MC	Motion Compensation
ME	Motion Estimation
MIP	Matrix-based Intra-Prediction
MPEG	Moving Picture Experts Group
MPM	Most Probable Mode
MRLP	Multi-Reference Line Prediction
MSE	Mean Squared Error
M&E	Media and Entertainment
MV	Motion Vector
NN	Neural Network
PB	Prediction Block
PCC	Pearson Correlation Coefficient

PredNet	Predictive coding Network
PReLU	Parametric Rectified Linear Unit
PSNR	Peak Signal-to-Noise Ratio
PU	Prediction Unit
QP	Quantisation Parameter
QT+MTT	Quad-Tree plus Multi-Type Tree
RA	Random Access
RAP	Random Access Point
RD	Rate-Distortion
ReLU	Rectified Linear Unit
RDO	Rate-Distortion Optimisation
RNN	Recurrent Neural Network
SAD	Sum of Absolute Differences
SATD	Sum of Absolute Transformed Differences
SMPTE	Society of Motion Picture and Television Engineers
SRCNN	Super Resolution Convolutional Neural Network
SSD	Sum of Squared Differences
SSIM	Structural SIMilarity
TB	Transform Block
TU	Transform Unit
UHD	Ultra High Definition
VCEG	Video Coding Experts Group
VDSR	Very Deep Super Resolution
VMAF	Video Multi-method Assessment Fusion
VoD	Video-on-Demand
VQM	Video Quality Metric
VTM	VVC Test Model
VVC	Versatile Video Coding
WCG	Wide Colour Gamut

Chapter 1

Introduction

The Media and Entertainment (M&E) industry comprises the production and distribution of media (e.g. films, TV, music, radio, advertising, video games and streaming content). The M&E market has shown to evolve with the audience consumption, which is driven by demand of high-quality content. This means the M&E industry must constantly innovate to keep the user experience at its highest. In this regard, M&E companies are actively adapting and providing disruptive services, more attractive to consumers, through the exploration of emerging technologies. For instance, data from digital transactions, such as online purchases and user preferences, are being exploited as contextual information to improve recommendation systems [39].

The M&E consumption panorama is rapidly growing. It is estimated that by 2022, global Internet traffic will reach 4.8 zettabytes and 82% will come from video distribution, of which 22% will be high-quality content [22]. Such rise in video traffic is a result of breakthroughs in screening and video recording technologies, as well as the evolution of smart devices. In addition, video traffic is being accelerated by the COVID-19 pandemic. Social distancing and self-isolation policies have made remote work the norm. Consequently, millions of people around the globe are working from home [135]. Conferences are now 100% virtual and in-home entertainment is the primary source of leisure, as shown by the boost in hours watched on Twitch since March 2020 [140].

The M&E industry is also being disrupted by over-the-top media services, such as Video-on-Demand (VoD), live and game streaming, broadcast TV and online videos, which consumers keep demanding at the highest possible quality.

High-quality video comes in the form of Ultra High Definition (UHD) that provides higher pixel count and higher level of detail; High Frame Rate (HFR) that gives smoother motion and clarity; High Dynamic Range (HDR) that allows for higher contrast and darker, brighter and more vivid content; Wide Colour Gamut (WCG) that brings a larger colour palette to represent content; and high bit depth [129]. These formats add more definition to video for a more immersive experience. Nevertheless, they require not only a large number of bits to represent video, but also high-speed bandwidth for playback. For example, transmitting a 10 bit raw (uncompressed) UHD 4K (3840 × 2160) video at 60 frames per second requires 7.5 Gbit/s [129], whereas the average broadband speed in the United Kingdom in 2020 is 71.8 Mbit/s [108].

Since storage and bandwidth capabilities are limited, video needs to be compressed. Video compression is tackled by a branch of signal processing known as video coding, which exploits statistical and subjective redundancy within a frame (intra coding) and between frames (inter coding) in order to reduce the bit rate of a video [121]. The goal of video coding is to provide high compression efficiency while minimising compression artefacts, finding a trade-off between bandwidth requirements and video quality.

Compressed video is displayed in a large range of devices and digital platforms, and the interoperability among those is enabled by international video coding standards. International standardisation is a joint effort between countries and companies that seek a compromise between compression efficiency, computational complexity and flexibility supported by the standard [121]. A video coding standard defines the syntax of the compressed video (bitstream) and the decoding process to reconstruct the signal for visualisation. Finally, standard-compliant video can be successfully exchanged and displayed across a wide variety of devices and platforms since video decoders are implemented by hardware and software manufacturers.

The first practical video coding standard was H.261 [55], developed by the International Telecommunication Union - Telecommunication (ITU-T) sector in 1988. H.261 was the first block-based hybrid video coding scheme, which is the basis for most modern video coding standards. In 1993, the Moving Picture Experts Group (MPEG) released MPEG-1 [102]. ITU-T and MPEG collaborated together and released H.262/MPEG-2 [139] in 1995. The same partnership released the most successful video coding standard, H.264/MPEG-4 Advanced Video Coding (AVC) [150] in 2003; H.265/High Efficiency Video Coding (HEVC) [130] in 2013; and H.266/Versatile Video Coding (VVC) [56] in 2020.

Moreover, other companies have been developing their own video codecs in order to cut expenses on royalty-bearing standards [97]. In 2015, Amazon, Cisco, Google, Netflix, among others, formed the Alliance for Open Media (AOMedia) to pursue open and royalty-free video technology. Their first video coding standard is known as AOmedia Video 1 (AV1) [19], it was finalised in 2018 and is being used by YouTube [153] and Netflix [42].

Modern video coding standards include coding configurations (suitable to different applications) that specify how a video is to be encoded. For instance, video distribution relies on Random Access (RA) configurations, in which most frames are inter-coded and a few intra-coded frames are inserted periodically in the sequence. Intra-coded frames are used as Random Access Points (RAPs), where the decoding process may start [44]. Additionally, as intra-coded frames are typically referenced by inter-coded frames, they should be encoded at the highest quality.

The increasing demand for high-quality video has accelerated the development and adoption of new video formats, resulting in a high amount of data that needs to be compressed for efficient storage and distribution. Hence, video coding techniques must evolve in order to provide higher compression rate and visual quality.

1.1 Problem statement

The coding of emerging video formats requires more efficient video coding methods than conventional ones.

1.2 Objectives

In recent years the availability of hardware capable of deploying Neural Networks (NNs) has increased considerably, reducing the barrier to entry of applied NNs. In that regard, the main goal of this thesis is to leverage NNs and exploit their potential in hybrid video coding schemes. More specifically, Convolutional Neural Networks (CNNs) and Fully Connected Neural Networks (FCNs) are studied to develop new intra coding tools.

This research has the following set of specific objectives:

- to predict the quality of intra-coded frames, without multi-pass encoding;
- to predict the bits required to represent intra-coded frames, without multi-pass encoding;
- to evaluate the effectiveness of the Rate-Distortion (RD) predictions to estimate the RD model;
- to simplify FCN-based intra-prediction; and
- to design an intra-prediction technique with learnable resolution adaptation.

1.3 Contributions

As a whole, the work presented in this thesis leverages NNs and explores its usage in hybrid video coding schemes. All techniques in this work are focused on intra coding and target prediction tasks. The main contributions of the work presented herein are:

It is generally difficult to predict the effects that a video encoder has on a frame, in terms of number of bits and distortion, prior the actual encoding process. **Chapter 3** describes a CNNs-based framework that estimates RD metrics associated to intra coding. Since the predictions are computed from the uncompressed content (original frames), multi-pass encoding can be omitted.

Most NNs are black boxes, and it is nearly difficult to figure out how the model variables are combined to produce predictions. **Chapter 4** describes techniques to simplify an FCN for intra-prediction in such a manner that the resultant model is clearly understandable, less complex and provides similar compression efficiency.

While linear FCNs can achieve similar compression performance as FCNs with non-linearities, the number of parameters of the FCN rises as the block size grows. **Chapter 5** provides a proof of concept that targets the reduction of parameters by means of using an end-to-end intra-prediction method with resolution adaptation.

1.4 Publications

The author has published peer-reviewed papers in international conferences during the course of her PhD research, some of which are directly related to the topics discussed in this thesis.

- L. Murn, M. Gorriz Blanch, **M. Santamaria**, F. Rivera and M. Mrak, "Towards transparent application of machine learning in video processing," *International Broadcasting Convention (IBC)*, 2020.
- **M. Santamaria**, S. Blasi, E. Izquierdo and M. Mrak, "Functional interpretation of fully connected neural network for intra-prediction," *Women in Computer Vision workshop at European Conference on Computer Vision (WiCV @ ECCV)*, 2020.

- **M. Santamaria**, S. Blasi, E. Izquierdo and M. Mrak, "Analytic simplification of neural network based intra-prediction modes for video compression," *IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, 2020.
- **M. Santamaria**, S. Blasi, M. Mrak and E. Izquierdo, "Estimation of rate control parameters for video coding using CNN," *IEEE International Conference on Visual Communications and Image Processing (VCIP)*, 2018.

1.5 Thesis outline

The rest of this thesis is organised as follows. **Chapter 2** provides a general background on the research field of this thesis. The chapter begins with a brief description of video coding fundamentals. It explains the hybrid block-based video coding scheme, the HEVC and the VVC standards. It also provides a literature review on NN-based video coding tools, which is summarised at the end of the chapter.

Chapter 3 focuses on the estimation of both rate and distortion metrics, resulting from intra coding, with CNNs. The predictions are computed from original data. Hence, it is possible to avoid the usage of multi-pass encoding.

Chapter 4 introduces two simplification methods for FCNs used as intra-prediction modes. The proposed modes aim to provide coding gains and reduce the computational cost of deploying the FCN.

Chapter 5 presents the super pixel intra-prediction technique, which includes resolution adaptation. The proposed technique learns the down-sampling of reference samples, refer to as super pixels, the computation of the prediction in Low Resolution (LR) and the up-sampling operation to obtain the prediction in High Resolution (HR).

Chapter 6 summarises the conclusions derived from this work, as well as ideas for improving and developing further the proposed contributions.

Chapter 2

Background

This chapter presents video coding concepts as well as the block-based hybrid video coding scheme. Both the High Efficiency Video Coding standard and the Versatile Video Coding standard are described, followed by an introduction to NNs. Finally, the state-of-the-art in NN-based video coding tools is presented.

2.1 Video coding preliminaries

This section explains the video coding concepts essential to the thesis.

2.1.1 Digital video representation

A digital video consists of a sequence of images, from now on referred to as frames, depicting a scene at different time instants. The speed at which the frames of a video are displayed is known as frame rate, and it is usually expressed as Frames Per Second (FPS) [95]. The frame rate influences the quality of the video and determines how realistic it looks like. Instead of abrupt changes from one frame to another, the Human Visual System (HVS) perceives a sense of motion due to small changes between consecutive frames and a high frame rate [71].

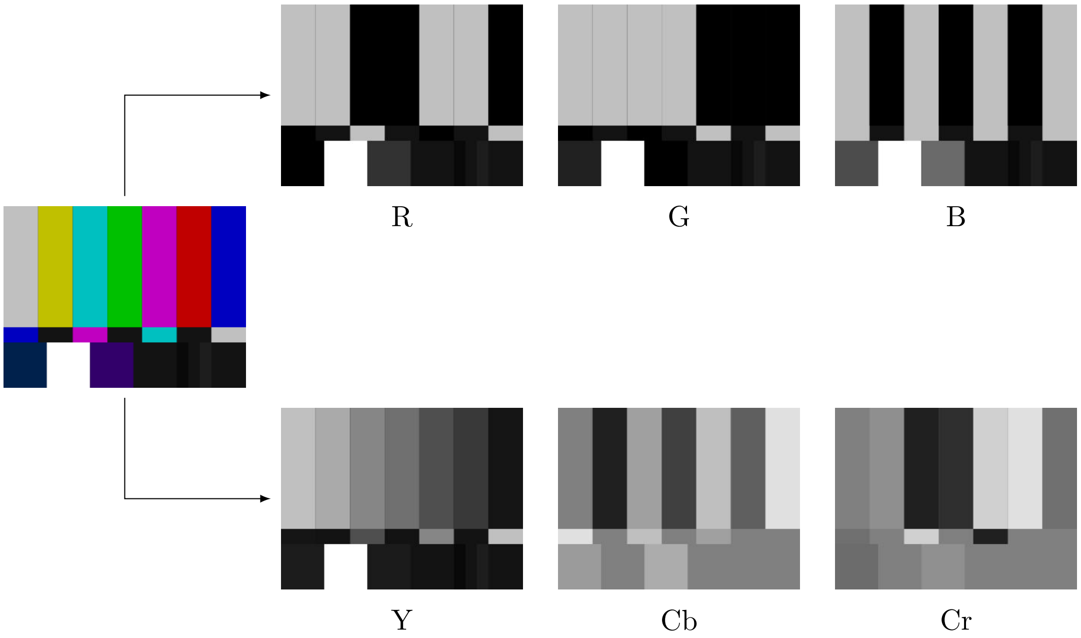
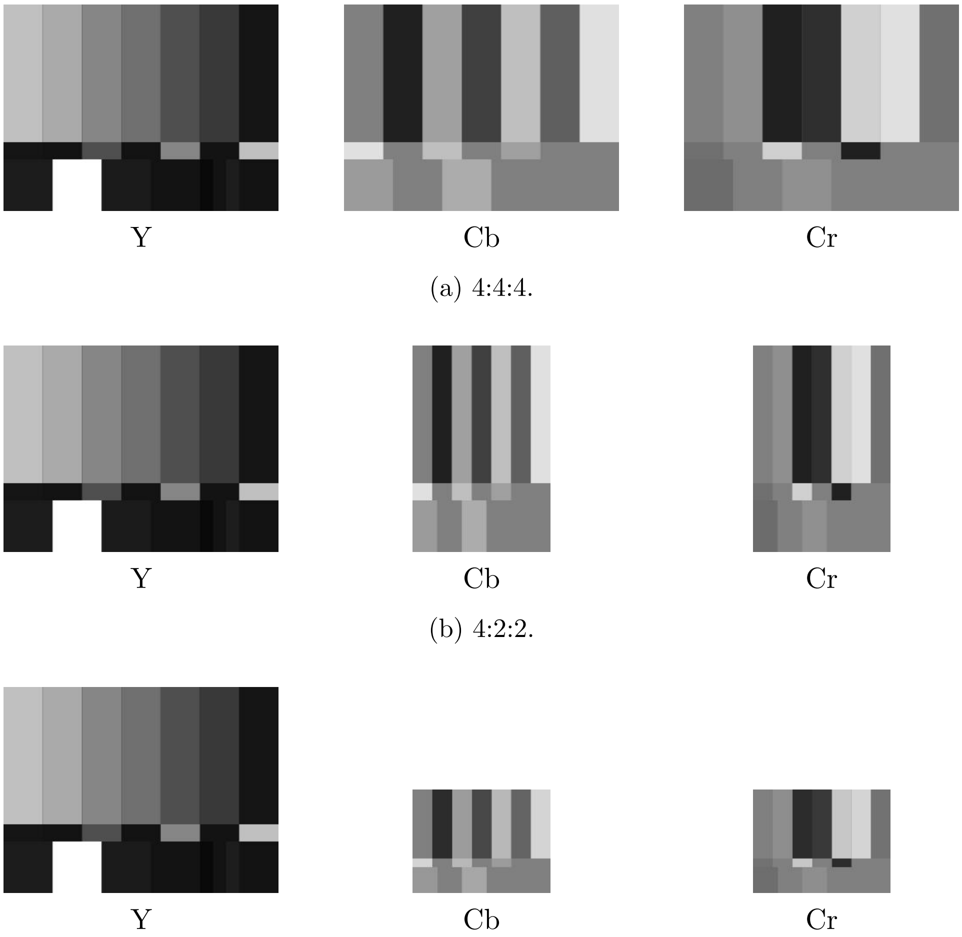


Figure 2.1: RGB and YCbCr colour spaces of SMPTE colour bars [27].

Each frame on a video is made of a set of samples, which are displayed in a 2D space following the spatial resolution of the video. A greyscale frame is represented by a single array of samples, where each sample represents the intensity value of the frame at a specific location. In addition, each intensity value is mapped to a digital representation within the range $[0, 2^B - 1]$, parametrised by the bit depth B of the video. Colour content is also mapped to a digital representation. For instance the RGB colour space, widely used for displaying colour images, expresses colour as the combination of red, green and blue lights. The digital representation consists of three arrays of samples, one per component, all of which capture both brightness (luma) and colour (chroma). Analogue video uses YUV colour space, which consists of a luma component (Y) and two chroma components (U and V). For digital video, YCbCr colour space is preferred. Y is the luma component and Cb and Cr are the blue-difference and red-difference chroma components [113]. As this thesis focuses coding techniques of digital videos, YCbCr is the colour representation assumed for videos throughout the rest of this document. Figure 2.1 shows the RGB and YCbCr colour representations for the Society of Motion Picture and Television Engineers (SMPTE) colour bars.



(a) 4:4:4.

(b) 4:2:2.

(c) 4:2:0.

Figure 2.2: Chroma subsampling applied to SMPTE colour bars.

As mentioned before, the HVS is less sensitive to chroma differences than to luma differences [52]. In this way video representation can be improved by reducing the spatial resolution of colour components, also known as chroma subsampling [152]. The most common formats are 4:4:4, meaning no chroma sub-sampling is applied; 4:2:2, meaning chroma components are sub-sampled by a factor of 2 horizontally; and 4:2:0, meaning chroma components are sub-sampled by a factor of 2 both horizontally and vertically. Figure 2.2 depicts how the SMPTE colour bars look like at different chroma subsampling formats.

2.1.2 Video coding

The perceived quality of a video is not only determined by its frame rate but also its spatial resolution. The latest video formats include High Definition (HD) (1280 × 720), Full High Definition (FHD) (1920 × 1020) and UHD, 4K (3840 × 2160) and 8K (7680 × 4320). As the frame rate and resolution increases, so does the number of bits required to provide a direct representation. Such large files demand higher bandwidths and storage capabilities, making transfer via email and share content over social media more difficult. Hence, video compression is essential to reduce the bandwidth usage and storage space associated to digital videos [161].

In a typical encoder-decoder framework, illustrated in Figure 2.3, an input video is encoded to produce a bitstream, which represents the video content using fewer bits without excessively reducing its quality. The bitstream (compressed video) is then stored and later transmitted. Once the compressed video is delivered, it needs to be decoded for viewing. Since compressed videos need to be displayed on a variety of devices, made by different manufactures, consistency is a necessity. Video coding standards provide bitstream reliability with two notions. Firstly, by defining a common bitstream syntax. Secondly, by introducing the decoding process required to generate the reconstructed video. A standard specifies only the decoding as the encoding is usually designed to meet certain usage needs.

Video compression reduces temporal, spatial and statistical redundancies; and it can be lossy or lossless. With lossless compression, the video is stored in an efficient way without discarding data, meaning the compressed signal can be restored to its original form. Contrarily, lossy compression discards irrelevant parts of the signal, as typically the input video contains more data than the HVS can perceive [124, 126]. The degradation of the quality in the reconstructed signal is known as distortion, and it is product of the compression technique applied. Several distortion metrics have been defined, and the most used of those are Sum of Squared Differences (SSD), Sum of Absolute Differences (SAD), Mean Squared Error (MSE) and Sum of Absolute Transformed Differences (SATD).

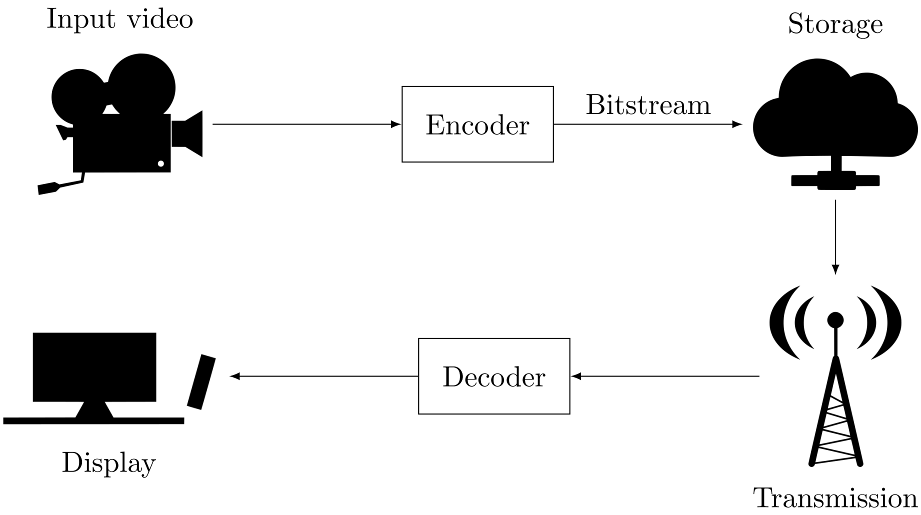


Figure 2.3: Video coding framework.

Given a raw frame $\mathbf{Y} \in \mathbb{R}^{H \times W}$ and its reconstruction $\hat{\mathbf{Y}} \in \mathbb{R}^{H \times W}$, the SSD and the MSE are defined as

$$\text{SSD} = \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} [y_{i,j} - \hat{y}_{i,j}]^2, \tag{2.1}$$

$$\text{MSE} = \frac{\text{SSD}}{W \cdot H}. \tag{2.2}$$

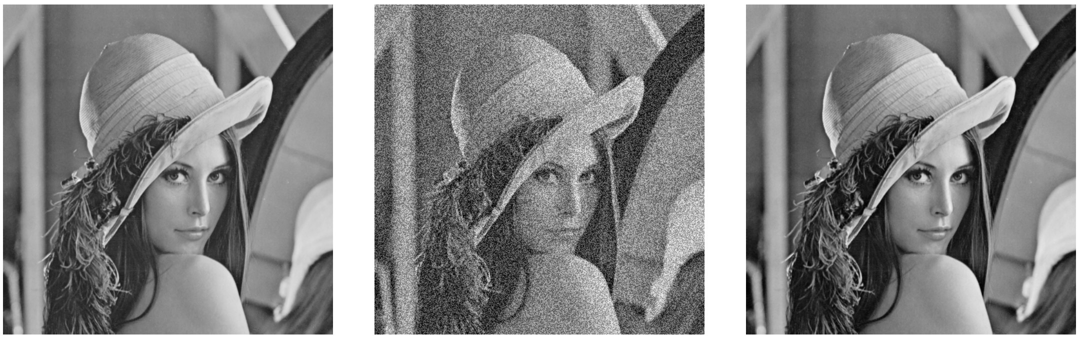
The SAD compares the signals in the spatial domain and is given by

$$\text{SAD} = \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} |y_{i,j} - \hat{y}_{i,j}|. \tag{2.3}$$

In contrast, the SATD applies a Hadamard transform $\mathbf{T}$ before computing the difference in order to estimate the effort needed to code the residual [151]:

$$\text{SATD} = \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} |f_{i,j}|, \tag{2.4}$$

$$\mathbf{F} = \mathbf{T} \times (\mathbf{Y} - \hat{\mathbf{Y}}) \times \mathbf{T}^{\top}. \tag{2.5}$$



(a) Original Y.

(b) Y + noise.
PSNR 13.979. SSIM 0.125.

(c) Y + constant.
PSNR 13.979. SSIM 0.923.

Figure 2.4: Comparison of PSNR and SSIM index.

SSD, MSE, SAD and SATD all have high values that indicate high numerical differences between the signals being compared. Let B be the bit depth of the signals, the quality of a reconstruction is measured using the Peak Signal-to-Noise Ratio (PSNR):

$$\mathrm{PSNR} = 10 \cdot \log_{10}\left(\frac{(2^B - 1)^2}{\mathrm{MSE}}\right). \tag{2.6}$$

The PSNR and other element-wise difference metrics fail to predict human perceived quality [49]. The Structural SIMilarity (SSIM) index [147] takes into account factors of the HVS and compares the luminance, contrast and structure of the signals based on mean intensity μ and standard deviation σ:

$$\mathrm{SSIM} = \frac{(2\mu_{\mathbf{Y}}\mu_{\hat{\mathbf{Y}}} + C_1)(2\sigma_{\mathbf{Y}\hat{\mathbf{Y}}} + C_2)}{(\mu_{\mathbf{Y}}^2 + \mu_{\hat{\mathbf{Y}}}^2 + C_1)(\sigma_{\mathbf{Y}}^2 + \sigma_{\hat{\mathbf{Y}}}^2 + C_2)}, \tag{2.7}$$

where C_1 and C_2 avoid numerical instability. SSIM is actually a map, so the mean SSIM is used to evaluate the overall frame quality:

$$\mathrm{SSIM}_\mu = \frac{\sum \mathrm{SSIM}}{W \cdot H}. \tag{2.8}$$

For both, PSNR and SSIM, high values indicate high similarity between the signals being compared. Figure 2.4 shows pictures of Lenna after adding (i) salt and pepper noise and (ii) a constant. The distorted pictures have same PSNR, but different SSIM.

2.1.3 Rate-distortion theory

Modern video encoders compress a video using an optimal set of coding tools, which are selected by minimising the size of the bitstream. The size is measured in terms of average of bits used to encode a second of video, and it is referred to as bit rate. As low bit rates are associated to high distortions, and vice versa, an efficient encoder should be capable of selecting and tuning its tools based on a trade-off between bit rate and distortion. This process is known as Rate-Distortion Optimisation (RDO) [131].

Given a number N of tools $T = \{T_0, T_1, \ldots, T_{N-1}\}$ and a raw signal $\mathbf{Y}$, the usage of the tool T_k generates a bitstream of bit rate R_k that once decoded produces the reconstruction $\hat{\mathbf{Y}}_k$.

It is the encoder job to find the best coding tool T^* subject to a given bit rate constraint R_c, such that the distortion is minimised:

$$\min_{T_k} D(\mathbf{Y}, \hat{\mathbf{Y}}_k), \text{ with } R_k < R_c. \tag{2.9}$$

This constrained problem is transformed into an unconstrained problem using a Lagrange multiplier λ:

$$T^* = \arg\min_{T_k} \{J(T_k)\}, \tag{2.10}$$

with the RD cost associated with the tool T_k

$$J(T_k) = D(\mathbf{Y}, \hat{\mathbf{Y}}_k) + \lambda R_k, \tag{2.11}$$

where D is a distortion metric, i.e. Eqs. (2.3) and (2.2).

In this context, the Lagrange multiplier is positive, following Shoham's and Cersho's theorem [125]: for any $\lambda \geq 0$, the solution to Eqs. (2.10) and (2.11) is also a solution to Eq. (2.9), which is proven in [31].

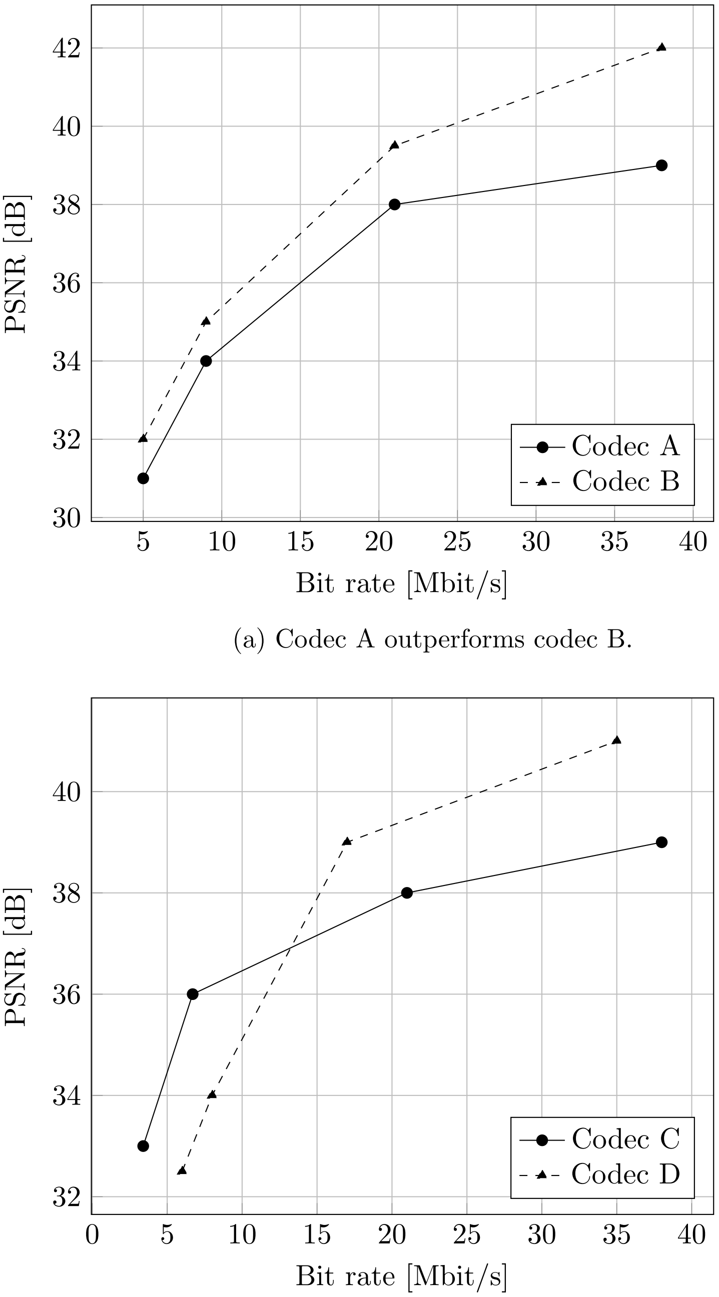
The choices made by an encoder by means of RDO depend on the target quality or bit rate. The same encoder may be used to compress a video optimally for different scenarios, maximising the user experience. Accordingly, the performance of the encoder can be evaluated at different compression levels.

The resulting bit rate and distortion values can be visualised using the so-called RD curves, which are useful to compare the efficiency of different encoders or coding tools. For instance, a new tool vs a benchmark or anchor.

Figure 2.5(a) compares the performance of two different codecs. It can be observed that codec B curve (marked with triangles) is above codec A curve (marked with circles). This indicates the codec B outperforms the codec A. From the bit rate point of view, the codec B achieves higher quality than the codec A and still uses less bit rate; and from the PSNR point of view, the codec B uses fewer bit rate than the codec A and achieves the same quality.

On the other hand, Figure 2.5(b) shows that codec C (marked with triangles) outperforms codec D (marked with circles) at low bit rates, whilst codec D outperforms codec C at high bit rates.

Another way to compare different coding schemes consists on using Bjøntegaard Delta (BD) rate, which measures the difference between the area below the RD curves of the schemes being compared [8, 9]. BD-rate is measured as a percentage and indicates the average bit rate savings for the same quality. Negative values represent coding gains and positive values represent coding losses. For both RD curves and BD-rate the larger the difference, the better the method with respect to the anchor.



(a) Codec A outperforms codec B.

(b) Codec C outpeforms codec D in low bit rates and viceversa.

Figure 2.5: RD curves.

2.2 Block-based hybrid video coding

Modern video coding standards, such as H.264/AVC [150] and H.265/HEVC [130], follow a block-based hybrid video coding scheme. The hybrid design uses both temporal prediction and transform coding of the prediction error [109]. Furthermore, a frame is partitioned into non-overlapping blocks that are independently coded. First, each block is compressed with predictive coding to exploit either temporal (inter-prediction) or spatial redundancies (intra-prediction). Next, transform coding enables further compression through quantisation. Afterwards, entropy coding is used to generate a bitstream that is suitable for storage and transmission. Consequently, significant compression efficiency is achieved by the synergy of the whole coding scheme and not by a single coding tool [122].

Figure 2.6 outlines the block-based hybrid video coding scheme.

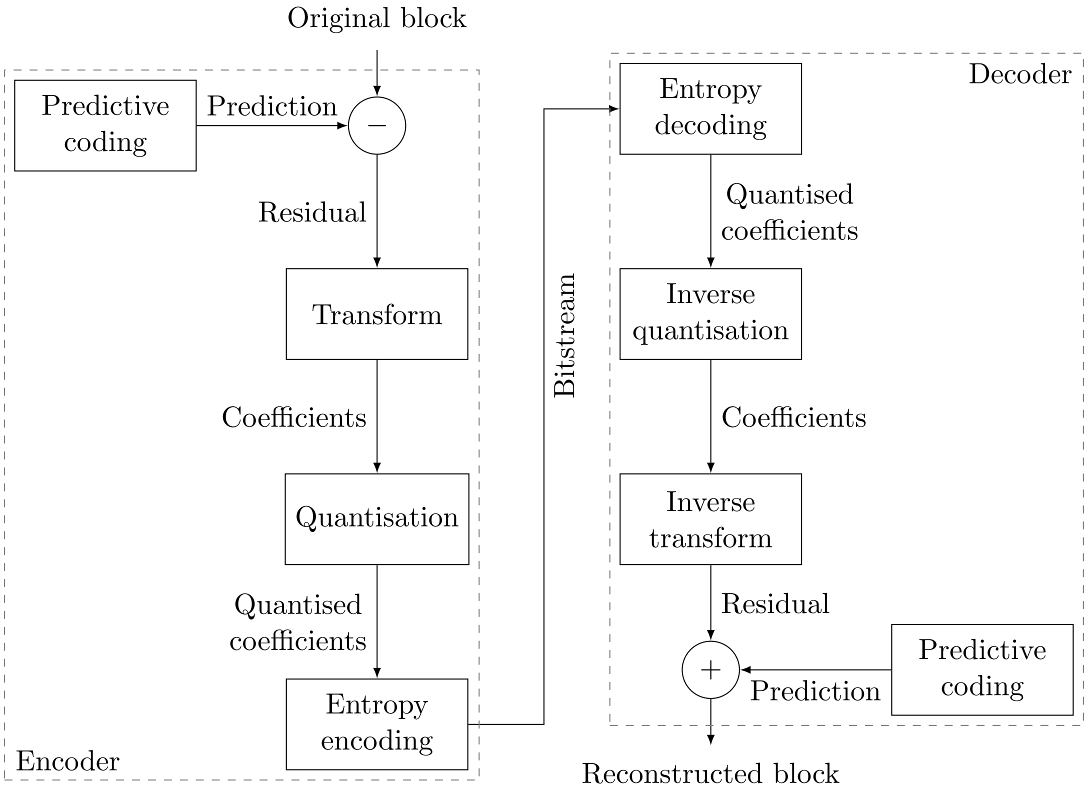


Figure 2.6: Block-based hybrid video coding scheme.

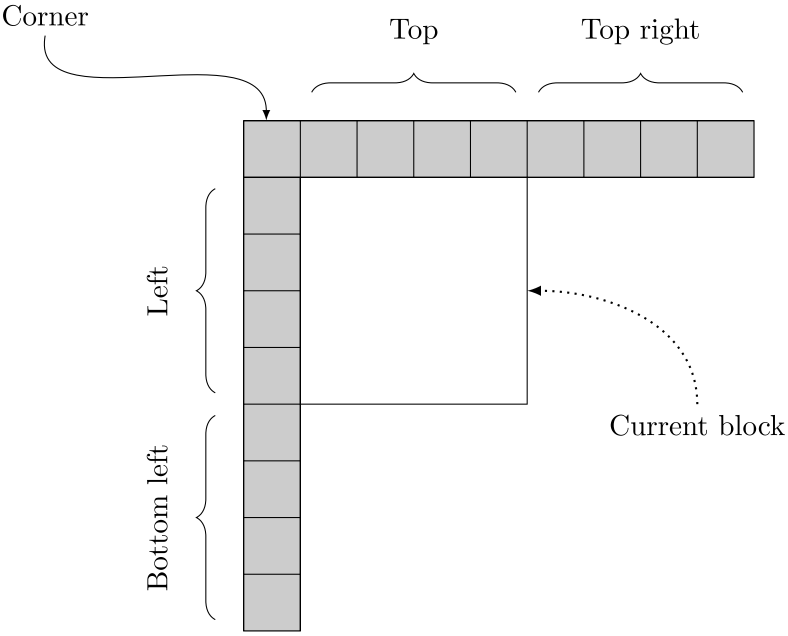


Figure 2.7: Common reference samples (in grey) used in intra-prediction.

Intra-prediction reduces spatial redundancies, resulting from the high correlation between neighbouring samples in a frame [60]. The prediction of a block is computed as a combination of reference samples from neighbouring blocks, meaning no reference to other frames in the video is needed. The process must be replicable at the decoder side. For this reason, only samples that have been previously decoded can be used as references. Specifically, those in the upper or left borders, as shown in Figure 2.7.

Inter-prediction exploits temporal redundancies, which are inherent in close by frames due to small changes between them. Given a current block, a reference frame and a search area, the process consists in searching for the block of same dimensions that minimises the distortion metric. The best match is marked by a Motion Vector (MV). The MV represents the motion of the block from the location in the current frame to the location in the reference frame [38] (Figure 2.8). Finding the MV is known as Motion Estimation (ME) and using the reference frame and the MV to get the prediction is known as Motion Compensation (MC).

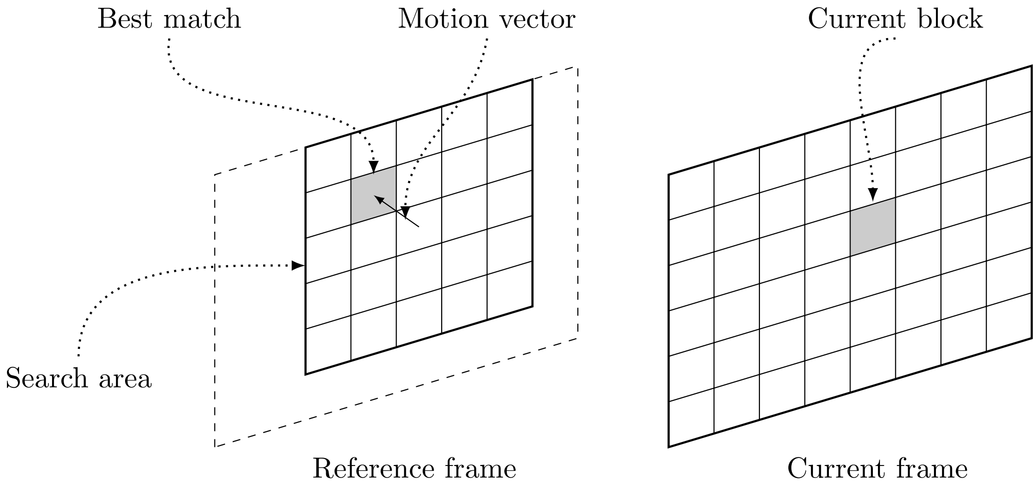


Figure 2.8: Motion estimation.

Once the prediction block is generated, the residual signal is computed. Namely, the sample-wise difference between the original block and its prediction. The residual is transformed into the frequency domain to decorrelate the signal, obtaining transform coefficients that enable the identification of the energy concentration. The transform coefficients are compressed by applying quantisation, where each coefficient is independently divided by a quantisation step, in order to map a range of values into a single value [41]. Transform and quantisation are used since most of the residual energy is expected to be contained in a few large coefficients, whereas smaller coefficients can be coarsely removed without affecting too much the quality of the reconstruction.

Finally, the quantised coefficients are compressed by means of entropy coding, which is lossless and assigns a unique codeword to each unique symbol in the compressed signal. According to Shannon's source coding theorem [123], the entropy is a lower bound of the average number of bits per symbol that can be achieved by any lossless coding scheme. The entropy H of a set of symbols $E = \{e_0, e_1, \ldots, e_{N-1}\}$ with probabilities $P = \{p_0, p_1, \ldots, p_{N-1}\}$ is given by

$$H = -\sum_{i=0}^{N-1} p_i \cdot \log_2(p_i). \tag{2.12}$$

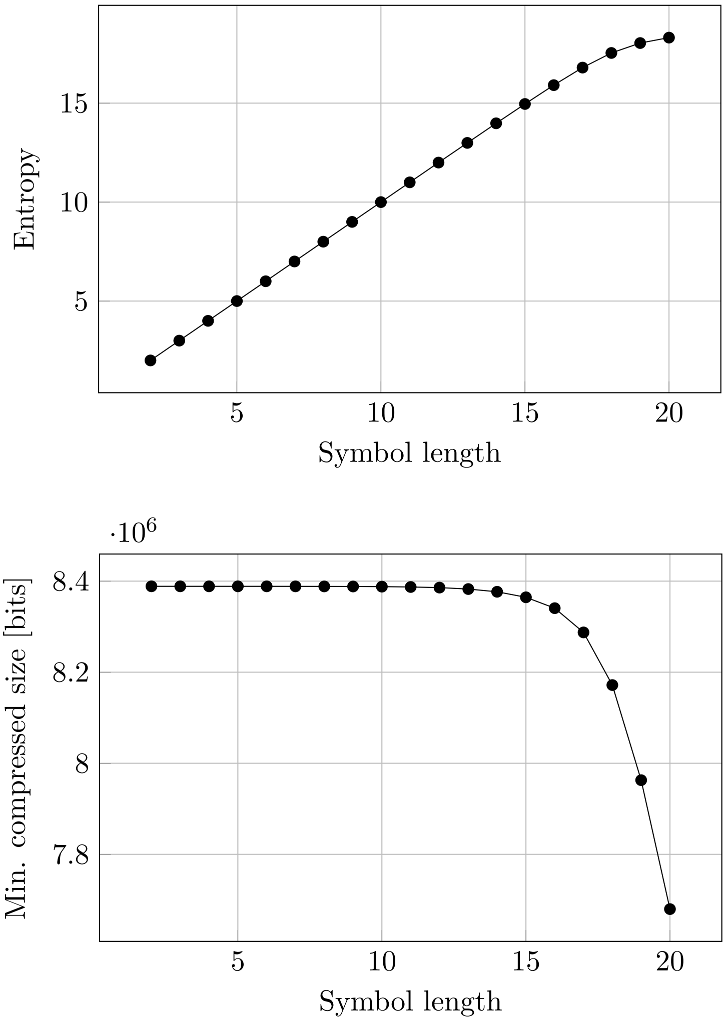


Figure 2.9: Shannon's source coding theorem. The average compressed size is inversely proportional to the entropy.

Each term of the summation is the length of a codeword. In this way, the most common symbols are represented with the shortest codewords. Consequently, an entropy encoder achieves compression by representing each symbol with a variable-length codeword. Figure 2.9 illustrates both the entropy and the minimum compressed size of a signal using symbols of up to 20 bits. The signal is a 1MB file created with the Linux random number generator. The compressed size is minimised for symbols of 20 bits, in which case the signal is represented using only 34% of them. In contrast, when using symbols of 15 and 18 bits, the signal is represented with 100% and 83% of them, respectively.

2.3 High Efficiency Video Coding standard

HEVC is the state-of-the-art standard and is developed by the Joint Collaborative Team on Video Coding (JCT-VC), a partnership between two standardisation organisations: the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC MPEG [130]. HEVC can deliver the same subjective visual quality as AVC with 50%, or even less, bit rate [133]; and compared to JPEG 2000 [134], it can achieve up to 44% coding gains [143].

2.3.1 Coding configuration

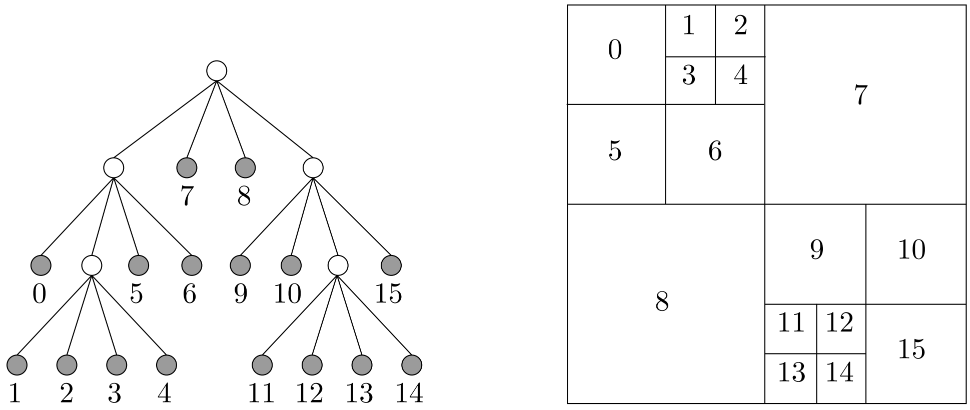
A frame can be categorised based on the prediction it uses. HEVC supports three frame types namely I, P and B. I frames use only intra-prediction. P and B frames use both inter- and intra-prediction. P frames can only reference previous decoded frames, whereas B frames can reference either past or future decoded frames.

HEVC has three coding configurations known as Low Delay (LD), RA and All Intra (AI). They are relevant to different applications and describe how the frames are going to be encoded. In LD configuration, frames are encoded in display order, which is suitable for scenarios like live video streaming where low delay is desired. The first frame is an I frame and the remaining frames are P frames. In RA configuration, frames are encoded in groups referred to as intra-periods, each of them starting with a RAP frame that is an I frame. The rest of the frames in the intra-period are B frames and they are not necessary encoded in display order. In addition, a decoder can start decoding a bitstream from any RAP frame [44], which means that frames following a RAP frame cannot depend on any frame preceding this RAP frame. RA is more complex than LD; nonetheless, it provides better compression efficiency, which can be more convenient for applications such as VoD. Finally, in AI configuration, all frames are I frames. AI may be more valuable for applications like frame-by-frame video edition, as there are no dependencies to other frames [112].

2.3.2 Block partitioning

In HEVC, a frame is partitioned into Coding Tree Units (CTUs) of size $N \times N$, supporting a maximum $N = 64$. Each CTU can be split into four Coding Units (CUs) of size $N/2 \times N/2$, where each one of them can be further split into four CUs of size $N/4 \times N/4$, and so on. Hence, a CU is recursively split until it is no longer necessary or the minimum CU size, $N = 8$, is reached. Figure 2.10 shows an example of a CTU being partitioned into CUs of variable sizes, which along with larger block sizes is key to increased coding efficiency [59, 94]. It has been shown that restricting the maximum CU size to $N = 16$ (the same size as that of the macro blocks in AVC) considerably affects the performance of HEVC in up to 30% BD-rate loss [149].

A CTU consists of one luma Coding Tree Block (CTB) and two collocated chroma CTBs. For the 4:2:0 chroma subsampling, a CTU consists of a $N \times N$ luma CTB and two $N/2 \times N/2$ chroma CTBs, such that there is the same number of CTBs for luma and each chroma component. The same relationship applies for CUs and its Coding Blocks (CBs). In addition, the decision whether to use inter- or intra-prediction occurs at the CU.



(a) Quad-tree.

(b) CTU.

Figure 2.10: Quad-tree partitioning in HEVC.

2.3.3 Intra-prediction

An intra CU can be predicted using a Prediction Unit (PU) of the same size as the CU. For the smaller CU size, the luma CB can also be split into four squared Prediction Blocks (PBs) of equal size (Figure 2.11), which are predicted independently. The reference samples are selected as in Figure 2.7, and the ones not available are replaced by predefined values [65]. Additionally, discontinuities in the reference samples are smoothed by applying a three-tap filter based on the prediction mode and the PU size [166].

HEVC defines 35 intra-prediction modes: planar, DC and 33 angular modes that follow the directions in Figure 2.12. Planar mode (0) produces a smooth gradient-like block. Each sample in a block is computed using a weighted average of four reference samples (out of the available neighbouring reference samples), where each weight depends on the position of the reference relative to the target sample, as illustrated in Figure 2.13(a) for a 4×4 block. DC mode (1) creates an even block by averaging the top and left reference samples (Figure 2.7) and using the resulting value to fill the whole block, as shown in Figure 2.13(b). Angular modes (2 to 34) model high frequencies and create blocks with edges following the directions in Figure 2.12. Angular modes 2, 18 and 34, 10 (horizontal) and 26 (vertical) copy the reference sample values, as displayed in Figures 2.13(c) and 2.13(d), the other modes interpolate the two references that pertain to the angle of the mode [65].

(a) Luma/chroma PBs of size $N \times N$.

(b) Luma PBs of size $N/2 \times N/2$.

Figure 2.11: PB sizes for intra-prediction in HEVC. (a) represents a PB that has the same size as the parent CB. (b) represents the PB resulting from partitioning the parent CB into four.

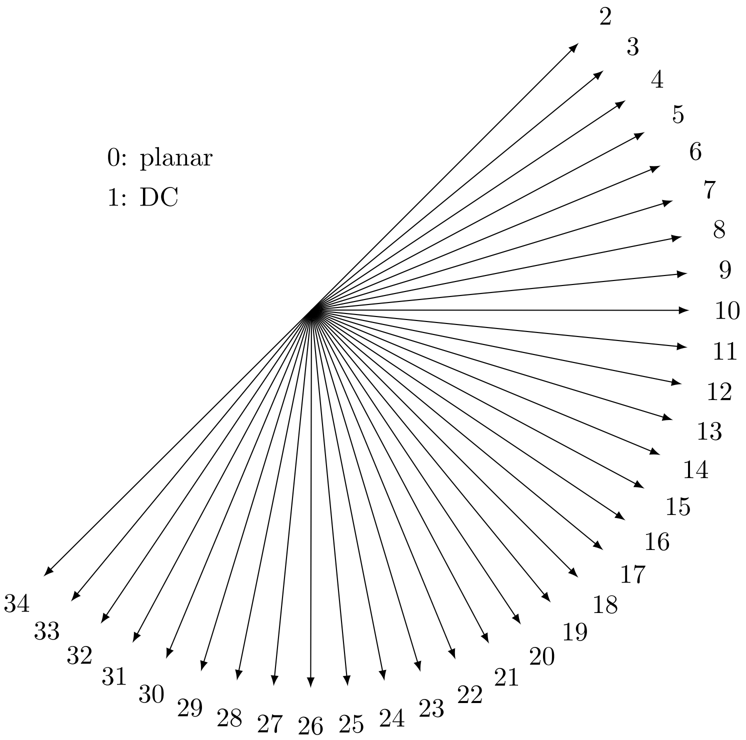
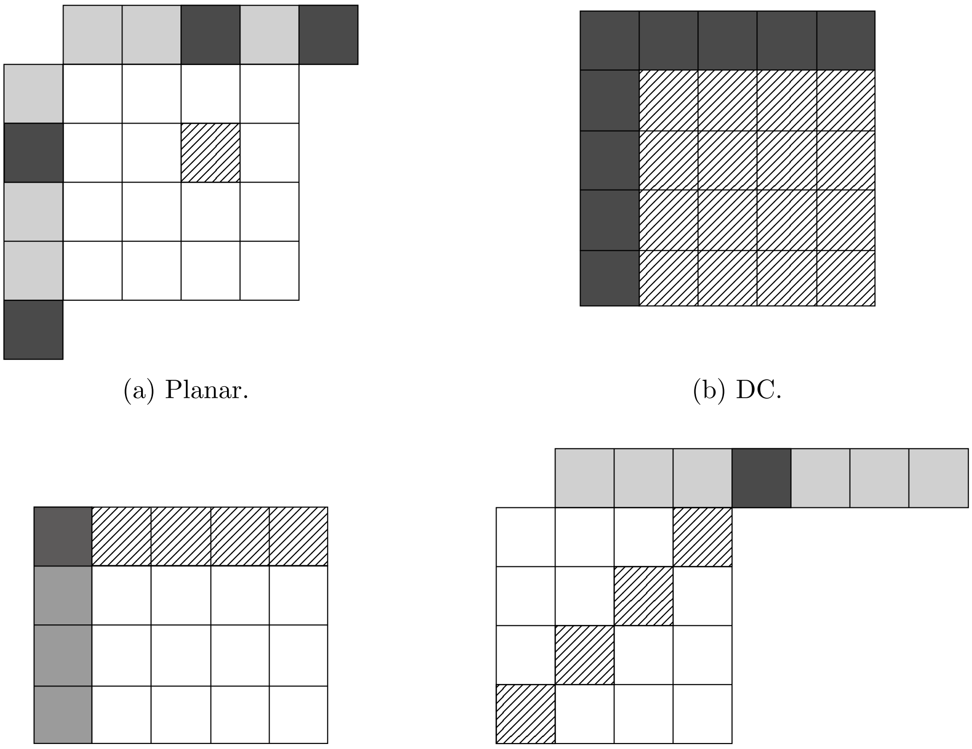


Figure 2.12: Intra-prediction modes supported by HEVC. 2 to 34 are angular modes.

Finally, discontinuities are smoothed by filtering the boundary of blocks predicted with DC [99], horizontal and vertical modes.

All 35 intra modes are available for luma PBs. The selected mode is signalled using a list of 3 Most Probable Modes (MPMs) that is populated with the modes selected for left and top neighbouring CBs, if they are available, and with some default modes. If the selected mode is in the MPM list, an index is signalled to indicate the element of the list to be used. Otherwise, the intra mode is signalled with a fixed codeword of 5 bits. Intra-prediction for chroma CBs is slightly different. In the interest of reducing the complexity and the number of bits needed to signal the chroma intra mode, only up to 5 intra modes are available. These modes are the selected mode for the luma PB (derived mode), planar, DC, horizontal and vertical. If the selected mode is not the derived mode, a fixed codeword of 2 bits is used to signal the chroma intra mode [65].



(a) Planar.

(b) DC.

(c) Angular 10 (horizontal) →.

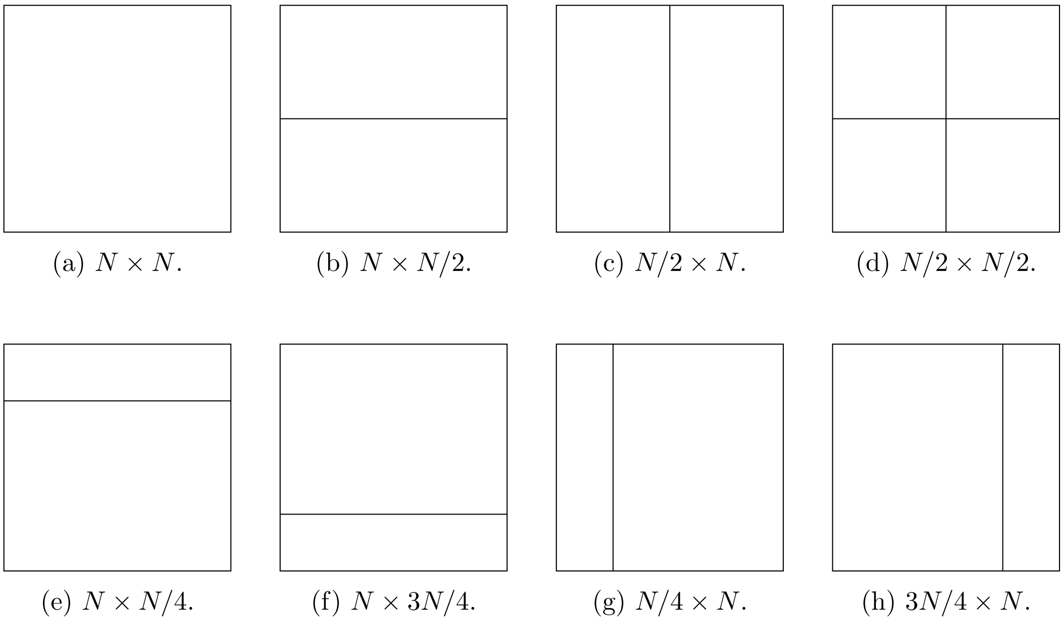
(d) Angular 34 ↙.

Figure 2.13: Intra-prediction modes in HEVC. Grey samples are the references used by the mode. Dark grey samples are only used in block samples marked with diagonal lines.

Even though all samples within a PU are predicted with the same intra mode, the actual process is not done at PU level. As explained later in this section, each PU can be further partitioned into square blocks, referred to as Transform Units (TUs), which are used in residual coding. Intra-prediction is performed separately for each TU within a PU, using different reference samples.

2.3.4 Inter-prediction

When the CU is coded with inter-prediction, it can be partitioned into eight different PUs, as illustrated in Figure 2.14. Namely, a CU can be predicted as a single PU of the same size or it can be split into four squared or two rectangular PUs. The latter includes symmetrical PUs and asymmetrical PUs.



(a) $N \times N$.

(b) $N \times N/2$.

(c) $N/2 \times N$.

(d) $N/2 \times N/2$.

(e) $N \times N/4$.

(f) $N \times 3N/4$.

(g) $N/4 \times N$.

(h) $3N/4 \times N$.

Figure 2.14: PU sizes for inter-prediction into in HEVC.

Inter-prediction is computed at the PU level using one or two previously decoded frames from two lists of reference frames. These processes are known as uni-directorial and bidirectional prediction. Unidirectional prediction uses a single frame from either of the two lists; whereas bidirectional prediction is the average of the two predictions produced using one reference frame from each list. These two temporal predictions are displayed in Figure 2.15.

HEVC adds fractional-pel accuracy [37] as frame motion is not necessarily pixel by pixel. Half-pel and quarter-pel [141, 142] are supported. Since there are no values at the fractional-pel locations, they are computed by means of interpolation. Half-pel samples are computed using an 8-tap filter and quarter-pel samples are computed using a 7-tap filter.

Even though CTU partitioning added adaptability to HEVC, often neighbouring PUs have the same motion information [46]. This redundancy is exploited with the merge mode, which groups PUs with common motion information and signals a single motion information set for a whole region. A list of five candidates is created considering decoded spatial and temporal collocated neighbouring PUs, as well as generated PUs.

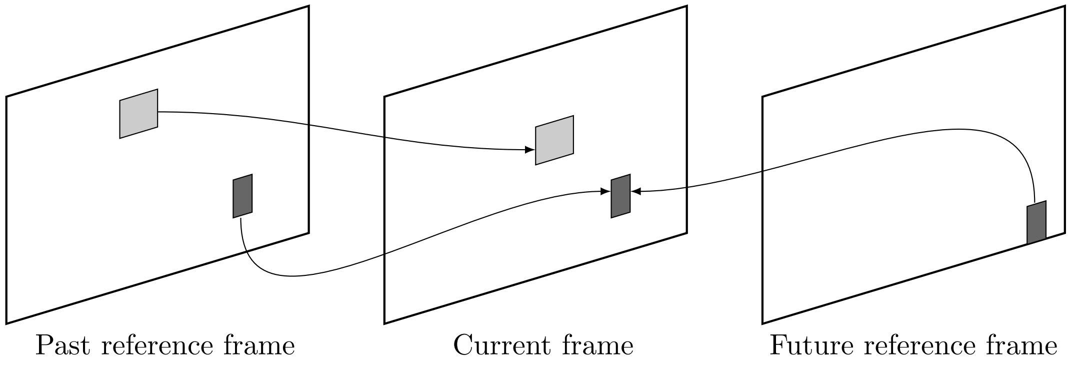


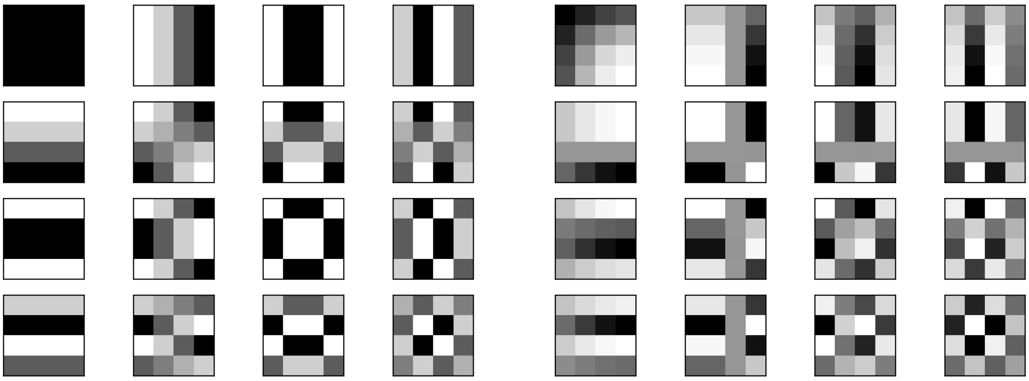
Figure 2.15: Unidirectional (light gray) and bidirectional (dark gray) prediction in HEVC.

The conventional prediction is tested in an RD sense against the predictions obtained with the merge candidates. A flag is signalled to indicate whether merge mode used. If it is, an index is signalled to indicate the candidate. Otherwise, the motion information is signalled once for uni-directional prediction or twice for bidirectional prediction. One special case of the merge mode is the skip mode, which is applied at CUs level. This mode is signalled with a skip flag and the candidate index. If a CU is encoded using skip mode, it is not split into smaller PUs and there is no residual coding as the prediction becomes the reconstruction.

If neither merge mode nor skip mode are used, the motion information to signal are the reference index, the motion direction and the MV, transmitted as the difference between the MV and a predictor [83].

2.3.5 Residual coding and other tools

When a CU is not skipped, the residual is transformed and quantised. For this purpose, a CU is split into TUs using a residual quad-tree [43]. The residual can be processed as a single TU of same size as the CU or it can be recursively split into four squared TUs, until either no further splitting is required, or the minimum TU size is reached. HEVC supports TU sizes from 4×4 to 32×32.



(a) DCT II. (b) DST VII.

Figure 2.16: 4 × 4 transform basis.

Each TU is transformed and quantised independently using an integer approximation of the Discrete Cosine Transform (DCT) [5] II matrix of the same size. Additionally, 4 × 4 luma Transform Blocks (TBs) that use intra-prediction are transformed with an integer implementation of the Discrete Sine Transform (DST) [13] VII.

HEVC also introduces the transform skip mode [36] where transform and inverse transform are ignored, which showed to be beneficial for screen content [103].

Let **T** be the transform matrix and **R** be the residual, the transformed coefficients **C** are computed as

$$\mathbf{C} = \mathbf{T} \times \mathbf{R} \times \mathbf{T}^{\top}. \tag{2.13}$$

Since HEVC uses orthogonal transforms, the inverse transform is given by

$$\tilde{\mathbf{R}} = \mathbf{T}^{\top} \times \tilde{\mathbf{C}} \times \mathbf{T}, \tag{2.14}$$

where $\tilde{\mathbf{C}}$ are the transform coefficients obtained after quantisation and inverse quantisation, and $\tilde{\mathbf{R}}$ is the residual block obtained after the inverse transform [106].

Figure 2.16 depicts the 4 × 4 basis for both DCT and DST.

Next, the transform coefficients are quantised. As mentioned in Section 2.2, quantisation is a division by a quantisation step. Additionally, it is a lossy process as transform coefficients cannot be fully recovered after the quantisation. The inverse quantisation is a multiplication by the quantisation step, which depends on the Quantisation Parameter (QP) $\in [0, 1, \ldots, 51]$ [106].

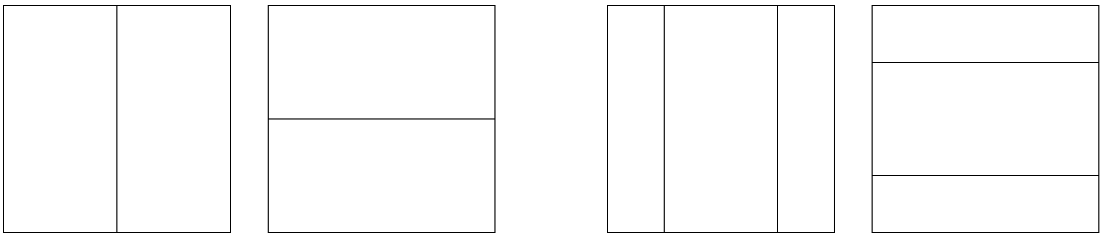
Finally, all quantised coefficients are entropy encoded, using the Context-Adaptive Binary Arithmetic Coding (CABAC) [96], and written to the bitstream to be sent to the decoder. The symbols are encoded according to their probability distribution. Hence, frequently used symbols are assigned shorter codewords and vice versa.

More HEVC tools, not as relevant as the aspects described in this chapter, can be found in the literature [132, 151].

2.4 Versatile Video Coding

To meet new demands, the next generation video coding standard H.266/VVC [14] was developed by the Joint Video Experts Team (JVET) and finalised in July 2020. Like its predecessors, VVC is based on a hybrid coding architecture where each frame is split into blocks that are compressed independently. VVC includes many new coding tools, and refines and improves inherited tools from the HEVC standard.

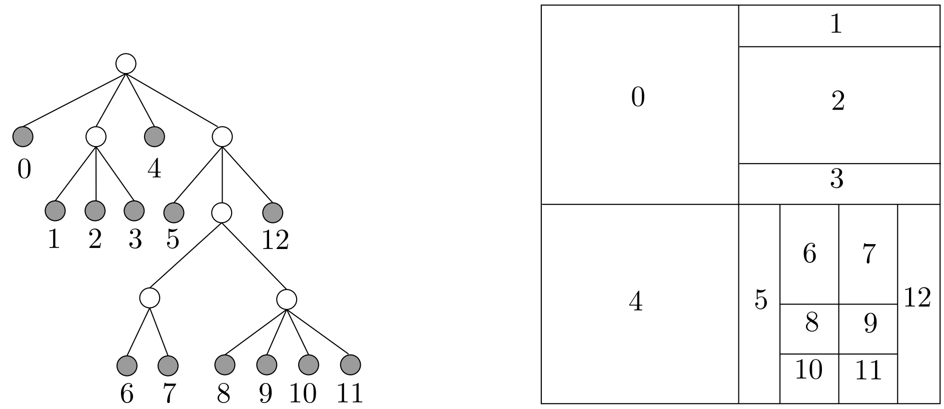
In VVC, frames are partitioned into CTUs, where the maximum size is 128×128 for luma. Each CTU can be split using a Quad-Tree plus Multi-Type Tree (QT+MTT) [77], using binary or ternary splits. The splits are illustrated in Figure 2.17 anda QT+MTT and the corresponding CTU partitioning is displayed in Figure 2.18. Each leaf of a multi-type tree is called a CU, which can have either a square or rectangular shape. Moreover, CUs are input to the prediction and transform stages without any extra partitioning.



(a) Binary splits.

(b) Tertiary splits.

Figure 2.17: Multi-type tree splitting modes.



(a) QT+MTT

(b) CTU.

Figure 2.18: QT+MTT in VVC.

Several coding tools have been proposed to improve intra-prediction. Namely, the number of angular modes was extended from 33 (in HEVC) to 65 [21], including wide angle extensions [164] for rectangular CUs. Additionally, there is the Multi-Reference Line Prediction (MRLP). In this case, the prediction can be computed from reference lines non-adjacent to the current block [16], as shown in Figure 2.19, and using for reference samples. According to the mode and the CU size, the interpolation is done with either a 4-tap Discrete Cosine Transform based Interpolation Filter (DCTIF) or a Gaussian filter [33]. The luma intra mode is signalled with six MPMs [75].

Furthermore, position dependent prediction combination [118] combines unfiltered boundary reference samples and predictions computed with filtered samples. It is also possible to perform both prediction and transform independently on smaller sub-partitions of a CU [26].

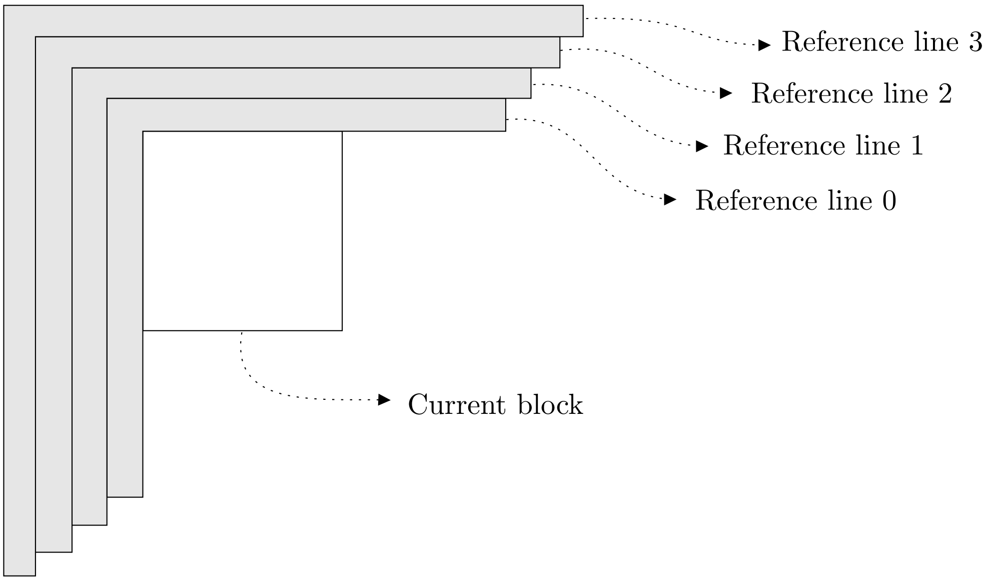


Figure 2.19: Multiple reference lines.

In addition, new modes are provided by Matrix-based Intra-Prediction (MIP) [120], which (i) down-samples the references, (ii) computes a down-sampled prediction and (ii) applies up sampling to get the final prediction. Finally, cross component linear model is used to predict chroma samples from reconstructed luma samples of the same CU.

Inter-prediction tools include geometrical CUs; affine motion prediction that enables other types of motion such as scaling, rotation, shape changes and shearing; $^1/_{16}$ pel motion vector storage accuracy and MC; and combination of spatial predictions and temporal predictions.

VVC also incorporates an enhanced and faster CABAC; new transform matrices, including secondary transforms; and the adaptive loop filter. These and other tools can be found in the literature [14]. Table 2.1 presents the key differences between HEVC and VVC.

Table 2.1: HEVC vs VVC.

Tool	**HEVC**	**VVC**
Partitioning	CTU up to 64 × 64 Square CUs Quad-tree	CTU up to 128 × 128 Square and rectangular CUs QT+MTT
Intra-prediction	33 angular modes 2-tap filter Square blocks 1 reference line 3 MPMs	65 angular modes 4-tap filter Square and rectangular blocks MRLP 6 MPMs MIP Chroma cross component
Inter-prediction	1/2 accuracy Translation motion Square and rectangular blocks	1/16 accuracy Affine and translation motion Square, rectangular and geometrical blocks Optical flow Combined inter- and intra-prediction
Residual coding	Blocks up to 32 × 32	Blocks up to 64 × 64

2.5 Neural networks

NNs are an attempt to imitate how the human brain works in terms of data processing and pattern recognition. In this way, a system learns to perform a task by analysing training samples instead of relying on hand-crafted heuristics [68].

In general, NNs are made of stacked layers. Each layer receives an input and transforms it with linear and non-linear functions. Next, the resulting values are passed to a subsequent layer. An NN learns a set of parameters θ, which are utilised to compute layer operations. These parameters are trained by minimising a loss function $\mathcal{L}$, such as Eqs. (2.2) and (2.4), that measures the error between ground-truths (target data, commonly hand-labelled) and predictions generated by the NN.

The rest of this section describes common layers in NNs as well as how the training is done.

2.5.1 Layers

Convolution

Convolutions are the core of CNNs. Each convolution layer consists of a set of kernels and a set of additive biases that are learned during the training stage. The convolutions generate feature maps from the inputs [40].

Given a matrix $\mathbf{X}$, a kernel $\mathbf{F} \in \mathbb{R}^{M \times M}$ and a bias b, the convolution layer $\mathcal{C}$ is defined as

$$\mathcal{C}(\mathbf{X})_{i,j} = b + \sum_{u=-M'}^{M'} \sum_{v=-m'}^{m'} y_{i-u,j-v} \cdot f_{u,v}, \tag{2.15}$$

where M is typically odd and $M' = \lfloor M/2 \rfloor$.

Fully connected

Fully Connected (FC) layers connect all units to every unit in the previous layer. Typically, the input to a FC layer is a vector and the operation carried out is a matrix-vector multiplication, followed by a bias addition.

Activation function

NNs deals with complex tasks by applying an element-wise non-linear operation called activation function ρ, which typically clips the input values to certain range [40]. Some of the most common activation functions include the Rectified Linear Unit (ReLU) [105]:

$$\rho(\mathbf{z})_i = \max(0, z_i); \tag{2.16}$$

The Parametric Rectified Linear Unit (PReLU) [45]:

$$\rho(\mathbf{z}, \mathbf{a})_i = \begin{cases} z_i & \text{if } z_i > 0 \\ a_i z_i & \text{otherwise,} \end{cases} \tag{2.17}$$

where **a** is a trainable parameter; and the Exponential Linear Unit (ELU) [23]:

$$\rho(\mathbf{z})_i = \begin{cases} z_i & \text{if } z_i > 0 \\ \exp(z_i) - 1 & \text{otherwise.} \end{cases} \tag{2.18}$$

Pooling

The pooling operation is widely used in CNNs to achieve invariance to frame transformations and data perturbations. It also reduces the number of parameters of following layers [12]. Max-pool is perhaps the most used pooling operation due to its remarkable performance [11]. However, this and other regular pooling operations result in the loss of data. This issue is addressed with the lossless pooling operation that downscales a single-channel frame to a multi-channel frame [138].

2.5.2 Back-propagation

The parameters θ of an NN are typically updated using back-propagation [69], that minimises the loss function $\mathcal{L}$ using the gradient descent algorithm. The gradient is computed, and the update is done iteratively guided by the negative gradient direction, moving towards the steepest descent [115]. The step size of the update is given by the learning rate η:

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla_{\theta_t} \mathcal{L}(\theta_t). \tag{2.19}$$

Consequently, a higher learning rate results in bigger steps taken in the update process, which may result in less time for the model to converge on an optimal set of parameters. However, using a learning rate too high means that the jumps are too large and not precise enough to reach the optimal point.

Regularisation

Over-fitting is a common problem for NNs, and it occurs when a model is too complex and too closely fits to the training data, leading to a degraded generalisation. This problem can be addressed by increasing the size of the training dataset; reducing the size of the network; or utilising weight decay, which adds a regularisation term to the loss function, such as ℓ_1 or ℓ_2 norm of the parameters, in order to encourage sparsity and the usage of low coefficients [40]. The regularisation term is typically defined using the ℓ_2 norm:

$$\lambda||\theta||_2^2, \tag{2.20}$$

where λ is the regularisation parameter. An additional technique is early stopping [7], for which the network performance is evaluated every N updates using a validation set (different than the training set) and if the error drops after M consecutive checks, then the training is stopped.

2.6 Neural networks in video coding

NNs have been used to improve different modules of traditional video coding schemes, with either of two goals in mind: optimise the encoder or improve the compression efficiency.

2.6.1 Encoder optimisations

Video encoders tend to be more complex than decoders as the encoder evaluates multiple tools to find the best ones to compress a video, whereas the decoder only applies the tools used to generate the bitstream. An encoder that performs exhaustive search will get the highest compression efficiency, but also a high computational complexity. Accordingly, a practical encoder is likely to adopt heuristic algorithms to support mode decision or reduce the search space.

Block partitioning

Intra partitioning has been addressed from different perspectives. For instance, Liu et al. [87] proposed a CNN to predict the partitioning of CU/PU. The CNN predicts whether a block has homogeneous content, and if not the block is split. Li et al. [76] introduced a CNN classifier for CTU partitioning where the class is the depth of the quad-tree and indicates what CUs are to be used. Similarly, Amer et al. [6] utilised a CNN classifier at each depth of the quad-tree to either select the current CU or split it further. Feng et al. [32] addressed CTU partitioning in three steps. First, a CNN is used to determine if the CTU is complex or not. Second, this information is used to select a CNN that predicts a depth range for the quad-tree. Finally, the best partitioning is selected in an RD sense by evaluating the CUs in the predicted depth range. Diversely, Xu et al. [155] focused on both inter and intra CTU partitioning. The latter is predicted with a CNN, whereas the former is predicted by an Long Short-Term Memory (LSTM) [48] as neighbouring frames have correlated CTU partitioning. Moreover, Wang et al. [148] use a CNN to predict the depth range of the tree.

Intra-prediction

Intra-prediction mode decision is tackled by Laude and Ostermann [67] with a CNN classifier. In contrast, Song et al. [127] used a CNN to predict a limited number of PUs candidates. More recently, Chen et al. [20] created a CNN framework that combines CU/PU size decision and selection of intra PU candidates.

Rate control

A video encoder is responsible for producing bitstreams that do not overflow the bandwidth, for which rate control is essential. For intra coding, Li et al. [79] used a CNN to predict, per CTU, the parameters α and β of the R-λ model. Xu et al. [154] presented a CNN framework for RD modelling, where the distortion is the SSIM index. Hu et al. [50] utilised reinforcement learning [34] to estimate the QP to use in a CTU.

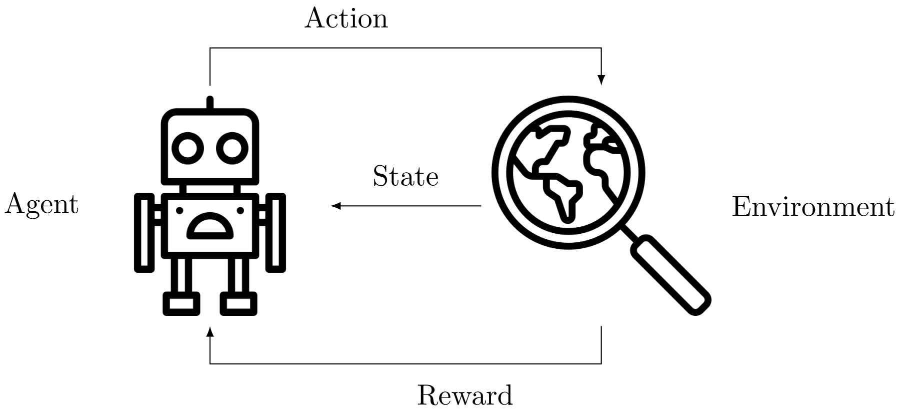


Figure 2.20: Reinforcement learning.

In reinforcement learning, an agent learns to evolve in an interactive environment using the reward from its actions, as depicted in Figure 2.20. In this scenario, the environment state consists of the texture of the CTU and the bit balance, the action is the selection of the QP value (modelled with a CNN) and the reward is the negative distortion to be maximised.

2.6.2 Improving compression efficiency

So far the approaches mentioned serve as decision tools. In contrast, the following tools constitute new ways to achieve compression.

Intra-prediction

New intra-prediction modes may be selectively used in each block in addition to traditional intra modes. For luma prediction, Li et al. [73] proposed the Intra Prediction Fully Connected Network (IPFCN), which learns to predict complex textures from multiple lines of reference samples. The encoder signals a flag to indicate whether to use a traditional intra mode or the NN-based mode. A subsequent approach [74] consists on learning two new modes. The IPFCN is trained twice, first with blocks that are well predicted with DC and planar modes, and next with blocks that are well predicted with angular modes. As before, a flag is signalled to indicate the type of intra-prediction, plus a bin to distinguish the IPFCN to use.

A more flexible approach is devised by Pfaff et al. [111] where an FCN design generates several modes that are jointly learned during the training, and do not require creating separate datasets. In addition, the predictions are in the transform domain. A flag is signalled to specify the intra-prediction type. For the actual mode, a second FCN predicts a list of MPMs. This idea is further developed by Helle et al. [47], where the FCN is implemented as an affine linear function that is further simplified.

Alternatively, Hu et al. [51] used a Recurrent Neural Network (RNN) that learns how to use only informative reference samples to produce complex textures, as well as to identify when a new texture is to be predicted. In contrast, Cui et al. [24] introduced CNN to refine blocks generated with traditional intra modes. The CNN estimates the residual signal from the reference samples and the prediction block. The target block is computed as the difference between the prediction block and the learned residual.

Chroma prediction is addressed by Li et al. [80]. This method combines a CNN to extract features from the luma prediction and an FCN to extract features from luma and chroma reference samples, which are later fused to generate both chroma predictions. This mode is added as a new option to be evaluated. Therefore, the length of the codeword may need to be increased in order to derive correctly the selected chroma mode. Meyer et al. [98] focused on cross-component prediction using two CNNs: one for luma prediction and another for chroma prediction. The CNNs have the same structure, except for the inputs and outputs: the luma CNN produces a single prediction, whereas the chroma CNN generates the two chroma predictions. The list of MPM is expanded to hold an extra element, and the luma CNN mode is always the last one. The cross-component comes into play since a chroma block is only predicted with the CNN if the corresponding luma block is predicted with the CNN.

Inter-prediction

For uni-directional prediction Lin et al. [84] extrapolate a frame from previous reconstructed frames using a Generative Adversarial Network (GAN). The new frame is added to the list of reference frames.

Likewise, Laude et al. [66] employed Predictive coding Network (PredNet) [88] to create a virtual frame out of all reference frames that replaces one of these in the list.

Unidirectional luma prediction has also been refined by adding spatial contextual information. For instance, Huo et al. [53] devised a CNN that improves the accuracy of the prediction block using its spatial neighbouring samples. Similarly, Wang et al. [146] utilised a FCN to extract features from (i) neighbouring samples of the target block (in the current frame) and (ii) neighbouring samples of the prediction block (in the reference frame). Finally, a CNN learns the actual prediction from the prediction block and the features learned by the FCN.

For bidirectional prediction, Zhao et. al [165] designed a CNN that fuses the two candidate blocks, replacing the traditional method. A whole new approach synthesizes a new reference frame [163], using the deep voxel flow CNN [86], from two reconstructed frames with the same temporal distance to the target frame. The synthesized frame is enhanced using a second CNN and added to the list of references. Since it can be derived at both encoder and decoder, no extra signalling is required.

Fractional-pel is addressed as a super resolution problem. Zhang et al. [160] use the Very Deep Super Resolution (VDSR) network [61] to replace the traditional half-pel interpolation in luma blocks. A mask is added to the VDSR, since the integer positions need to be preserved. The network learns the residual between the original frame and the interpolated low resolution frame, which is created by interpolating the integer samples using the DCTIF [3].

A similar approach by Yan et al. [157] use the Super Resolution Convolutional Neural Network (SRCNN) [28], where the original frame is blurred and the samples in even positions are used to create the low resolution frame, whereas the samples in odd positions are considered the fractional samples. Further developments showed that integer samples can be interpolated from fractional samples [156, 158], which is exploited to train the network in an unsupervised way.

Entropy coding

Proposals for entropy coding address the estimation of probability distributions. Song et al. [128] used a CNN to predict the probability distribution of intra modes from neighbouring blocks and MPMs. Puri et al. [114] designed a CNN to estimate the probability distribution of transforms from the quantised coefficients.

Ma et al. [91] proposed a CNN to predict the probability distribution of DC coefficients of intra-prediction residuals, from the current reconstructed block and three neighbouring reconstructed blocks. This work is extended for AC coefficients [90] and inter-prediction syntax elements [92], such as merge flag, merge index, MV, among others. In the latest approach, the CNN is trained with the syntax elements from both the current block, and spatial and temporal neighbouring blocks.

Luz et al. [89] proposed a saliency-driven adaptive quantisation model, aiming overall bit rate reduction for a target perceptual quality. This technique allocates more bits on regions where viewers focus their attention, and vice versa. To this end, first, the saliency map of a frame is calculated using a CNN. Next, the average saliency per CU is calculated and used to compute the QP.

Afonso et al. [2] introduced a framework to maximise rate-quality. The first step consists of estimating the parameters required to compute the temporal and spatial down-sampling of a video, which are obtained using quality resolution optimisation and signalled within the bitstream. Second, the resolution optimised video is calculated and encoded. At the decoder, the re-sampled frames are decoded and finally they are up-sampled spatially (deploying a CNN) and temporally (computing frame average and nearest neighbour interpolation).

In-loop filtering

Zhang et al. [162] proposed a CNN as an additional in-loop filter. The CNN is trained separately for I frames, P frames and B frames. Additionally, the QP range is split in bands and a dedicated CNN is trained per band.

Dai et al. [25] designed a CNN that replaces traditional in-loop filters in I frames. For B frames, the CNN is considered an optional filter at CTU or CU level. A flag is added at CTU level to indicate if the CNN filter is to be used. Otherwise, a classifier decides whether to use the CNN filter at CU level.

Likewise, Jia et al. [58] apply a CNNfilter at CTU level and a flag is encoded to specify whether it is used. This work also considers training multiple CNNs and a content analysis network to select the model to be applied.

Post filtering

Wang et al. [145] designed a CNN to enhance the quality of reconstructed frames at the decoder side. The CNN is tested in all coding configurations, even though the training is only done with frames compressed under AI configuration. Similarly, Yang et al. [159] devised two CNNs to improve I frames and P/B frames.

2.7 Summary of neural network based video coding tools

Table 2.2 summarises the key NN-based contributions to block-based hybrid video coding schemes.

Table 2.2: Summary of key NN-based video coding tools.

Encoder optimisation	Block partitioning	Predict tree for intra blocks [76, 155] Predict tree for inter blocks [148, 155] Stop split per block size [6, 87] Prune tree [32]
	Intra-prediction	Mode decision [20, 67] Prune search space [20, 127]
	Rate control	Parameter estimation [50, 79] RD modelling [154]
Compression efficiency	Intra-prediction	Prediction refinement [24] Luma prediction [47, 51, 73, 74, 98, 111] Chroma prediction [80, 98]
	Inter-prediction	Prediction refinement [53, 146] Fuse blocks [165] New reference frame [66, 84, 163] New prediction [163] Fractional-pel [156–158, 160]
	Entropy coding (probability distribution)	Intra modes [128] DC coefficients of intra residual [90, 91] Inter-prediction syntax elements [92] Transforms [114]
	Enhancement	In-loop filtering [25, 58, 162] Post filtering [145, 159]

Chapter 3

Estimation of rate-distortion of I frames

Due to the inherent complexity of video coding schemes, it is generally difficult to estimate the effects of an encoder on a frame in terms of number of bits and distortion, without doing the actual encoding. As such, schemes to accurately predict the number of bits and distortion generated by an intra encoder are highly beneficial. This chapter presents an CNN-based approach to estimate RD parameters for I frames. The parameters are predicted from original content and, therefore, the model can be known before the actual encoding takes place, avoiding the usage of multi-pass encoding. The results are accurate and show the potential on using NN-based tools in video coding.

The rest of the chapter is organised as follows. Section 3.1 provides a brief introduction to rate control. Section 3.2 presents the proposed method. Section 3.4 reports the experiments and performance evaluation of the techniques proposed in this chapter, followed by conclusions in Section 3.5.

3.1 Background work

As mentioned early in Chapter 2, video coding standards define the bitstream syntax, as well as the decoding process to generate the reconstructed video. In contrast, the encoding is usually designed to meet usage needs, such as quality, bit rate and memory consumption. In metered networks, like mobile data plans, consumers pay to use a given amount of data in a period. Extra data is allowed too, and entails an extra fee. Hence, bitstreams with low bit rate may be more suitable for distribution under limited connectivity.

For distribution, frames are encoded using the so-called RA configurations, in which most frames are B frames and a few I frames are inserted periodically in the sequence. An RA simulation of the reference software for HEVC, called HEVC test Model (HM), version 16.9 [57, 117] showed that, on average, a B frame is represented with significantly fewer bits than an I frame (see Table 3.1) for high bit rate (QP 22) and low bit rate (QP 37). Furthermore, as shown in Figure 3.1, a B frame requires less than 15% of the bits used by an I frame. As I frames are used for reference by subsequent B frames, they should be encoded at the highest quality too [144].

3.1.1 Rate control

Rate control aims to maximise the video quality under bit rate constraints. Typically, rate control methods allocate the target bit rate among the frames in the sequence, and then selecting the coding parameters to meet this allocation. Rate control can be single- or multi-pass. In single-pass rate control, the bit allocation and setting of parameters are done according to prior knowledge on the video, like data collected over previously encoded frames. Moreover, single-pass rate control is preferred for applications where the encoding speed is a priority, such as video streaming. On the contrary, multi-pass rate control encodes the video multiple times and the results of one encoding are transferred to and used by subsequent encodings. Multi-pass rate control takes much

Table 3.1: Average amount of bits used to represent I and B frames.

Sequence	Frame	Bits per frame QP 22	 QP 37
Tango	I	6001522	328210
	B	370304	27988
FoodMarket	I	2867984	484046
	B	242358	26945
RitualDance	I	1189666	225732
	B	137744	17880
BQMall	I	561592	109444
	B	41179	4036

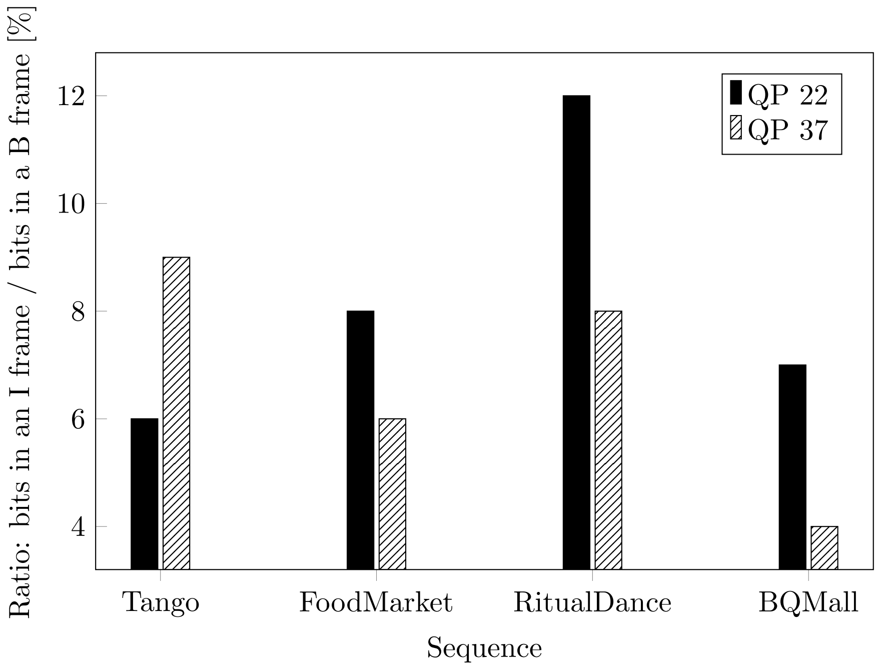


Figure 3.1: Bits of an I frame used to represent a B frame.

longer as the whole video is encoded more than once. Hence, multi-pass rate control is most used in applications where low delay is not critical, like VoD [167].

Video coding standards offer a wide set of tools to compress a video, and RDO is used to select the optimal coding tools by minimising Eq. (2.9). Hence, RD performance plays a key role in rate control algorithms, which are expected to accurately achieve the target bit rate and to minimise the distortion.

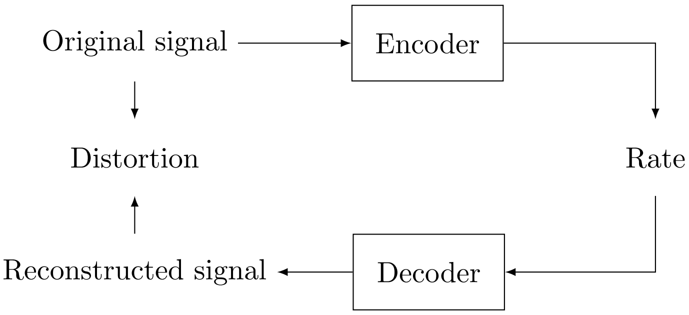


Figure 3.2: RD scheme

RDO can be solved when the RD cost, Eq. (2.11), is a convex function and both R and D are continuous and differentiable everywhere, by setting the derivative to zero [78]:

$$\frac{\partial J}{\partial R} = \frac{\partial D}{\partial R} + \lambda = 0. \tag{3.1}$$

The Lagrange multiplier λ can be derived as

$$\lambda = -\frac{\partial D}{\partial R}. \tag{3.2}$$

In order to determine λ, the RD curve of the video to be encoded should be known, for which two-pass encoding is needed. In the first pass, the RD characteristics are computed, as illustrated in Figure 3.2. In the second pass, the data is used to achieved the best encoding quality and fit the bit rate budget.

The RD hyperbolic model [72] has been used for HEVC and expresses

$$D = a \cdot R^{-b}, \tag{3.3}$$

in which case λ can be estimated as

$$\lambda = ab \cdot R^{-b-1} \tag{3.4}$$

Therefore, having accurate predictions of both R and D can provide early data on the effects of an encoder as well as be helpful for rate control.

3.1.2 Modelling RD relationship

CNNs have been previously used to model the relationship between rate and distortion in HEVC. For instance, in [154] both rate R and distortion are estimated in order to establish a relationship between bits per pixel (bpp) and the SSIM as

$$\mathrm{SSIM} = a \cdot R^b. \tag{3.5}$$

The rate and distortion are predicted using two CNNs. The rate CNN takes an original frame and predicts a set of rate values, measured at different QPs. On the other hand, the distortion CNN takes an original frame and predicts a single SSIM map, from where the mean SSIM can be obtained. Hence, the distortion CNN needs to be trained as many times as QPs are considered. Both CNNs have a U-Net [116] structure, with down-sampling and up-sampling layers.

This CNN framework was used as baseline for the work proposed in this chapter. As opposite to SSIM, most video encoders rely on MSE based distortions to select the best coding tools. Additionally, due to the non-linearity of several of the encoder blocks, using only linear activations may not be enough to provide accurate estimates. As such, the approach proposed here is different from the baseline in that can predict MSE distortions instead of SSIM, and makes use of non-linear activation functions. Moreover, a second CNN is used to estimate average distortions for the whole frame, referred to as global distortions, and number of bits for a variety of QPs, in a single pass.

3.2 Estimation of rate and distortion

This work is developed in the context of HM, which employs MSE based distortions for RDO. Additionally, HM utilises the PSNR to measure the video quality. The PSNR, which is part of the Video Quality Metric (VQM) software [54] developed by the Institute for Telecommunication Sciences (ITS), can be computed from the MSE.

For this reason, the distortions predicted with the proposed framework are MSE related. While perceptual distortion/quality metrics, such as the SSIM and the Video Multimethod Assessment Fusion (VMAF) [82], could be considered as well, they are out of the scope of this research since they are not used for decision making by the default HM.

The estimation of distortion maps was performed using a CNN with two inputs. The first input is the original luma frame $\boldsymbol{M} \in \mathbb{R}^{H \times W}$, which is normalised as follows

$$\hat{M}_{i,j} = \frac{M_{i,j}}{2^{B-1}}, \tag{3.6}$$

where B is the bit depth. The second input is a normalised QP map $\hat{\boldsymbol{Q}} \in \mathbb{R}^{H \times W}$

$$\hat{Q}_{i,j} = \frac{\mathrm{QP}}{\mathrm{QP}_{max}}, \tag{3.7}$$

where QP_{max} is the maximum QP supported.

For the training, a set of ground truth distortion maps $\boldsymbol{D}$ were used, namely sample-wise maps of absolute differences between the original and reconstructed frame. The goal of the network is to estimate the distortion map $\boldsymbol{N} = G(\hat{\boldsymbol{M}}, \hat{\boldsymbol{Q}}) \approx \boldsymbol{D}$. As shown in Figure 3.3, G is an CNN formed of residual connections, convolutions, non-linear mapping, down-sampling, up-sampling and skip connections. CNN G initially learns the differences between inputs and outputs, where such difference is modelled in the last layer as an element-wise summation between the output of the previous layer and $\hat{\boldsymbol{M}}$. Secondly, convolution layers use a stride of 1 × 1, and kernel sizes of 3 × 3, except the final layer which uses a 5 × 5 filter. Thirdly, non-linear mapping is achieved by adding PReLU (Eq. 2.17) after each convolution layer, which increases the flexibility of the network. Max pooling layers adopt a filter size of 2 × 2, the stride is 1 × 1 and the output represents one quarter of the input. Up-sampling layers balance the size reduction introduced by max pooling layers. Finally, skip connections serve to aggregate multi-level features, which are modelled by concatenating the features learned in the 2nd and 4th convolution layers with features learned in 9th and 7th convolution layers, respectively.

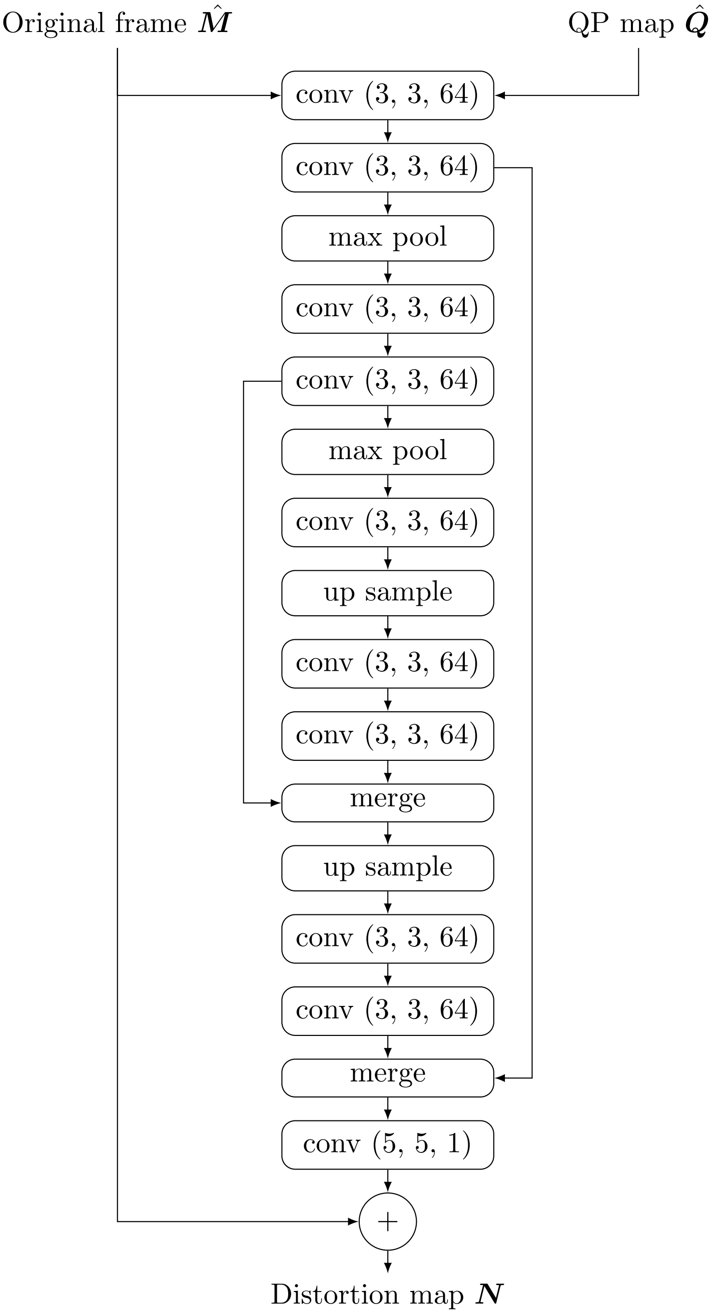


Figure 3.3: CNN G that predicts distortion maps. All convolution layers, except the last one, use PReLU activation function.

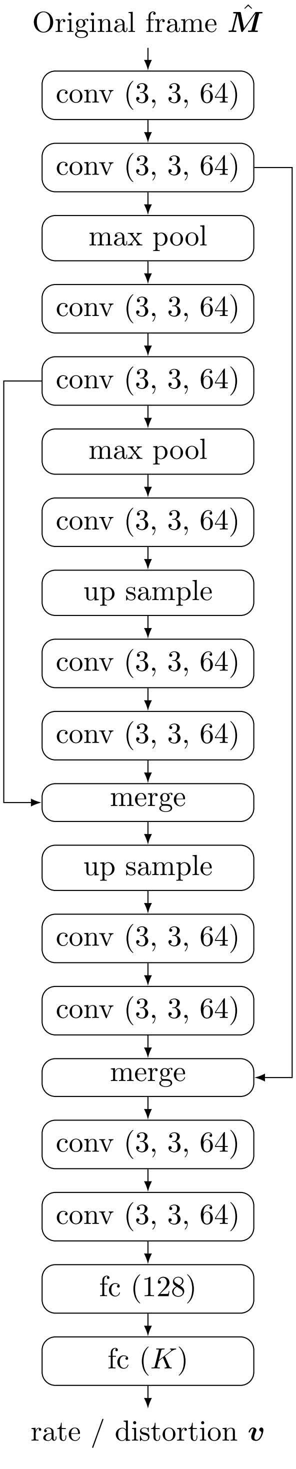


Figure 3.4: CNN F. All convolution layers and the first FC layer use ReLU activation function.

The loss function $\mathcal{L}_G$ used for training is the MSE (Eq. 2.2).

An additional CNN was modelled to produce the estimate of the rate, measured as the number of bits, obtained when intra coding a frame with HEVC. The CNN takes as input $\hat{\boldsymbol{M}}$, and is given ground truths in the form of a vector of scalars $\boldsymbol{v}$, where each element is the number of bits necessary to encode the frame with a certain QP. A total K QPs are considered. The goal is to estimate the vector $\boldsymbol{p} = F(\hat{\boldsymbol{M}}) \approx \boldsymbol{v}$. As shown in Figure 3.4, the CNN F is similar to the CNN G. Nevertheless, the CNN F uses FC layers that extract meaningful data from features and from those predict $\boldsymbol{v}$. All convolution layers as well as the first FC layer are activated using ReLU (Eq. 2.16).

The loss function $\mathcal{L}_F$ is the Mean Absolute Error (MAE)

$$\mathcal{L}_F = \frac{\sum_{i=0}^{K-1} |v_i - p_i)|}{K}. \tag{3.8}$$

In addition to being used for predicting the number of bits, the CNN F is also trained to predict global average distortions. In this case, each element in the ground truths $\boldsymbol{v}$ is the mean of the distortion map between original and reconstructed frame, as obtained when encoding with a given QP value.

3.2.1 Training methodology

COCO 2017 datasets [85] are used for running the experiments. Figure 3.5 shows sample frames. 20,000 frames are used for training, 5,000 for validation and 20,000 for testing. All frames are cropped into 128 × 128 patches and converted to 8 bit YUV with 4:2:0 chroma subsampling. Each frame is encoded with HM 16.9 [117] using four different QPs, namely 22, 27, 32 and 37, where the QP$_{max}$ is 51. The QPs selection is done according to the HM Common Test Conditions (CTC) document [10].

The proposed CNNs are implemented in TensorFlow [1] 1.14 and the source code was released under Apache 2.0 Licence [119].

Figure 3.5: COCO 2017 dataset [85]. Frames are cropped to the same size for visualisation purposes.

The training uses a batch size of 32, Adam optimiser [63], a learning rate of 1×10^{-4} and early stopping. In particular, the training was stopped in case the validation loss did not result in any improvement after 10 consecutive epochs.

The base CNNs are implemented using the description provided in [154], which indicates the usage of linear activations, Adam optimiser and learning rate of 1×10^{-3}. The training also considers a batch size of 32 and the same stop condition mentioned previously. Additionally, the distortion is computed as the sample-wise map of absolute differences, instead of SSIM, between original and reconstructed frames.

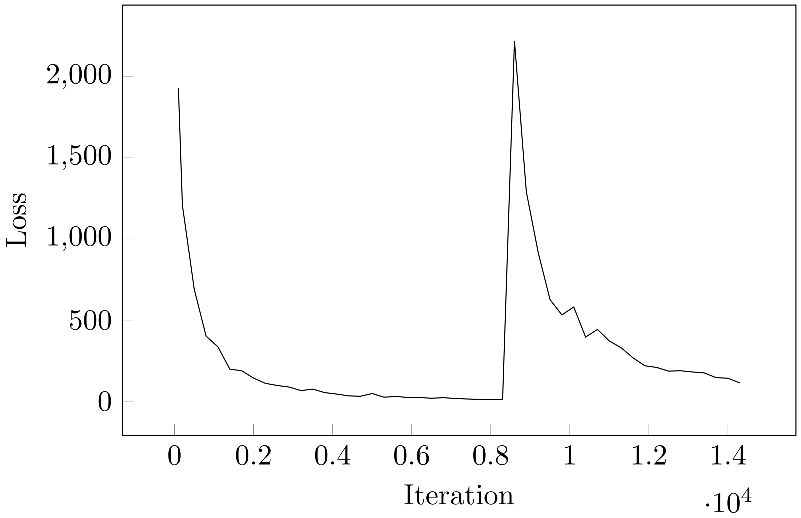


Figure 3.6: Training loss for base rate CNN. The gradients explosion may be due to the training dataset not being large enough or the variable updates using a learning rate toohigh.

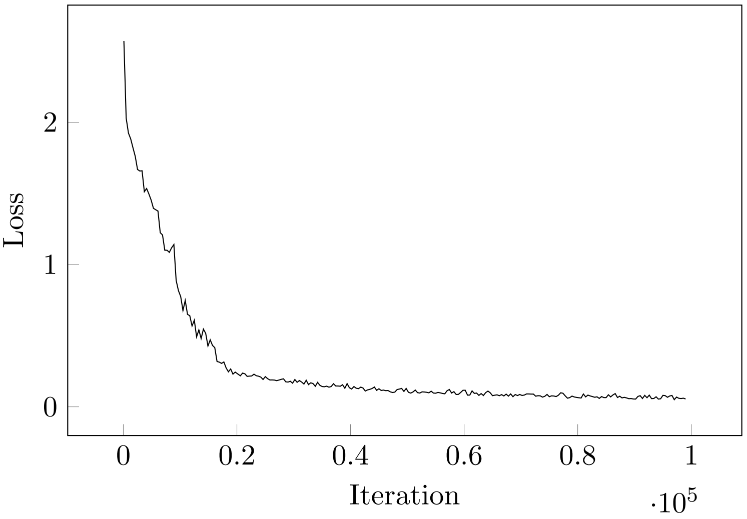


Figure 3.7: Training loss for CNN F.

The training of the base CNNs showed exploding gradients, as illustrated in Figure 3.6. This behaviour may be due to several factors, including the training dataset not being large enough or the variable updates using a learning rate too high. The proposed CNNs solve this issue by means of adding an ℓ_2 weight decay with 1×10^{-4}. For comparison, the loss during training for CNN F is displayed in Figure 3.7.

3.3 Derivation of rate-distortion parameters

The proposed CNNs estimate a set of points of rate and distortion from original content, avoiding the multi-pass encoding. These predictions can be utilised to find the parameters of the RD model by linearising Eq. 3.3 as

$$\ln D = \ln a - b \cdot \ln R \tag{3.9}$$

and using least squares (Figure 3.8). Let $\hat{D} = \ln D$, $\hat{a} = \ln a$ and $\hat{R} = \ln R$

$$\hat{a} = \frac{\sum_i \hat{D}_i \sum_i \hat{R}_i^2 - \sum_i \hat{R}_i \sum_i \hat{R}_i \hat{D}_i}{N^2 \sum_i \hat{R}_i^2 - \left(\sum_i \hat{R}_i\right)^2}, \tag{3.10}$$

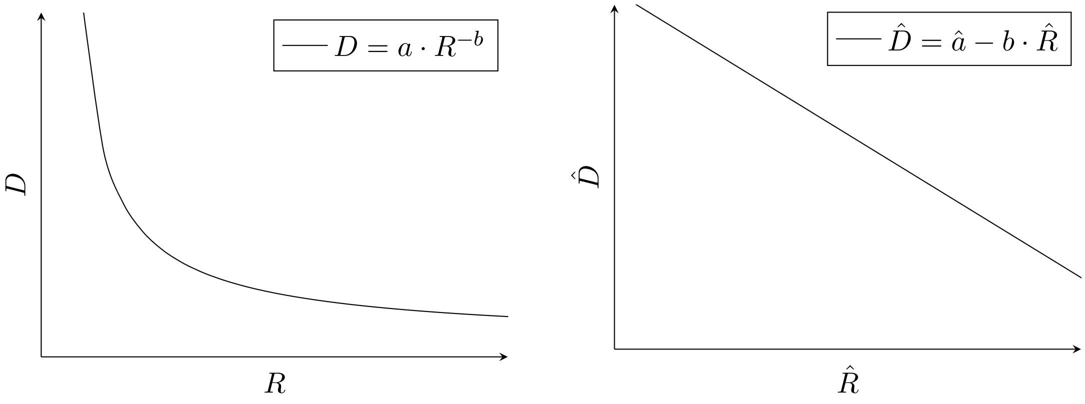
$$b = \frac{N \sum_i \hat{R}_i \hat{D}_i - \sum_i \hat{R}_i \sum_i \hat{D}_i}{N \sum_i \hat{R}_i^2 - \left(\sum_i \hat{R}_i\right)^2}, \tag{3.11}$$

the RD model can be expressed as

$$D = \exp^{\hat{a}} \cdot R^{-b} \tag{3.12}$$

and, finally, the Lagrangian multiplier λ is given by

$$\lambda = \exp^{\hat{a}} \cdot b \cdot R^{-b-1}. \tag{3.13}$$



(a) Hyperbolic function.

(b) Linearisation.

Figure 3.8: RD model.

3.4 Experiments

This section presents the evaluation of the proposed CNNs and the base CNNs, which are deployed on an NVIDIA GeForce GTX 1080 GPU. First, the two approaches are compared. Later, the predictions obtained with the proposed CNNs are used to estimate the RD model and its parameters.

3.4.1 Evaluation metrics

The evaluation metrics include the Pearson Correlation Coefficient (PCC), defined as

$$r = \frac{\sum x_i y_i - n\bar{x}\bar{y}}{\sqrt{\left(\sum {x_i}^2 - n\bar{x}^2\right)}\sqrt{\left(\sum {y_i}^2 - n\bar{y}^2\right)}}, \tag{3.14}$$

the area between two curves and the discrete Fréchet distance [30], which is the shortest distance between two curves.

Let S and U be two curves of length L_S and L_U that can be modelled as functions of t using $\alpha(t)$ and $\beta(t)$, such that $\alpha(0) = 0$, $\alpha(1) = L_S$, $\beta(0) = 0$ and $\beta(1) = L_U$, the Fréchet distance is given by

$$\mathcal{F}(S,U) = \inf_{\alpha,\beta} \max_{t\in[0,1]} \left\{ \delta\big(S(\alpha(t)), U(\beta(t))\big) \right\} \tag{3.15}$$

where δ is the Euclidean distance:

$$\delta(X,Y) = \left\{ \sum_{i=0}^{N-1} |x_i - y_i|^2 \right\}^{\frac{1}{2}}. \tag{3.16}$$

In an ideal scenario, the ground truth curve and the predicted curve would match perfectly and the Fréchet distance and area between curves would be zero.

3.4.2 Predicted distortion maps

The accuracy of predicted distortion maps is measured in terms of local correlation. Distortion maps (element-wise absolute differences) are split into blocks of the same size and the MSE is computed per block. For each frame, these values are arranged in two vectors: one for ground truths and one for predictions, following a Z-order as in Figure 3.9, which are then compared using the PCC.

Table 3.2 presents the average PCC values for the testing data, per QP and block size. For both, the CNN G and the base CNN, the lower the QP, the lower the correlation between ground truths and predictions. This indicates that the CNNs predict more easily higher distortions (obtained with high QPs). Moreover, higher correlations are obtained when considering larger block sizes, which shows that the evaluated models are more suitable to predict global trends.

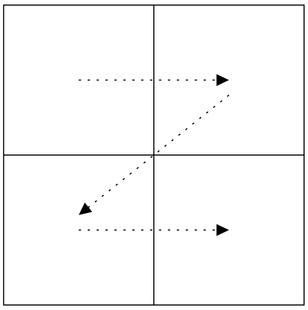
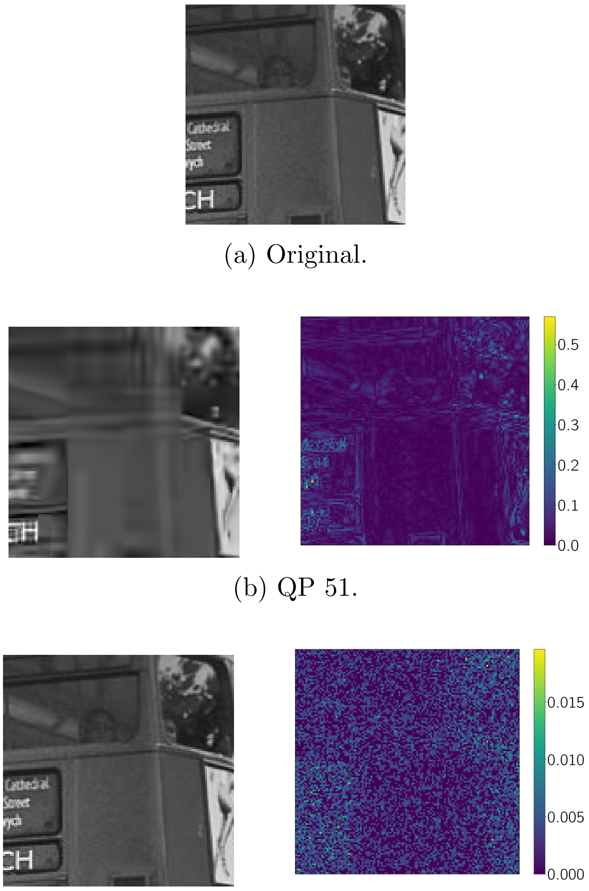


Figure 3.9: Traversing block sub-partitions in Z-order.

Table 3.2: Local correlation of predicted distortion maps.

Model	Block size	PCC			
		QP 22	QP 27	QP 32	QP 37
Base CNN	64×64	0.58	0.57	0.79	0.86
	32×32	0.54	0.59	0.71	0.79
	16×16	0.48	0.56	0.66	0.76
	8×8	0.41	0.51	0.62	0.72
CNN G	64×64	0.51	0.78	0.89	0.92
	32×32	0.53	0.79	0.90	0.92
	16×16	0.54	0.78	0.88	0.90
	8×8	0.54	0.76	0.85	0.87



(a) Original.

(b) QP 51.

(c) QP 22.

Figure 3.10: Reconstructed frames and distortion maps (ground truths).

Such behaviour is due to three factors: the quantisation component of the encoding process, the texture of the content and the fact that a distorion map is predicted as a pixel-wise difference.

The quantisation function transforms a value from a bigger domain to a smaller domain by mapping close values to the same one. In that way, by using high QPs the quantisation attenuates textured content and contributes in producing clusters of smooth reconstructed content. This also means that similar neighbouring pixels would have low distortion and form clusters as well. Under these circumstances, the CNN G produces more accurate distortion maps since there are more smooth regions in them. This case is illustrated in Figure 3.10(b), which presents an original frame, a reconstructed frame for QP 51 and the corresponding distortion map.

On the other hand, the usage of low QPs preserves details of the original content. In this scenario, the distortion could be seen as noisy, since the differences are granular details spread across the frame, even in neighbouring pixels. This is shown in Figure 3.10(c) which displays an original frame, a reconstructed frame for QP 22 and the corresponding distortion map.

As previously mentioned, the CNN G produces more accurate distortion maps for high QPs. This tendency is confirmed through a visual comparison as exhibited in Figures 3.11 and 3.12.

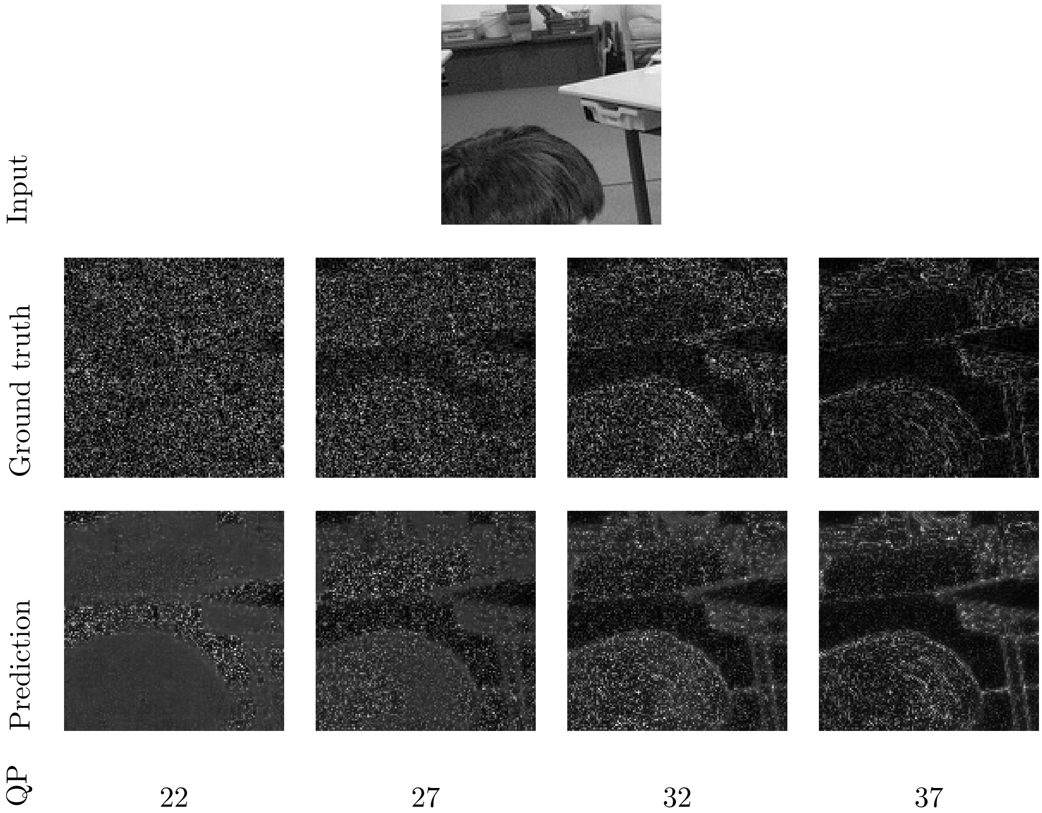


Figure 3.11: Comparison of distortion maps for sample 16. The predictions are obtained using the CNN G.

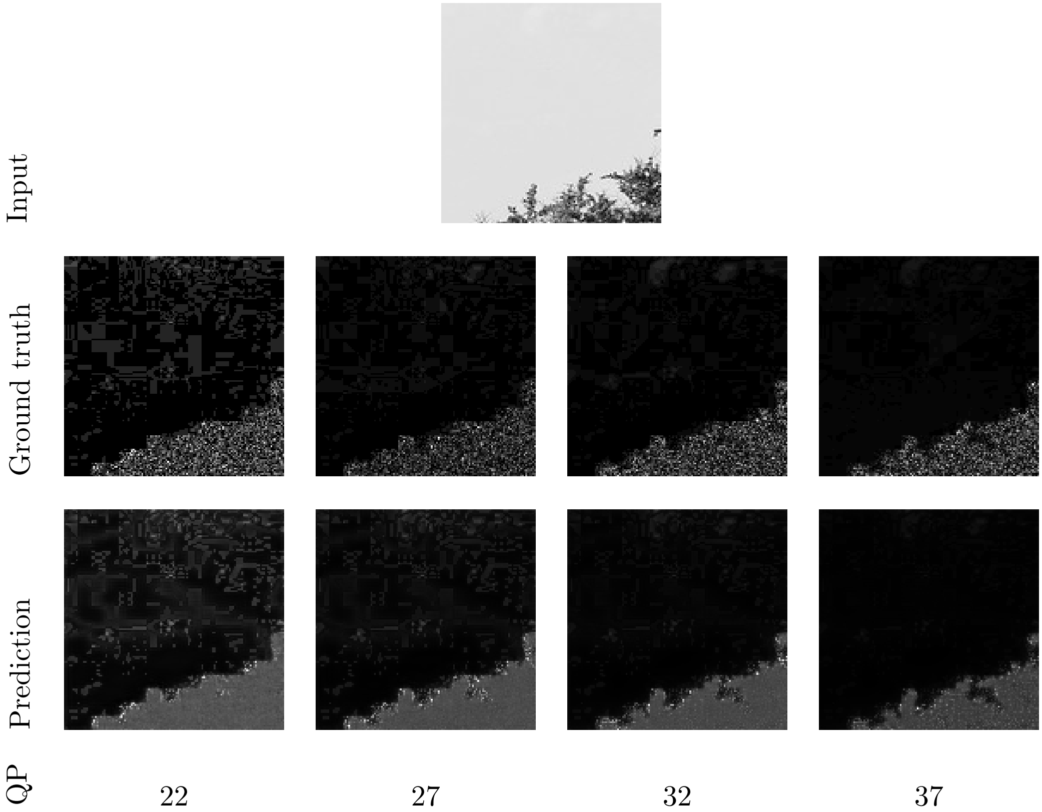


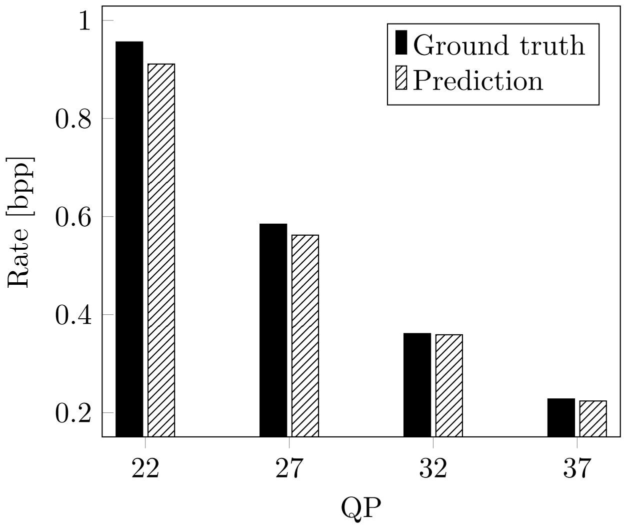
Figure 3.12: Comparison of distortion maps for sample 300. The predictions are obtained using the CNN G.

3.4.3 Predicted rates and global distortions

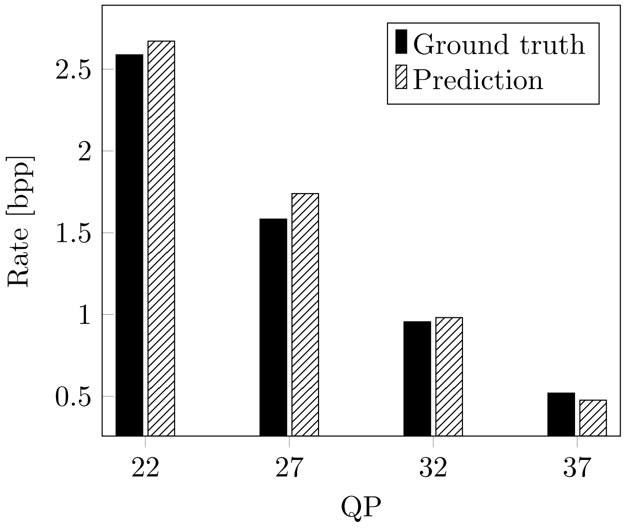
When considering predictions of bpp, results show that the proposed CNN F outperforms the base model since the MAE and Fréchet distance are lower, as shown in Table 3.3. Figure 3.13 displays bpp predictions, computed with the CNN F, per QP for two frames. It can be observed there is a strong correlation between ground truths and predictions, as well as low differences between them.

Table 3.3: Comparison of rates.

Model	Loss	Fréchet distance
Base CNN	10.45	19.00
CNN F	0.07	0.14



(a) Frame 6,677



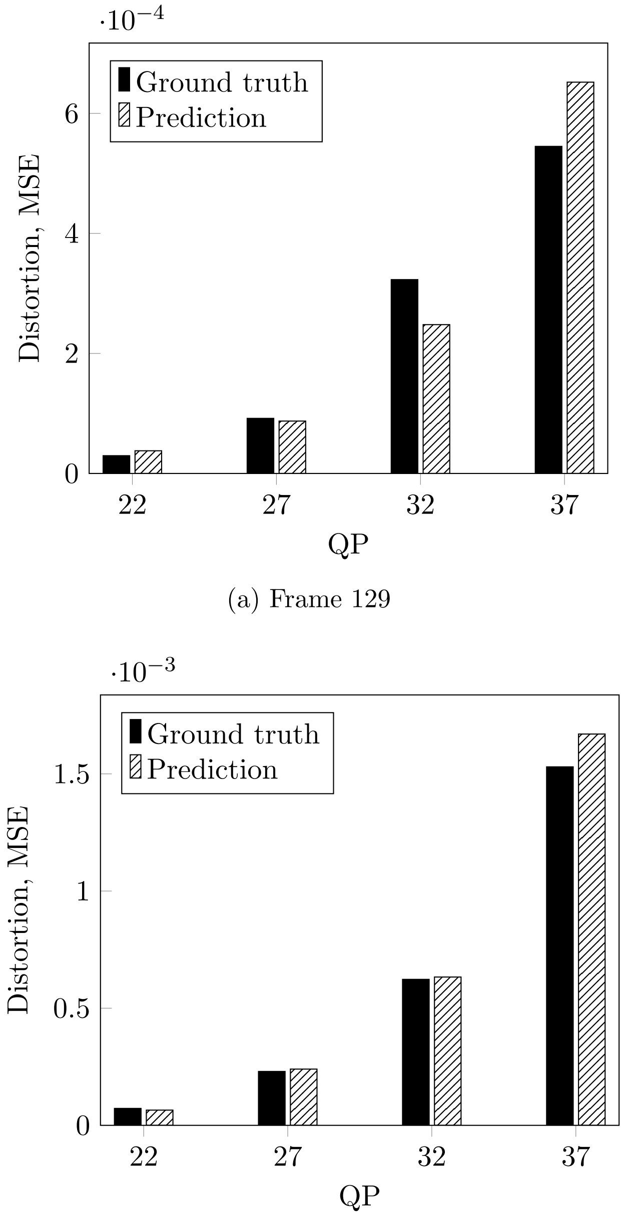
(b) Frame 19,995

Figure 3.13: Comparison of rate predictions obtained with CNN F.

Similarly, global distortions predicted with the proposed CNN F provide better results than the base CNNs, as shown in Table 3.4. The predictions for two different frames, using the testing QPs, are displayed in Figure 3.14. In this scenario too, there is a strong correlation between ground truths and predictions with low errors. Nevertheless, the CNN F works better for predicting the MSE than the bpp, since the MSE is computed over normalised frames, meaning the MSE range is normalised; whereas the bpp fall in an open range, making the CNN more difficult to converge.

Table 3.4: Comparison of global distortions.

Model	Loss	Fréchet distance
Base CNN	3.02×10^{-2}	1.51×10^{-2}
CNN F	1.42×10^{-4}	5.81×10^{-5}



(a) Frame 129

(b) Frame 16,676

Figure 3.14: Comparison of global distortion predictions obtained with CNN F.

3.4.4 Rate-distortion model

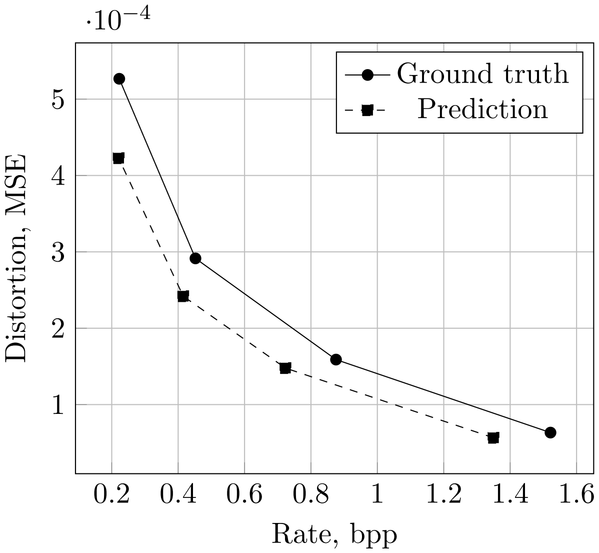
The rate and distortion predictions obtained with the CNN F are finally used to derive the RD model expressed by Eq. (3.3). The ground truth and predicted RD curves are compared using the Fréchet distance and the area between the curves. Table 3.5 presents the quantiles 1, 2, 3 and mean for both metrics. Moreover, Figure 3.15 offers a visual comparison of two pairs of RD curves (ground truths and predictions).

The results for the RD model indicate low error. From Figure 3.15 it can be concluded that the highest differences lie in the rate axis, as a consequence of the lower prediction accuracy obtained for bpp predictions than for MSE predictions. While Figure 3.15(a) shows consistency between the predicted RD model and the ground truth, Figure 3.15(b) is less precise as the predicted model behaves in different way than the ground truth. The reason for this is that the CNNs were trained independently, which could cause an inconsistent relationship between the rate and the distortion across a set of QPs.

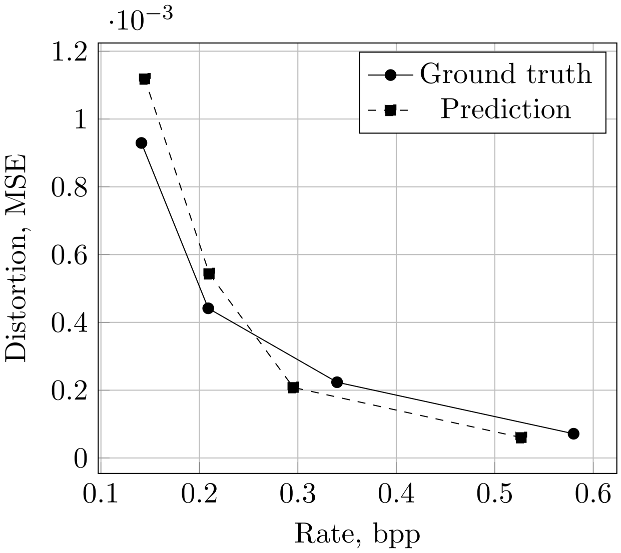
Taking this into account, the λ estimation is skipped as there is a significant difference in the prediction accuracy of the metrics required. It is perhaps better to find first a way to increase the prediction accuracy for the bpp predictions, before moving forward with estimations that depend on these.

Table 3.5: Comparison of RD models.

		Fréchet distance	Area between curves
RD model	Q1	5.01×10^{-2}	4.86×10^{-5}
	Q2	1.01×10^{-1}	1.55×10^{-4}
	Q3	1.85×10^{-1}	3.94×10^{-4}
	Mean	1.45×10^{-1}	6.45×10^{-4}



(a) Frame 7,984.



(b) Frame 19,863

Figure 3.15: Comparison of RD models.

3.5 Conclusion

This chapter presents a CNN-based methodology to predict the rate and distortion metrics obtained when intra coding original frames using different QPs. One CNN is used to predict local distortion, which provides a visualisation on how the reconstruction errors are distributed within a frame. A second CNN is used to predict a set of global distortions and rates from an original frame. All these predictions can be computed from the original data, meaning that it is possible to know such predictions before the actual encoding takes place, avoiding the multi-pass encoding.

Overall, in most cases the predictions are correlated to ground truth values. For the local distortion predictions, it was found that the CNN performs the best in regions with plain textures and high QP values, as in such cases there are clusters of low distortions which are easier to predict. On the other hand, the distortion of content encoded with low QP values is more difficult to predict since the distortions are granular and spread across the whole frame.

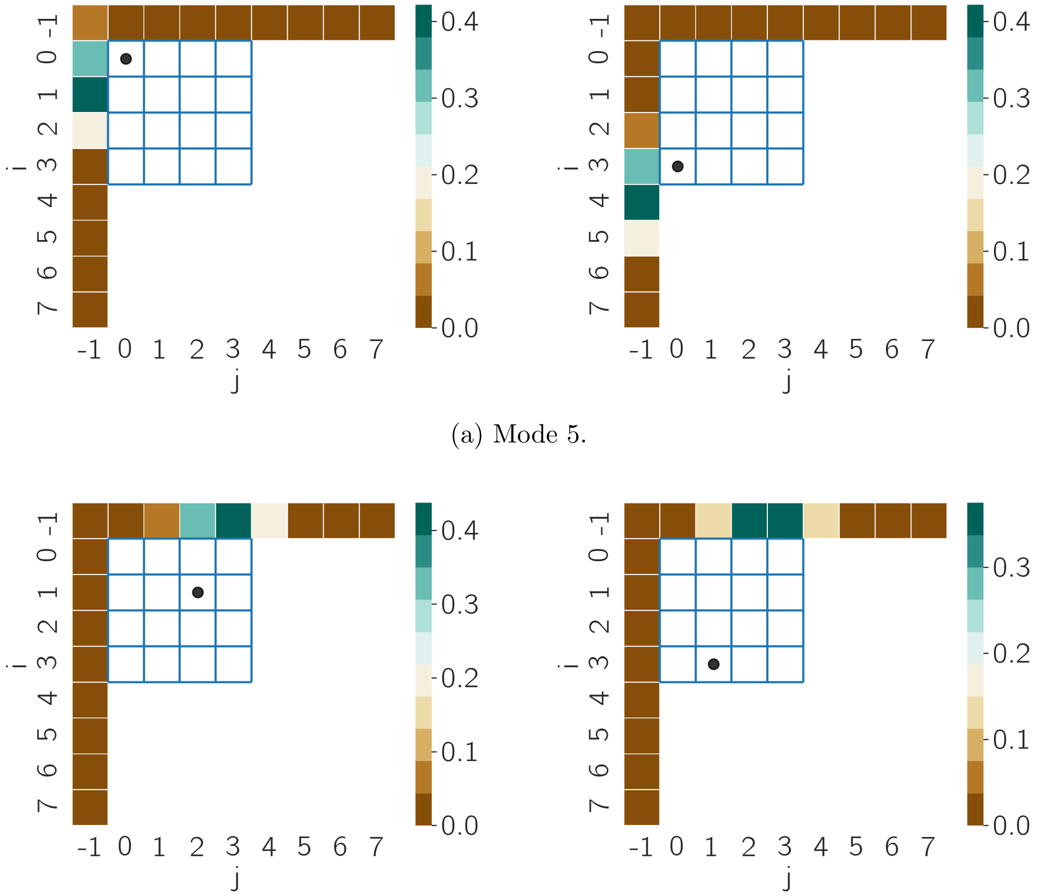
The CNN that predicts global metrics offers a better performance for distortion than for rate. Mainly due to the fact that global distortions are normalised, as the computation uses normalised frames too. In contrast, rate metrics are not normalised and this makes more difficult the training and convergence of the CNN. These results are promising, but since they affect the estimation of the RD model, there is room to improve the models before moving to estimations that depend on them.

Chapter 4

Functional interpretation of FCN for intra-prediction

Intra-prediction is essential to video compression. I frames are completely independent of previously encoded frames. They provide RAP to video streams and avert error propagation. Since I frames serve as references for P and B frames, by improving the quality of I frames, the performance of inter-prediction can be improved too. Traditional intra modes combine a limited number of reference samples to produce the prediction. For example, angular modes in VVC interpolate up to four reference samples, located in the same reference line (Figure 4.1), in order to compute a single prediction sample. Furthermore, angular prediction in VVC can be done using reference samples that are not necessarily located next to the current block.

This chapter presents the functional interpretation of FCN for intra-prediction, which aims to improve the coding efficiency of VVC at a lower computational cost than a "default" FCN. The learned intra modes can be used alongside traditional modes. Nonetheless, their signalling changes the syntax of VVC and, as a result, the bitstreams are not compliant with the standard. Two approaches to simplify FCN-based intra-prediction are proposed. The aim is to provide insight into how NN-based methods may generate their results to support the design of new intra modes.



(a) Mode 5.

(b) Mode 58.

Figure 4.1: Angular modes in VVC. The • marks the target sample. The reference samples are interpolated.

The rest of the chapter is organised as follows. Section 4.1 briefly describes interpretability in NNs. Section 4.2 introduces the FCN used as baseline in the analysis. Sections 4.3 and 4.4 present the proposed simplifications. Section 4.6 focuses on the training process. Section 4.8 reports the performance evaluation of the techniques proposed in this chapter, followed by conclusions in Section 4.9.

4.1 Motivation

NNs can detect complex patterns and have demonstrated great capabilities to predict information about data unknown to the model. Since most NNs remain black boxes, there is an increasing demand to understand the logic behind them, as well as to explain how predictions are made. In addition, create trustworthy learned-based systems, for which interpretability is key [64]. Interpretability can be intrinsic to the model, if the structure is simple enough (e.g. decision tree or sparse linear model). It can also be achieved by applying post hoc analysis to the trained model in order to explain the relationship between inputs and outputs [101, 104]. Moreover, interpretability can be used to identify redundancies within a learned model. The gained knowledge may support the design of simpler models.

The deployment of NN-based tools in video coding standards faces different challenges. For instance, the high computational complexity of NN solutions affects both encoding and decoding time, which may not be suitable for practical applications. Compression is applied to a wide variety of content yet learned parameters may not generalise well. Furthermore, the lack of predictability and transparency of some models makes their adoption more difficult, as standards prioritise explainable and well understood algorithms. In intra-prediction, an FCN can be used to describe a complex relationship between reference samples and prediction samples, which might instead be a linear or monotonous dependency [15]. Accordingly, it is possible to use post hoc analysis to simplify an FCN, as illustrated in Figure 4.2.

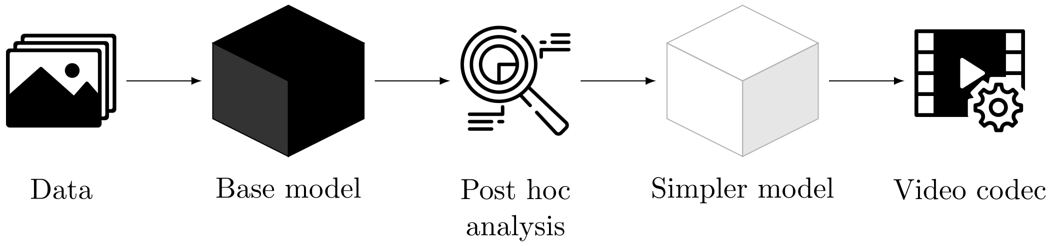


Figure 4.2: Interpreting an NN.

4.2 Background work

FCNs have been used as intra modes [47, 73, 74, 111] quite successfully. In particular, [111] demonstrated good performance within VVC when used alongside traditional intra modes. At the encoder side, the best mode is selected from a pool of traditional intra modes and FCN intra modes in an RD sense, as shown in Figure 4.3. Observe that traditional intra modes use one line of reference samples and the FCN intra modes use multiple reference lines instead.

The FCN in [111], from now on referred to as base FCN, exploits redundancies that may not be otherwise detected by traditional intra modes. Consequently, it is used as basis for the work described in this chapter, which focuses on predictions in the spatial domain.

Given a current block of size $H \times H$, represented as a vector $\boldsymbol{s} \in \mathbb{R}^N$ with $N = H^2$, the base FCN takes the reference samples $\boldsymbol{r} \in \mathbb{R}^M$ as input and produces K different predictions; $M = I(2H + I)$ and I is the number of reference lines.

The network consists of four layers and the first three are defined as

$$\boldsymbol{l}_i = \begin{cases} \rho(\boldsymbol{W}_i \times \boldsymbol{r} + \boldsymbol{b}_i) & \text{if } i = 0 \\ \rho(\boldsymbol{W}_i \times \boldsymbol{l}_{i-1} + \boldsymbol{b}_i) & \text{if } 1 \leq i < 3; \end{cases} \tag{4.1}$$

where $\boldsymbol{W}_1, \boldsymbol{W}_2 \in \mathbb{R}^{M \times M}$, $\boldsymbol{W}_3 \in \mathbb{R}^{Q \times M}$, $\boldsymbol{b}_1, \boldsymbol{b}_2 \in \mathbb{R}^M$, $\boldsymbol{b}_3 \in \mathbb{R}^Q$. M is the number of reference samples, Q is a reduced cardinality that depends on H and ρ is the ELU described in Eq. (2.18).

K modes are considered and each mode is defined with three common layers and a mode-specific final layer. For each k-th mode, $0 \leq k < K$, the prediction $\boldsymbol{p}_k$ is given by

$$\boldsymbol{p}_k = \boldsymbol{l}_4^{(k)} = \boldsymbol{W}_4^{(k)} \times \boldsymbol{l}_3 + \boldsymbol{b}_4^{(k)}, \tag{4.2}$$

where $\boldsymbol{W}_4^{(k)} \in \mathbb{R}^{N \times Q}$, $\boldsymbol{b}_4^{(k)} \in \mathbb{R}^N$.

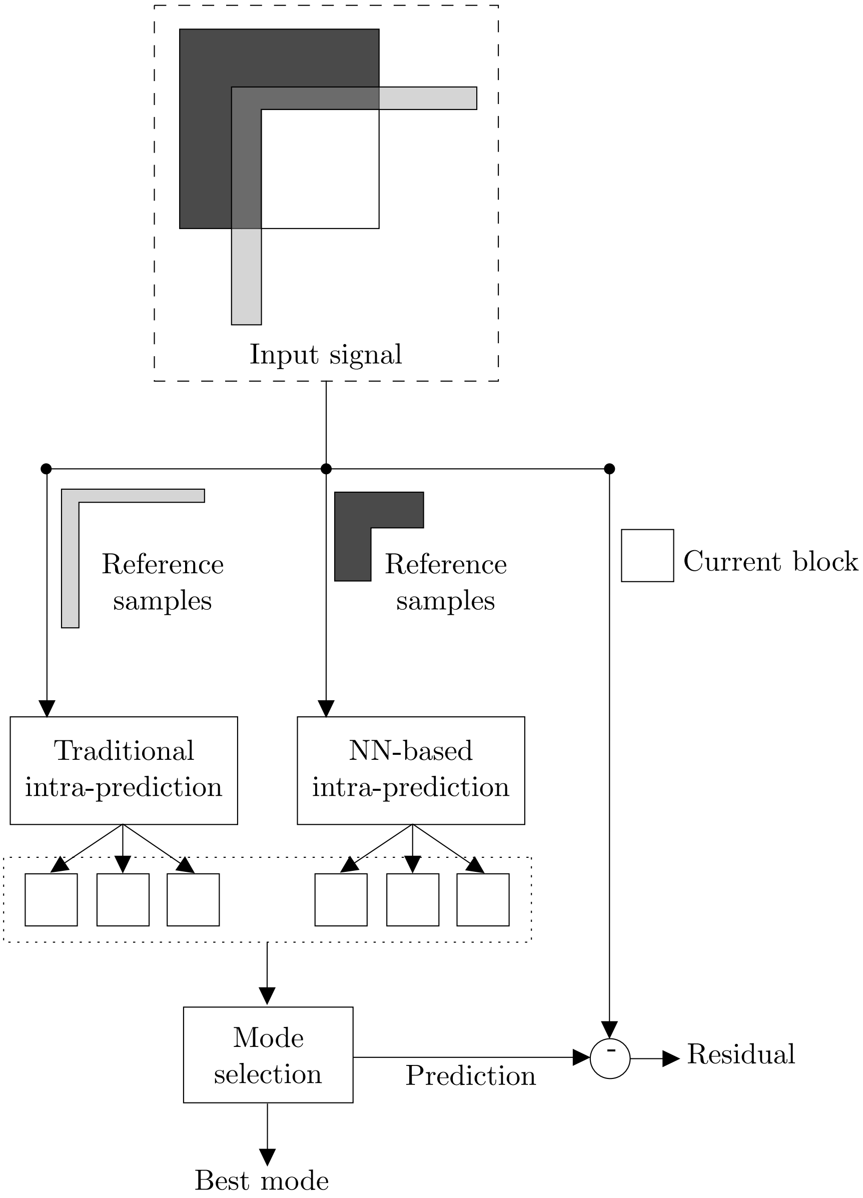


Figure 4.3: Selection of best intra mode from a pool of traditional and NN modes. Transparent light-gray samples are used as reference samples for traditional intra modes. Dark-gray samples are used as reference samples for NN modes.

Figure 4.4 depicts the structure of the base FCN, which performs matrix-vector operations. For visualisation purposes, the vectors have been reshaped to follow either the spatial location of the reference samples or the shape of the prediction block.

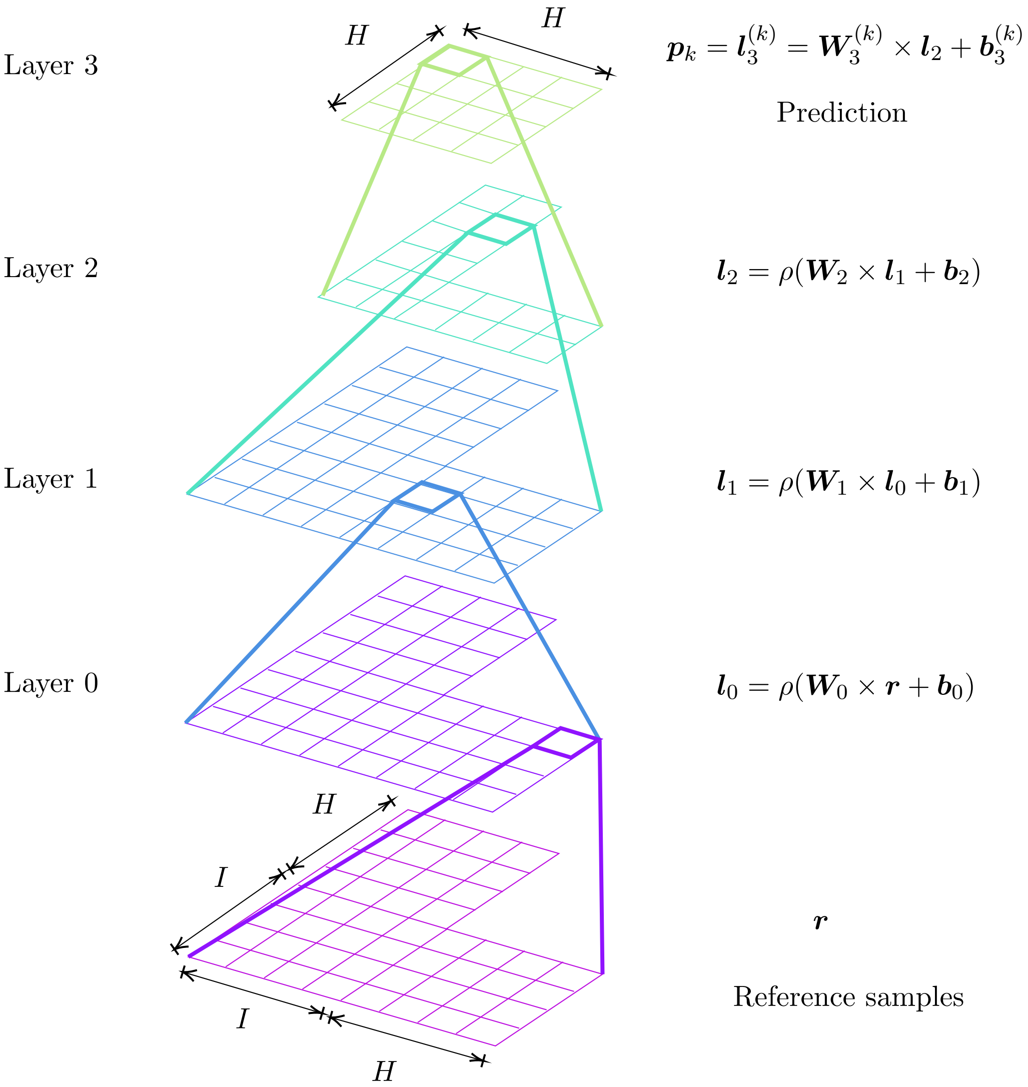

Layer 3
H
H
$\boldsymbol{p}_k = \boldsymbol{l}_3^{(k)} = \boldsymbol{W}_3^{(k)} \times \boldsymbol{l}_2 + \boldsymbol{b}_3^{(k)}$
Prediction
Layer 2
$\boldsymbol{l}_2 = \rho(\boldsymbol{W}_2 \times \boldsymbol{l}_1 + \boldsymbol{b}_2)$
Layer 1
$\boldsymbol{l}_1 = \rho(\boldsymbol{W}_1 \times \boldsymbol{l}_0 + \boldsymbol{b}_1)$
Layer 0
$\boldsymbol{l}_0 = \rho(\boldsymbol{W}_0 \times \boldsymbol{r} + \boldsymbol{b}_0)$
H
I
$\boldsymbol{r}$
Reference samples
I
H

Figure 4.4: Base FCN for intra-prediction. The network consists of four layers, the input is the reference samples and the output is the prediction. ρ is the ELU function.

As the activation functions are used throughout the FCN, the interpretability is limited due to the ELU-related non-linearities. Attempts to simplify the FCN are presented next.

The proposed solutions also address the challenge of the increased computational complexity of NN-based coding tools by reducing the number of operations required to compute a prediction sample. This is further explained in the following sections.

4.3 Linearising the base FCN

As mentioned previously, post hoc analysis can be used to gain insights on the relationship between the inputs and outputs of an NN. In this way, a linear FCN can be derived from the base FCN in order to define the contribution of each reference sample in computing prediction samples.

For each k-th mode a master matrix $\boldsymbol{V}^{(k)} \in \mathbb{R}^{N\times M}$ is computed from the weights of NN layers, Eqs. (4.1) and (4.2), as

$$\boldsymbol{V}^{(k)} = \boldsymbol{W}_4^{(k)} \times \boldsymbol{W}_3 \times \boldsymbol{W}_2 \times \boldsymbol{W}_1. \tag{4.3}$$

The coefficients in $\boldsymbol{V}^{(k)}$ are row-wise normalised, and the final predictor is given by

$$\boldsymbol{U}^{(k)} = \left(\frac{V_{i,j}^{(k)}}{\sum_{h=0}^{M-1} V_{i,h}^{(k)}} \right). \tag{4.4}$$

Due to the removal of activation functions and bias, the coefficients are normalised to make sure the final prediction samples assume values with energies comparable to the target block. Hence, the prediction $\hat{\boldsymbol{p}}_k$ is obtained as:

$$\hat{\boldsymbol{p}}_k = \boldsymbol{U}^{(k)} \times \boldsymbol{r}. \tag{4.5}$$

In this approach, each k-th FCN-based predictor (Figure 4.4) was simplified as a matrix to estimate a target block from a set of reference samples, as shown in Figure 4.5. The same number of modes K as in the base FCN was considered.

Even though this approach achieved coding gains at a lower computational complexity (Subsection 4.8.3), it is an imperfect representation of the relationships learned by the base FCN. This was observed by comparing the coding gains achieved by both solutions. Nonetheless, a benefit is that such a simple FCN enables a clear indication on how each reference sample can contribute in producing a prediction sample.

4.4 Defining a linear FCN

The aforementioned FCN showed that at some extent a linear FCN can be used as new intra modes. For that reason, an intrinsic FCN is defined following the structure of the base FCN (Figure 4.4). In this case, all activations are linear. As such, a single layer FCN can be derived as

$$\check{\boldsymbol{p}}_k = \boldsymbol{V}^{(k)} \times \boldsymbol{r} + \boldsymbol{t}^{(k)}, \tag{4.6}$$

where $\boldsymbol{t}^{(k)} \in \mathbb{R}^N$ is calculated as:

$$\boldsymbol{t}^{(k)} = \boldsymbol{W}_4^{(k)} \times (\boldsymbol{W}_3 \times (\boldsymbol{W}_2 \times \boldsymbol{b}_1 + \boldsymbol{b}_2) + \boldsymbol{b}_3) + \boldsymbol{b}_4^{(k)}. \tag{4.7}$$

This model, illustrated by Figure 4.5, makes use of the non-normalised matrix as defined in the previous subsection. Furthermore, this simplification is trained from scratch. Again, the same number of modes K was considered in this case.

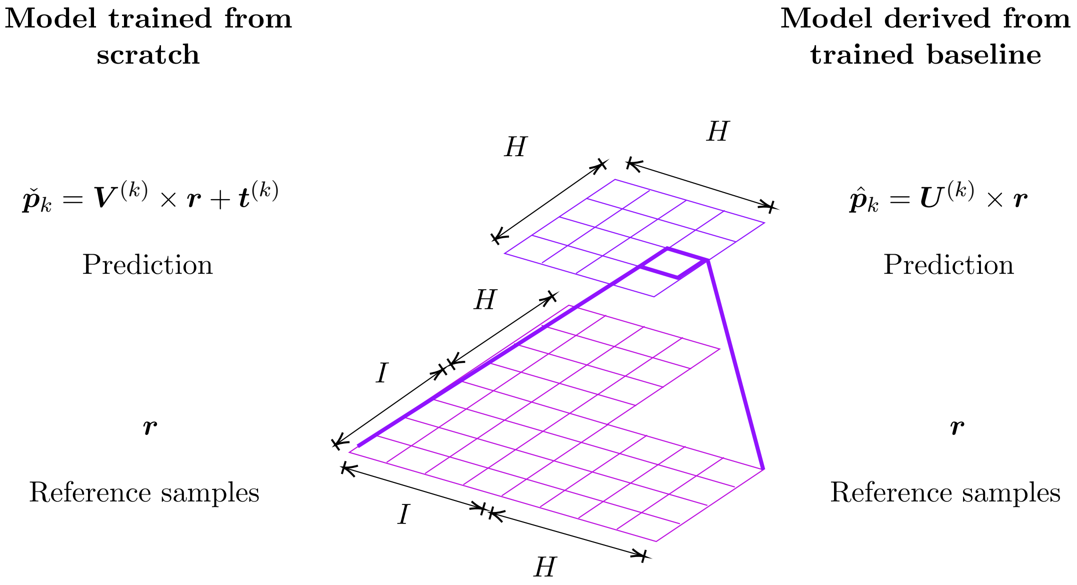

Model trained from scratch
Model derived from trained baseline
$\check{\boldsymbol{p}}_k = \boldsymbol{V}^{(k)} \times \boldsymbol{r} + \boldsymbol{t}^{(k)}$
Prediction
$\hat{\boldsymbol{p}}_k = \boldsymbol{U}^{(k)} \times \boldsymbol{r}$
Prediction
$\boldsymbol{r}$
Reference samples
$\boldsymbol{r}$
Reference samples
H
H
H
I
I
H

Figure 4.5: Simple intra-prediction models. The model on the left is the simplification trained from scratch. The model on the right is the simplification derived from the learned parameters of the base FCN.

4.5 Network pruning

A network can be further simplified and compressed by selectively pruning the learned weights [70]. By zeroing out network weights is possible to reduce the size of the network as well as to take advantage of sparse matrix storage and arithmetic.

A simple pruning strategy removes weights with small magnitude, which may have the least effect in the prediction error [70]. Under this strategy the network is first trained to convergence. Afterwards, each neuron is assigned a score. Next, the scores are used to selectively remove the least important neurons, namely those with absolute value below a given threshold. Finally, fine-tuning is carried out to compensate the decrease in the accuracy. Last but not least, pruning is commonly an iterative process that progressively removes neurons from the network, as depicted in Figure 4.6.

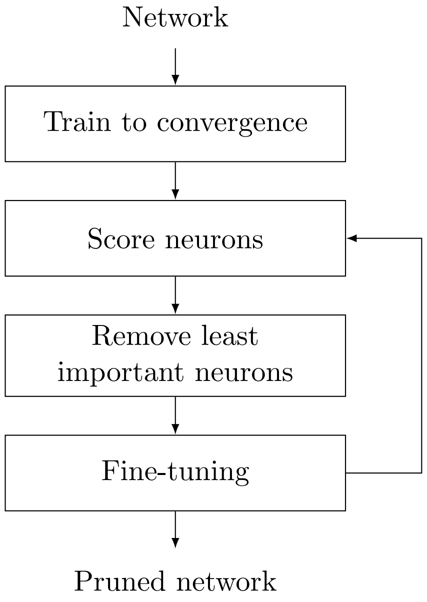


Figure 4.6: Network pruning.

4.6 Training methodology

DIVerse 2K (DIV2K) dataset [4, 136] is used for training purposes. DIV2K consists of HR frames and their corresponding LR frames, obtained with bicubic and an unknown down-sampling operators using three different factors: x2, x3 and x4. Figure 4.7 shows a sample frame in HR and the corresponding LR frames. All frames are converted to 10 bit YCbCr with 4:2:0 chroma subsampling.

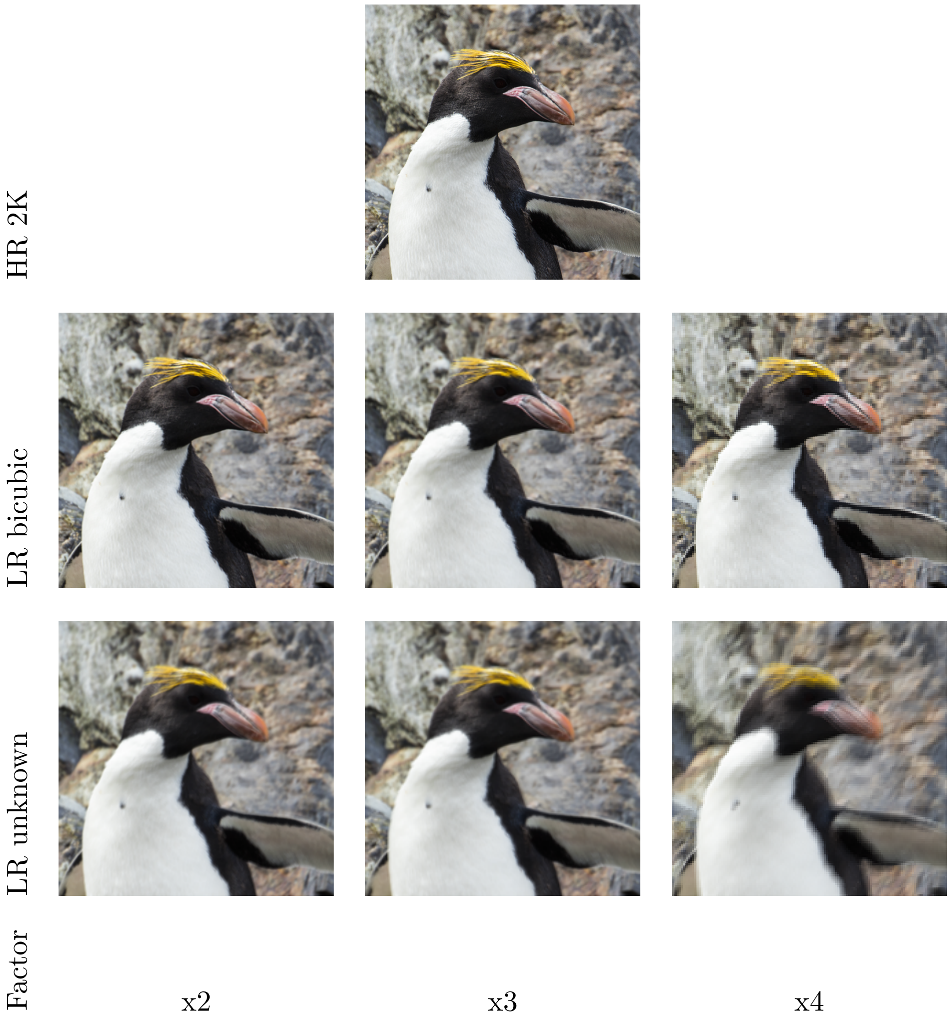


Figure 4.7: DIV2K dataset [4, 136]. The HR frames are 2K, the LR frames are downscaled with 3 different factors and using bicubic and an unknown downgrading operators.

Particularly, the networks are trained for 4 × 4, 8 × 8 and 16 × 16 blocks using 25,000 randomly selected luma patches. Both inputs $\mathbf{r}$ and ground truths $\mathbf{s}$ are normalised by

$$\mathcal{N}(\mathbf{y})_i = \frac{y_i}{2^{B-1}} - 1, \tag{4.8}$$

where B is the bit depth. Four lines of reference samples are considered and $K = 35$ modes are used.

4.6.1 Optimisation

As mentioned in Chapter 2, encoder decisions are made based on the RD cost. This requires passing through the whole encoder including the quantisation, which is typically a non-differenciable operation. Hence, the RD cost cannot be used as loss function. In the context of intra-prediction, video encoder reference software commonly uses less complex metrics to measure the distortion [18], e.g. SATD (see Eq. 2.4), in early steps, whilst the actual RD cost is used in final steps. In other words, a subset of intra-prediction candidates is selected by minimising the SATD, and the actual mode is selected among these candidates by minimising the RD cost. Taking this into account and that SATD estimates the effort needed to code the residual [151], the networks are trained using the SATD as loss function between a current block of samples $\mathbf{s}$ and a prediction $\mathbf{p}_k$.

The training is done as in [47] with some modifications. The overall loss function of a block minimises the loss of predicting a block as a whole or splitting it into four blocks of the same size, recursively, until the minimum supported size is reached. The best mode k^* is obtained as

$$k^* = \arg\min_k \mathcal{L}(\mathbf{s}, \mathbf{p}_k). \tag{4.9}$$

The overall loss is given by

$$\mathcal{L}^* = \min \left\{ \left\{ \sum_i \mathcal{L}\left(\mathbf{s}^{(j)}, \mathbf{p}_{k^*}^{(j)}\right) \right\} \cup \left\{\mathcal{L}(\mathbf{s}, \mathbf{p}_{k^*})\right\} \right\}, \tag{4.10}$$

where $\mathbf{s}^{(j)}$ denotes a sub partition of $\mathbf{s}$.

The usage of min and arg min functions enable mode specialisation since only the best modes will have gradient non-zero, and the parameters will be updated according to how suitable they are for certain kind of content. Nonetheless, using such optimisation from the start could mean that only a few modes and block sizes are always picked. This issue is tackled with a pre-training stage where three different functions are optimised and block splitting is not considered, meaning the modes are trained for each block size independently. The first function optimised is the average loss and the parameters are to be updated equally:

$$\bar{\mathcal{L}} = \frac{\sum_k \mathcal{L}(\mathbf{s}, \mathbf{p}_k)}{K}. \tag{4.11}$$

The second optimisation is the weighted sum of losses. The parameters are updated according to how good they are at predicting the block

$$\hat{\mathcal{L}} = \sum_k \left\{ \mathcal{L}(\mathbf{s}, \mathbf{p}_k) \odot \sigma(-\mathcal{L}(\mathbf{s}, \mathbf{p}_k)) \right\}, \tag{4.12}$$

where $\odot$ represents the element-wise product and σ is the softmax function [40] that gives the probability of a mode to be selected. The third optimisation is the minimum loss, meaning only the best mode is updated

$$\check{\mathcal{L}} = \min \left\{ \mathcal{L}(\mathbf{s}, \mathbf{p}_k \right\}. \tag{4.13}$$

The optimisation is done with Adam [63] optimiser, learning rate of 1×10^{-4} and ℓ_2 weight decay with 1×10^{-4}. The training also uses early termination with the condition of no improvement after 50 consecutive epochs. This stopping condition is also used to decide whether to continue pruning a network.

Finally, all trainings are performed using TensorFlow [1] 1.14 on a NVIDIA Titan X Pascal GPU[1].

4.7 Integration in the video codec

After the training is done, the parameters of the FCN modes are extracted from TensorFlow. Next, the modes are implemented on top of VVC Test Model (VTM) 1.0 [17] using C++ arithmetic operations. Moreover, the different FCN modes (base, linearised and linear) are implemented independently of each other.

The FCN modes are new intra modes that are incompatible with the VVC standard. They are used alongside traditional intra modes and, as a result, both the encoder and decoder are modified to be able to decode from the bitstream the intra mode to be used.

In particular, the FCN modes are signalled on two steps. First, a flag is transmitted at CU level to indicate whether the selected mode is an FCN mode. Next, let $x \in \mathbb{Z} \mid 0 \leq x < n$ be the mode index, a truncated binary code [110] is used (parametrised according to n). Let $k = \lfloor log_2(n) \rfloor \mid 2^k < n < 2^{k+1}$ and $u = 2^{k+1} - n$, with truncated binary encoding the first u mode indices use codewords of k bits and then remaining $n - u$ mode indices use last $n - u$ codewords of $k + 1$ bits. An example for $n = 5$ is shown in Table 4.1.

Table 4.1: Truncated binary encoding for $n = 5$.

Truncated binary	**Codeword**
0	0 0
1	0 1
2	1 0
3	1 1 0
4	1 1 1

[1] We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU used for this research.

4.8 Experiments

This section discusses the evaluation of the proposed intra modes. In order to make the results clear from now on $\boldsymbol{p}_k$ will be referred to as base modes, $\hat{\boldsymbol{p}}_k$ as linearised modes and $\check{\boldsymbol{p}}_k$ as linear modes. Subsection 4.8.1 compares the proposed modes, derived and linear, against the base FCN modes in terms of number parameters and computational complexity. Subsection 4.8.2 introduces the coding test conditions. Subsections 4.8.3 and 4.8.4 present the performance results for the linearised modes and the linear modes, respectively. Finally, Subsection 4.8.5 describes the performance of new linear modes resulting from changes to the design of the linear FCN.

4.8.1 Comparison of intra-prediction FCNs

The proposed intra modes, linearised and linear, reduce the number of parameters of the base modes in up to 87%, as shown in Table 4.2. A visual comparison is displayed in Figure 4.8.

The complexity of all FCN modes is measured in terms of number of multiplications needed to generate a prediction block (bias and activation function are not considered in this computation). The base modes require $4N(\sqrt{N} + 41) + 32(19\sqrt{N} + 18)$ multiplications, whereas the linearised and linear modes require $8N(\sqrt{N}+2)$ multiplications. Table 4.3 shows these values for the different block sizes supported.

Table 4.2: Number of parameters of FCN intra modes.

Block size	Base modes	Linearised modes	Linear modes	Reduction
4×4	6,020	768	784	87%
8×8	18,244	5,120	5,184	71%
16×16	69,284	36,864	37,120	46%

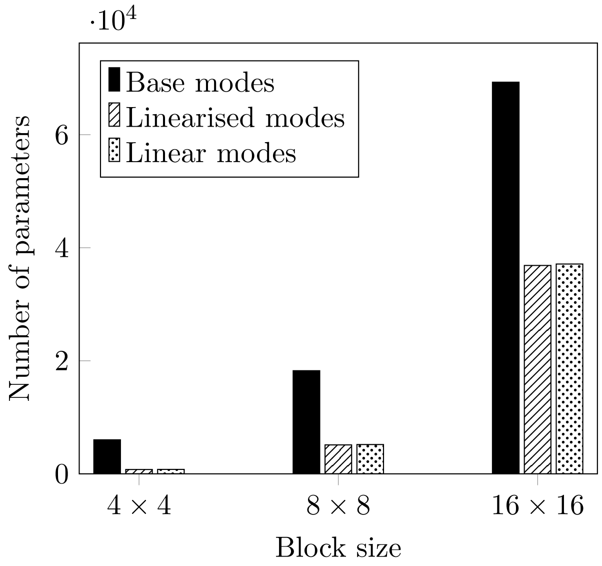


Figure 4.8: Number of parameters of FCN intra modes.

Table 4.3: Number of multiplications required to generate a block with FCN intra modes.

Block size	N	**Base modes**	**Simplified modes**	**Reduction**
4×4	16	5,888	768	87%
8×8	64	17,984	5,120	71%
16×16	256	68,672	36,864	46%

4.8.2 Coding test conditions

The coding tests are done under the AI configuration and the codec is constrained to utilise only squared CUs from 4×4 up to 16×16 (blocks for which the models are trained). Furthermore, the evaluated FCN modes are only applied to the luma component.

A variety of video sequences [10] are encoded and decoded with four QPs 22, 27, 32 and 37. In addition, the test utilised Queen Mary's Apocrita HPC facility, supported by QMUL Research-IT [62], and ran on Intel Xeon E5-2683V3 CPUs.

4.8.3 Linearised modes

The linearised modes are derived from the learned parameters of the base modes (Section 4.3). The performance of the linearised modes is presented in Table 4.4, which includes the BD-rate, encoding time (EncT) and decoding time (DecT) with respect to two different anchors: VTM 1.0 and VTM 1.0 enhanced with the base modes. Additionally, Figure 4.9 shows the percentage of blocks for all encoded sequences where the base modes and the linearised modes were selected by the encoder.

The results in Table 4.4 show that the linearised modes generate average coding gains of 0.78% BD-rate compared with VTM 1.0. Nonetheless, they achieve only 30% of the coding gains generated by the base modes. Further tests showed that predictions produced by the linearised modes are smoother than predictions generated by the base modes, which decreases the prediction accuracy for sharp content.

Table 4.4: Coding efficiency of linearised intra modes.

Class	**Sequence**	**Anchor: VTM 1.0**			**Anchor: base FCN [111]**		
		BD-rate Y	**EncT**	**DecT**	**BD-rate Y**	**EncT**	**DecT**
A	Traffic	-0.79%	211%	110%	0.75%	92%	79%
	PeopleOnStreet	-0.76%	209%	108%	0.58%	91%	82%
B	Kimono	-0.16%	225%	104%	0.15%	93%	94%
	ParkScene	-0.93%	219%	115%	1.07%	93%	74%
	Cactus	-0.80%	218%	109%	0.65%	95%	84%
	BQTerrace	-0.48%	208%	105%	0.41%	86%	85%
	BasketballDrive	-0.33%	216%	103%	0.25%	85%	91%
C	RaceHorses	-0.72%	213%	107%	0.38%	91%	80%
	BQMall	-0.98%	211%	104%	0.87%	91%	80%
	PartyScene	-1.16%	204%	108%	0.79%	92%	74%
	BaskebtallDrill	-0.44%	212%	101%	0.49%	89%	86%
D	RaceHorses	-1.01%	208%	108%	0.79%	90%	80%
	BQSquare	-1.12%	206%	105%	0.82%	93%	77%
	BlowingBubbles	-1.14%	200%	105%	0.74%	92%	73%
	BasketballPass	-0.75%	214%	103%	0.66%	94%	80%
E	FourPeople	-0.90%	221%	106%	1.32%	96%	84%
	Johnny	-0.64%	229%	101%	1.81%	98%	90%
	KristenAndSara	-0.89%	210%	105%	1.32%	95%	99%
	Average	-0.78%	245%	114%	0.71%	92%	82%

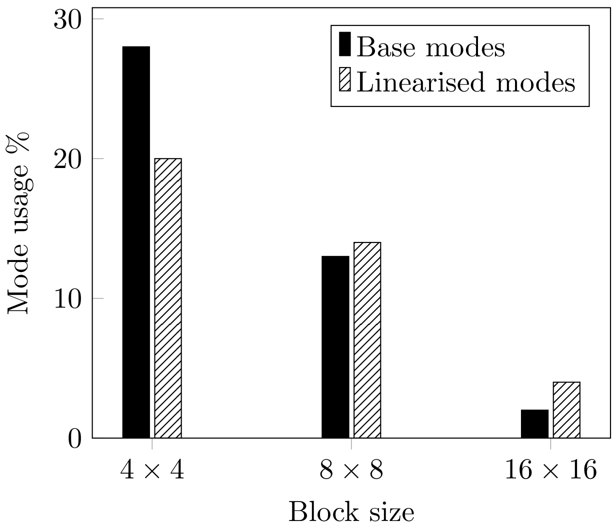


Figure 4.9: Comparison of mode usage between base FCN modes [111] and linearised FCN modes (Section 4.3). The mode usage indicates the percentage of intra-predicted blocks that use an FCN mode.

Taking this into account, smaller blocks are affected the most, as illustrated in Figure 4.9, given they are typically used to predict small details. In contrast, the linearised modes are used more often than the base modes in larger blocks. Still, the low prediction accuracy of the linearised and the signalling overhead result in lower coding gains than the base modes.

As mentioned in the previous subsection, the linearised modes are less complex than the base modes. Therefore, there is a reduction in the execution time: the encoder is 8% faster and the decoder is 18% faster. In addition, according to the mode usage, the linearised modes reduce the number of multiplications in up to two orders of magnitude, at the decoder side.

Even though the linearised modes do not represent the base modes completely, they show that simple models are useful when deployed as intra modes. A key benefit is that such simple modes provide insight on how each reference sample contributes in producing a prediction block. This knowledge was used to develop a different linear FCN for intra-prediction (Section 4.4), and the performance is reported in the next subsection.

4.8.4 Linear modes

This subsection presents the performance of the proposed linear modes (Section 4.4), which are trained from scratch using four layers. Notwithstanding, the codec implementation is done with a single layer. The results are reported in Table 4.5 in terms of BD-rate, encoding time (EncT) and decoding time (DecT), according to two different anchors: VTM 1.0 and VTM 1.0 enhanced with the base modes. Figure 4.10 displays the percentage of times the linear modes and base modes are selected at the encoder for each supported block size.

As shown in Table 4.5, the linear modes achieve average coding gains of 1.48% BD-rate compared to VTM 1.0. Furthermore, the results indicate that the linear modes can provide as much compression efficiency as the base modes, since negligible losses are reported for most video sequences.

Table 4.5: Coding efficiency of linear intra modes.

Class	**Sequence**	**Anchor: VTM 1.0**			**Anchor: base FCN [111]**		
		BD-rate Y	**EncT**	**DecT**	**BD-rate Y**	**EncT**	**DecT**
A	Traffic	-1.48%	231%	116%	0.05%	93%	83%
	PeopleOnStreet	-1.53%	231%	114%	-0.20%	91%	87%
B	Kimono	-0.26%	237%	107%	-0.04%	91%	95%
	ParkScene	-1.85%	221%	125%	0.14%	93%	81%
	Cactus	-1.44%	212%	112%	0.00%	92%	85%
	BQTerrace	-0.92%	229%	108%	-0.04%	95%	88%
	BasketballDrive	-0.60%	230%	106%	-0.01%	91%	94%
C	RaceHorses	-1.11%	222%	114%	-0.01%	95%	85%
	BQMall	-1.88%	225%	111%	-0.04%	97%	85%
	PartyScene	-1.84%	208%	112%	0.10%	91%	77%
	BaskebtallDrill	-1.02%	221%	106%	-0.09%	91%	90%
D	RaceHorses	-1.76%	218%	103%	0.03%	93%	84%
	BQSquare	-1.85%	215%	110%	0.08%	92%	81%
	BlowingBubbles	-1.79%	207%	112%	0.09%	90%	78%
	BasketballPass	-1.45%	230%	105%	-0.05%	95%	82%
E	FourPeople	-2.11%	233%	111%	0.09%	92%	88%
	Johnny	-1.46%	228%	106%	-0.02%	92%	94%
	KristenAndSara	-2.28%	223%	107%	-0.10%	93%	92%
	Average	-1.48%	223%	111%	0.00%	93%	86%

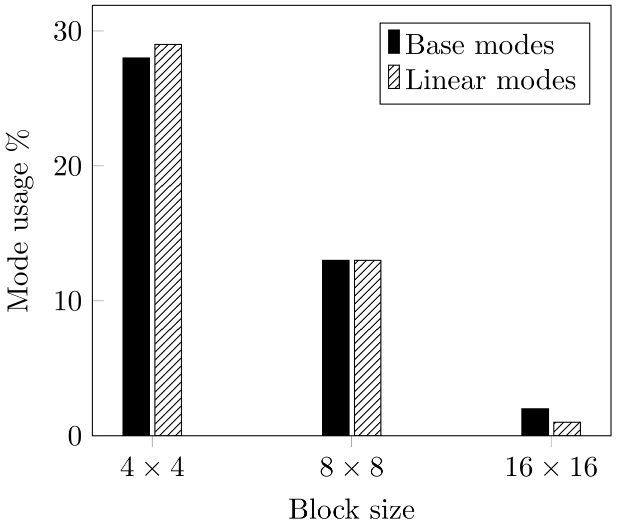
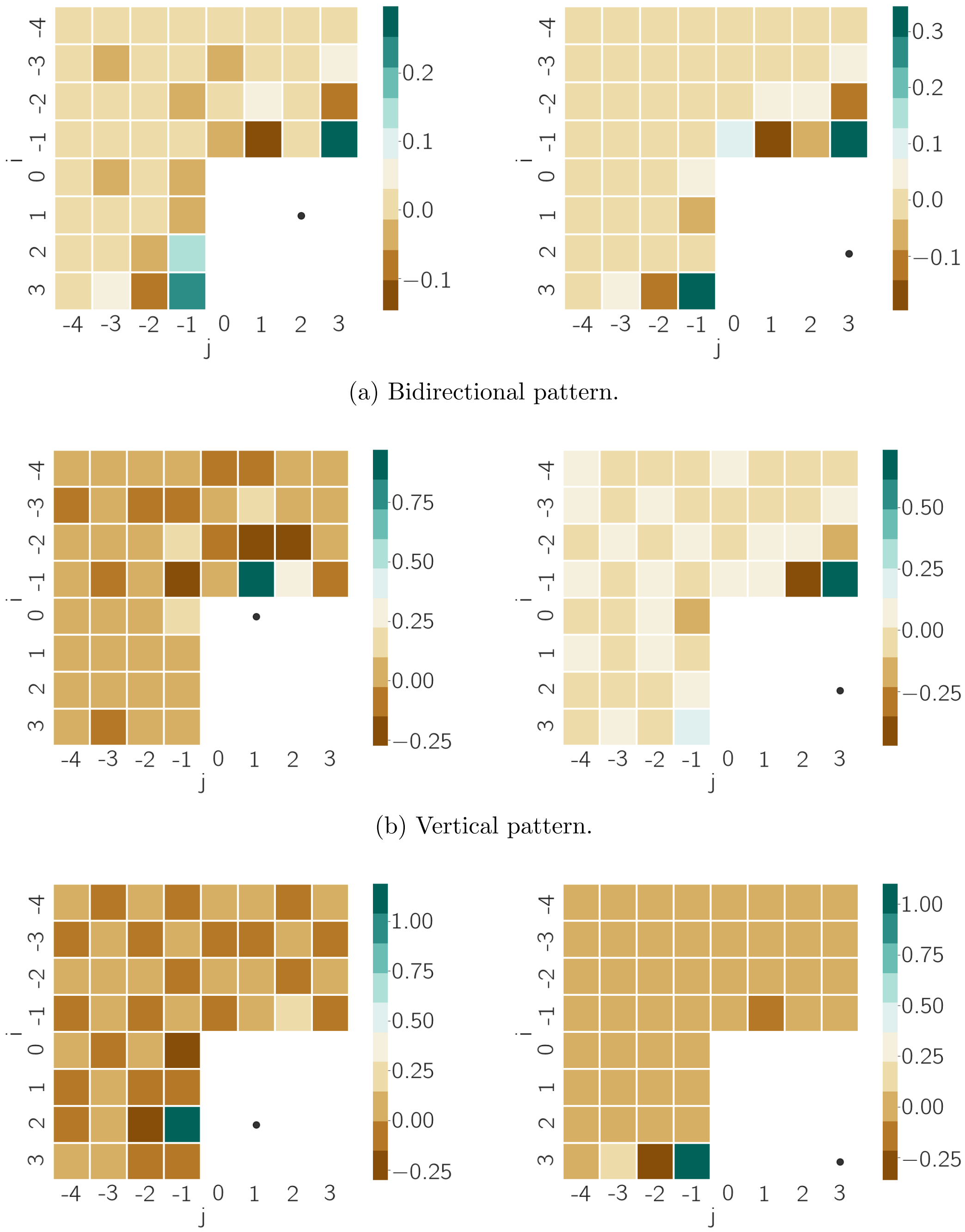


Figure 4.10: Comparison of mode usage between base FCN modes [111] and linear FCN modes (Section 4.4). The mode usage indicates the percentage of intra-predicted blocks that use an FCN mode.

These two approaches are also selected by the encoder in similar proportions, as shown Figure 4.10. At the same time, the linear modes reduce the runtime for the encoder by about 7% and the decoder by about 14%, meaning the linear modes achieve comparable coding gains to the base modes at a much lower computational cost.

The coding gains of VVC compared against HEVC are a product of the gains achieved jointly by the new coding tools. For instance, under the AI configuration, MRLP contributes in 0.3% and MIP contributes in 0.7% [35]. From that point of view, both the linearised and linear modes provide significant coding gains on top of VTM 1.0.

As previously stated, the key advantage of the proposed intra modes is that they allow for a clear explanation on how reference samples are processed in order to produce a prediction block. As revealed by Figure 4.11 there are modes that perform predictions following directional patterns, similarly to angular modes, while also introducing new gradient-like patterns which are not exploited by traditional modes. Another characteristic of the learned modes is that the most significant coefficients are usually located in reference lines closer to the target block.

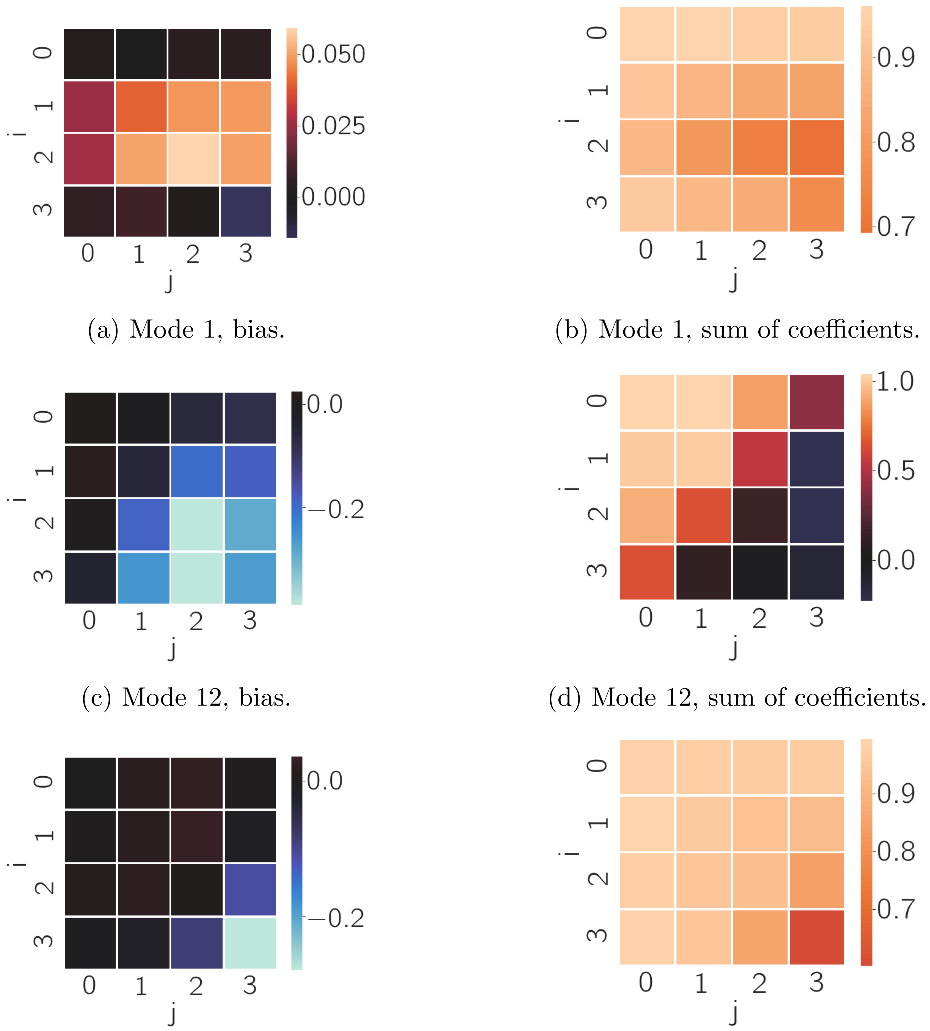


(a) Bidirectional pattern.

(b) Vertical pattern.

(c) Horizontal pattern.

Figure 4.11: Linear intra-prediction patterns for 4 × 4 blocks. The • marks the target sample.



(a) Mode 1, bias.

(b) Mode 1, sum of coefficients.

(c) Mode 12, bias.

(d) Mode 12, sum of coefficients.

(e) Mode 34, bias.

(f) Mode 34, sum of coefficients.

Figure 4.12: Bias of linear intra modes for 4 × 4

Last but not least, the linear modes not only combine the reference samples, but also they add a bias in order to obtain the final prediction. Each mode has a different bias per prediction sample. This is because the bias are related to the coefficients used to combine the reference samples. The learned bias are directly linked to the "energy" of initial predictions (combination of reference samples). When the sum of the coefficients used to combine the reference samples is close to one, then the bias is close to zero. In other words, the magnitude of the bias is inversely proportionate to the sum of the coefficients, as depicted in Figure 4.12.

4.8.5 Exploring the design of the linear FCN

So far it has been shown that, for intra-prediction, a linear FCN can provide as much coding efficiency as an FCN with non-linear activations, while reducing the computational cost. Looking at how to improve the coding efficiency and simplify the linear FCN further, this subsection evaluates two design considerations. Namely, the impact of (i) number of layers and (ii) pruning the coefficients.

Impact of number of layers in the linear FCN

As referred to earlier, the linear FCN is trained from scratch using the same number of layers as the base FCN (four). After training, the linear intra modes are defined by expressing the multi-layer model as a single-layer model. Taking this into account, the linear FCN could be defined with a lower number of layers too. Such scenarios are next explored by training linear FCNs with three, two and one layers, following the methodology described in Section 4.6.

The new linear modes are also implemented on top of VTM 1.0. Table 4.6 shows a comparison between the original linear modes (trained with four layers) and the new linear modes (trained with three, two and one layers) in terms of BD-rate. All approaches produce on average more than 1% BD-rate coding gains. Moreover, the results indicate that, even for linear FCNs, having more layers can increase the prediction accuracy.

Table 4.6: Coding efficiency of linear modes trained with different number of layers.

Class	**BD-rate Y**			
	4 layers	**3 layers**	**2 layers**	**1 layer**
A	-1.51%	-1.41%	-1.44%	-0.77%
B	-1.01%	-0.99%	-1.01%	-0.58%
C	-1.46%	-1.42%	-1.42%	-1.08%
D	-1.71%	-1.69%	-1.68%	-1.29%
E	-1.95%	-1.89%	-1.80%	-1.22%
Average	-1.48%	-1.44%	-1.43%	-1.22%

Pruning the linear FCN

Among the different linear FCNs evaluated, the one with four layers produce the highest coding efficiency. A further simplification step consists in removing from the FCN the coefficients that contribute the least in the prediction. Consequently, the pruning methodology described in Section 4.5 is applied to the linear FCN. The threshold used is 0.1, meaning coefficients with lower magnitude than 0.1 are pruned.

The resulting FCN is first evaluated in terms of sparsity and the coding efficiency of the new intra modes is measured with the BD-rate metric. Table 4.7 shows the percentage range of coefficients that are zero-out for the FCN of each block size supported. Additionally, Table 4.8 includes the BD-rate with respect to the reference VTM 1.0 and the original linear modes. The results in Table 4.7 indicate that at least 49% of the coefficients (per variable) are removed. The bias are consistently removed and, mostly, modes for larger block sizes add non-zero bias for a few samples in a prediction block. According to Table 4.8, the new linear modes achieve average coding gains of 1.46% BD-rate compared to VTM 1.0. Overall, after pruning coefficients with low magnitude, the original coding efficiency is preserved.

Table 4.7: Coefficients pruned in linear FCN.

Block size	**Coefficients pruned**
4×4	49% to 100%
8×8	57% to 100%
16×16	59% to 100%

Table 4.8: Coding efficiency of pruned linear modes.

Class	**BD-rate Y**	
	Anchor: VTM 1.0	**Anchor: Linear modes**
A	-1.38%	0.13%
B	-1.06%	-0.05%
C	-1.45%	0.02%
D	-1.68%	0.03%
E	-1.88%	0.08%
Average	-1.46%	0.02%

4.9 Conclusion

This chapter presents two different approaches to simplify FCNs for intra-prediction in order to devise simpler modes. The first simplification consists on linearising an FCN. The network is first trained. Afterwards, the learned coefficients are used to derive linear intra-prediction modes. The results show that the linearised modes require less computational resources than the base FCN. Moreover, they capture a few of the relationships between reference samples and prediction samples, as they achieve on average 30% of the coding gains generated by the base FCN.

The second simplification is a linear FCN trained from scratch and expressed as a single-layer model for analysis and implementation purposes. This approach reported coding gains of 1.48% BD-rate, comparable to the base FCN, and obtained at a lower computational cost. Notably, this approach also provides explainable modes that describe clearly how reference samples are to be combined in order to generate a prediction block.

More results showed that even a linear FCNs can benefit from having a multi-layer structure. In addition, removing less important connections between reference samples and prediction samples can reduce the complexity of the linear FCN further, while keeping the coding efficiency.

Equally important, intra-prediction modes where the sum of the coefficients associated to the reference samples is close to one have bias close to zero. Taking this into account, further experiments could include adding such constraints, which could help to reduce the number of parameters even during training. From the inference perspective, it is perhaps possible to explore model quantisation to allow for integer operations only, which could mean faster deployment.

Chapter 5

Super pixel intra-prediction

Chapter 4 showed that linear models FCNs can achieve similar coding performance as FCNs with non-linearities. However, as the block size increases so it does the number of reference samples to be used by the FCN. Taking this into account this chapter explores the usage of resolution adaptation for intra-prediction techniques, which could potentially help on reducing the number of parameters required by an FCN.

The rest of the Chapter is organised as follows. Section 5.1 introduces the background work. Section 5.2 presents the proposed approach. Section 5.3 reports the experimental results and Section 5.4 provides some final remarks.

5.1 Background work

Video resolution continues to increase thanks to advancements in video capture and display devices. Resolutions such as UHD offer sharper and more vivid visual content, but also require larger bit rates. In this context, resolution adaptation techniques, which have been studied towards bit rate reduction by compressing video at a lower scale than the original [29, 100, 107], are still relevant. Recent advances in computational capabilities, super resolution and NNs have also impacted resolution adaptation in video coding. Accordingly, the techniques mentioned in this chapter use NNs somehow.

5.1.1 Resolution adaptation in video coding

Perhaps the most common type of resolution adaptation used in video coding is the one in which the down-sampling and up-sampling operations are independent of the video codec. Hence, the down-sampling can be seen as a pre-processing step and the up-sampling is considered a post-processing stage, meaning the compression is carried out by a standard video codec [93] (Figure 5.1). Notwithstanding, the rate controller can be modified with two goals in mind. First, produce a higher rate-quality performance by encoding the LR video than by encoding the HR video [2]. Second, map the quality of the HR encoded video to the quality of the corresponding LR video encoded and up-sampled afterwards [137].

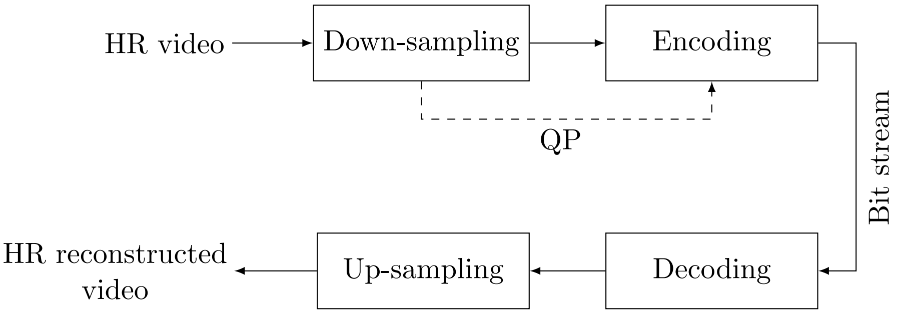


Figure 5.1: Resolution adaptation independent of the video codec. The dashed line represents the modification to the rate controller, which is optional.

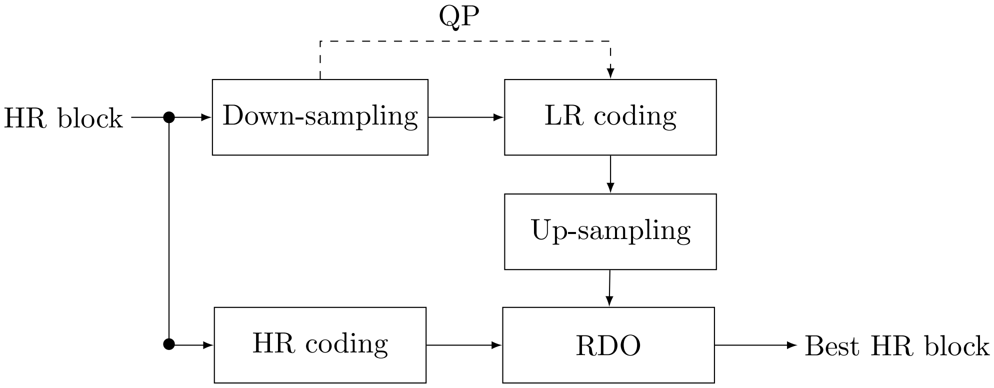


Figure 5.2: Resolution adaptation as part of the video codec. The dashed line represents the modification to the rate controller, which is optional.

Resolution adaptation has been embedded in the video codec too (Figure 5.2). For instance, a CTU can be encoded at HR or LR and the choice is made by the encoder using RDO [81]. To avoid disrupting the encoding of neighbouring CTUs, the down-sampled CTU (reconstruction) is up-sampled. Moreover, similar to the aforementioned solutions, the rate controller is modified.

5.1.2 Resolution adaptation in intra-prediction

A whole different technique, known as MIP [120], uses resolution adaptation within the video codec in order to exploit spatial redundancies. MIP consists of intra modes that consider both down-sampling and up-sampling as part of the prediction process. Fixed filters are used for both down-sampling the reference samples and up-sampling the down-sampled prediction, and the intra modes learn the mapping in between. Furthermore, as the intra modes are trained to provide the best prediction considering the down-sampled and up-sampling operations, it can avoid modifications to other modules of the codec, like quantisation.

MIP uses Haar wavelets for the down-sampling and conventional interpolation filters for the up-sampling. Namely, for each empty sample, its value is computed as a weighted average between the two closest known samples. The weights are given by the distances to these samples. As such, MIP learns to compute a down-sampled prediction. On the contrary, the technique proposed in this chapter learns all the operations in the workflow: down-sampling, prediction and up-sampling.

5.2 Intra-prediction with super pixels

Inspired by [120], this chapter presents an intra-prediction technique that learns to compute an intra-prediction from down-sampled samples. A set of super pixels are formed by down-sampling the reference samples. These are used next to compute a down-sampled prediction block that is afterwards up-sampled in order to obtain the prediction block in full size.

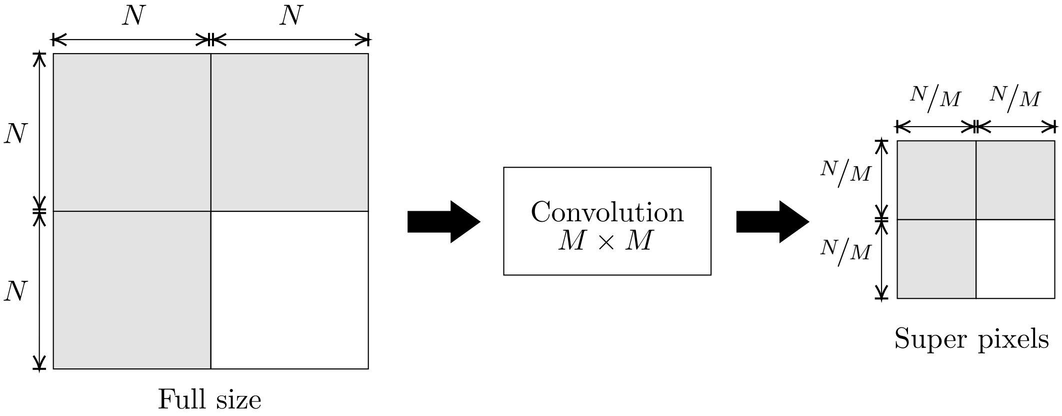


Figure 5.3: Down-sampling in super pixel intra-prediction. M is the down-sampling factor. The grey blocks are reference blocks and the white block is the target block. The down-sampling is only applied to the reference blocks, the target block is illustrated for reference.

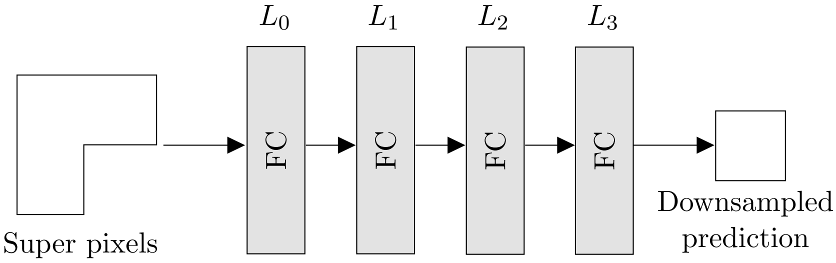


Figure 5.4: Linear FCN in super pixel intra-prediction. This FCN is proposed in Chapter 4.

In contrast to [120], in the super pixel intra-prediction all the steps, including the down-sampling and up-sampling operations, are learnable. The first step consists in down-sampling the reference samples. This operation is modelled with a convolution, where the size of the kernel is $M \times M$ and M is the down-sampling factor (Figure 5.3). Once the reference samples have been down-sampled, a subset of reference lines are selected and used to compute the prediction using the second module: a linear FCN as the one proposed in Chapter 4. Namely, the FCN has four FC layers, the input is the down-sampled reference samples and the output is the down-sampled prediction (Figure 5.4). The resulting down-sampled prediction is then up-sampled, via a learned interpolation, to full size and this one becomes the final prediction block. The interpolation is made in such a way that one line of reference samples is used alongside the down-sampled prediction in order to compute the final prediction.

Figure 5.5 illustrates the up-sampling process, which is split in two stages: preparation and interpolation. The preparation stage creates a prediction block in the original scale, ready to be up-sampled. For that purpose the down-sampled prediction of size ${}^{N}\!/_{M} \times {}^{N}\!/_{M}$ is transformed into a sparse block of size $N \times N$. Next, one line of the closest reference samples are added to the block to the top, left and corner. The interpolation stage is used to compute the missing samples in the prediction block. The interpolation is modelled with a set of $M^2 - 1$ convolutions with kernels of size $P \times P$, where $P = M + 1$. In this way, each convolution predicts a missing sample inside by the area enclosed by the kernel at each step. Since the prediction block is updated at each step, the interpolation considers not only the down-sampled prediction and reference samples, but also previous samples resulting from interpolation.

The implementation of this approach in a video codec, such as VTM, requires changing the bitstream syntax to support the signalling and usage of the new intra-prediction mode.

5.2.1 Training methodology

As in Chapter 4, the training is done using DIV2K dataset. Only that in this case the target blocks are of shape 64 × 64. Both inputs and ground truths are normalised by

$$\mathcal{N}(\mathbf{y})_i = \frac{y_i}{2^B - 1}, \tag{5.1}$$

where B is the bit depth. Four lines of super pixels are considered and 10 modes are used. The optimisation is also done as described in Subsection 4.6.1.

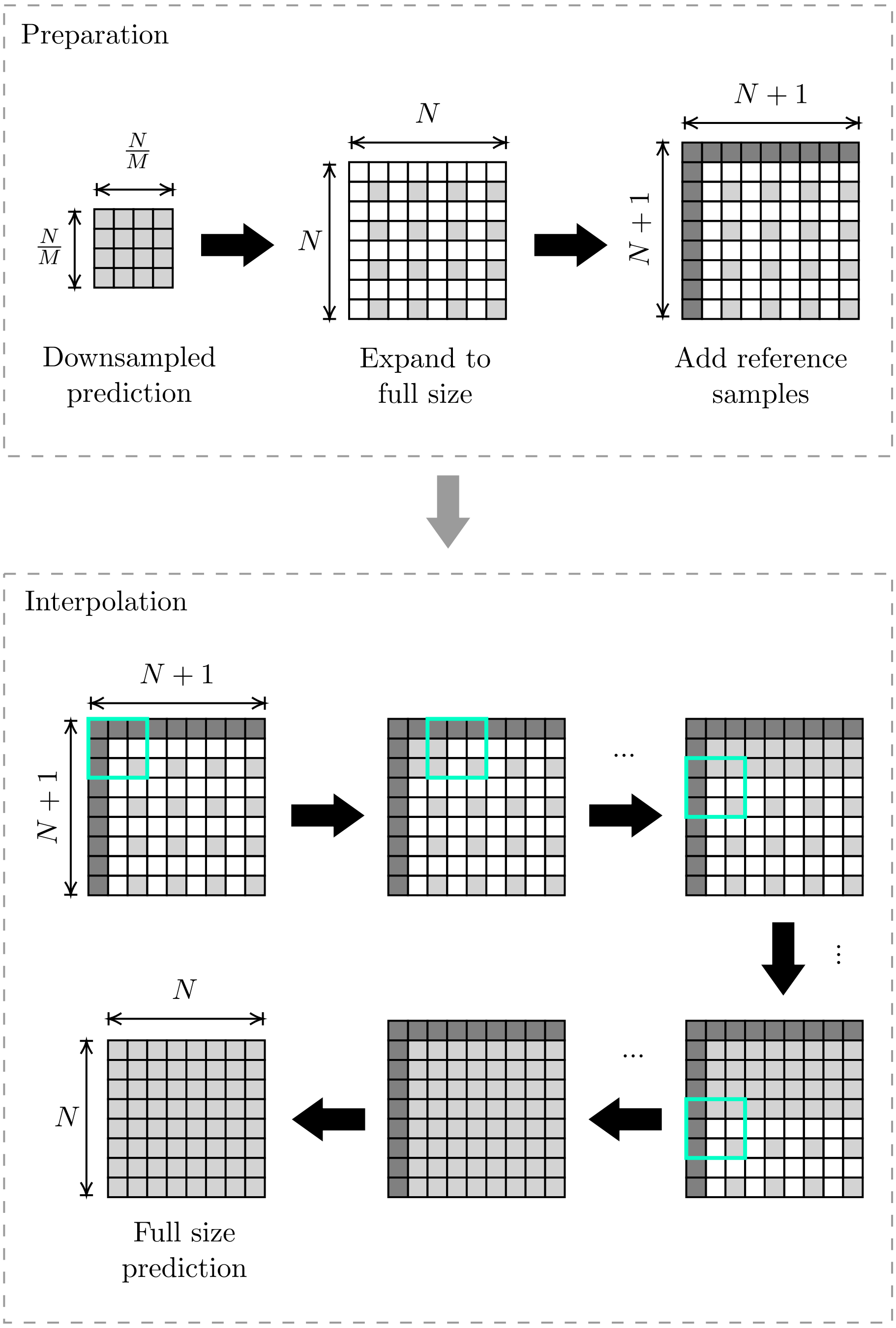


Figure 5.5: Up-sampling in super pixel intra-prediction. The light grey samples belong to the prediction block and the white samples are missing values. The dark grey samples are reference samples from the original scale. The aquamarine square represents the convolution kernel $P \times P$ used to compute the missing samples from existing ones.

5.3 Experiments

This section presents the experimental evaluation of the super pixel approach. The down-sampling factor used is $M = 4$. The evaluation is done in TensorFlow [1] 1.14 on a NVIDIA Titan X Pascal GPU[1].

The results are expressed in terms of average PSNR obtained by predicting 89,000 blocks of shape 64×64 from DIV2K validation dataset. Furthermore, the performance of the proposed approach is compared against that of MIP (11 modes) from VTM 6.0 [18] (from here on referred to as the reference).

The prediction of a whole frame using the reference and the super pixel intra-prediction, showed that for both techniques small details are difficult to predict. Considering that the prediction is done block-wise, both methods produce blocking artefacts. However, the artefacts are more evident when using the super pixel intra-prediction, even in neighbouring areas where smooth content is the target (Figure 5.6).

It was also observed tha the artefacts are also present in the down-sampled prediction, which might indicate the source is either the down-sampling stage or the intra-prediction stage. Currently, the down-sampling operation and the up-sampling operation are common among the different modes being trained. Accordingly, these stages could be redesigned and perhaps be fine-tuned separately in order to generate more accurate predictions.

The comparison of the prediction accuracy, in terms of average PSNR is shown in Table 5.1. The PSNR is computed per block and averaged afterwards. The results show the MIP technique achieves higher prediction accuracy than the super pixel approach (around 14 dB more), for which the strong blocking artefacts contribute to the loss in the accuracy. Taking this into account, the evaluation is not yet done in VTM since the super pixel approach needs to achieve a higher prediction accuracy first.

[1] We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU used for this research.

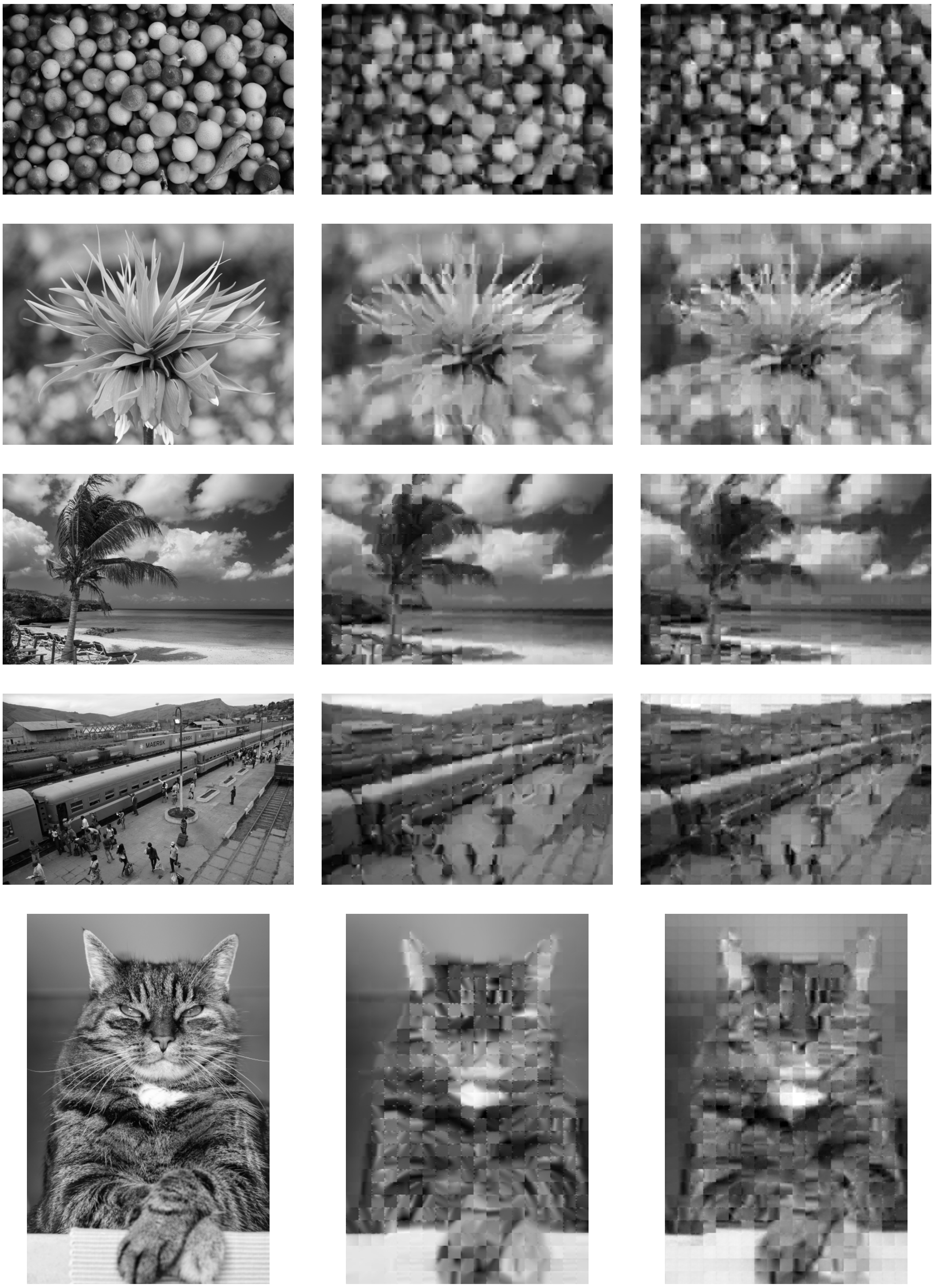
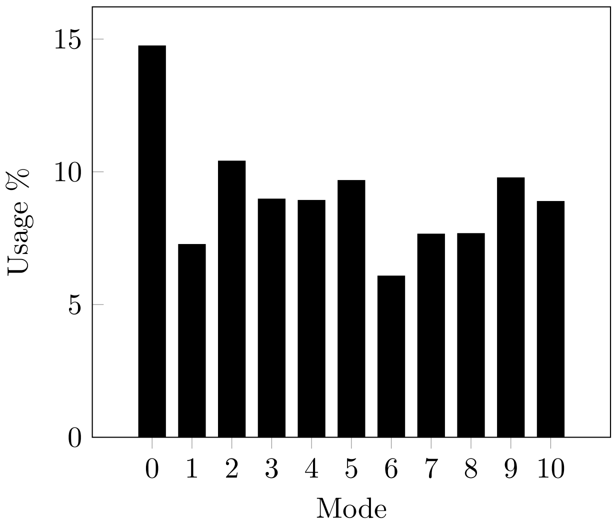


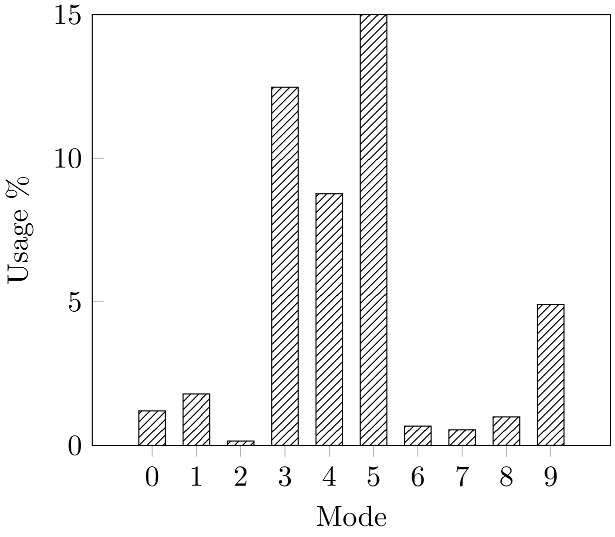
Figure 5.6: Prediction comparison of intra-prediction modes with resolution adaptation. Original frames are displayed on the left, MIP predictions are on the middle and super pixel prediction are on the right.

Table 5.1: Accuracy comparison of intra-prediction modes with resolution adaptation.

Intra-prediction	**PSNR**
MIP	34.65
Super pixel	20.07



(a) MIP modes



(b) Super pixel modes. Mode 5 usage is 68%.

Figure 5.7: Usage comparison of intra-prediction modes with resolution adaptation.

The comparison of mode usage between the reference and the proposed approach is displayed in Figure 5.7. It is found that there are super pixel modes used way more often than others, while the reference modes usage is more evenly distributed.

Another reason for the low performance of the proposed approach could be associated to having a dominant mode, as shown in Figure 5.7(b). During training, a single mode was back-propagated most of the time. Since the model itself is simple, the mode cannot generalise well for a large variation of content. This shortcoming could be tackled by introducing a more balanced training methodology.

5.4 Conclusion

This chapter presents the super pixel intra-prediction, which includes resolution adaptation technique. Since both the down-sampling and up-sampling are part of the proposed method, no further changes to other modules of the encoding are required (i.e. quantisation). Moreover, all the stages of the approach are learned. Namely, the down-sampling, the actual intra-prediction and the up-sampling.

The super pixel modes were compared against MIP modes available in VTM 6.0. The results showed the MIP technique achieves higher prediction accuracy than the proposed technique, which in turn produces predictions with blocking artefacts and noisy content. These limitations could be associated to the down-sampling operation, the actual intra-prediction technique and the narrow generalisation of a single learned mode for a wide variety of content. Nevertheless, the super pixel intra-prediction serves as a proof of concept aiming to simplify FCN-based intra-prediction for large block sizes. This is done by means of an end-to-end FCN-based intra-prediction method with resolution adaptation.

Chapter 6

Conclusion and future work

Delivery of high-quality video continues to be a challenge. Consumers expect more immersive video enhanced by video formats, such as HFR, HDR and UHD. This trend and the increasing interest in VoD and video streaming services calls for more efficient video compression, which might be achieved by the usage of unconventional techniques, such as NNs.

In that regard, the aim of this thesis was to explore the usage of NNs as intra coding tools for hybrid video coding schemes. The research work carried out focused on the estimation of RD metrics and intra-prediction techniques. The proposed approaches are self-supervised since the training data is either the original data (video frames) or data generated by a deterministic process: the video encoder.

6.1 Conclusion

The main contributions presented in this thesis, organised by chapter, are presented below.

Chapter 3 introduced a CNN framework utilised to estimate the RD metrics that would be obtained when applying intra coding. The framework consists of two CNNs.

One is used to predict distortion maps, element-wise difference between the original frame and its reconstruction. The second CNN is used to prediction rate and distortion metrics for a set of QPs. The CNNs are self-supervised as the input data consists not only on the original data, but also the metrics obtained as a result of the video encoding process. Furthermore, the CNNs are trained to predict metrics that would otherwise be known only after the encoding process is carried out. In this way, the CNNs allow to know the RD metrics without multi-pass encoding. The evaluation showed that there is a strong correlation between ground truths and prediction. However, some limitations were observed as well. For distortion maps, the CNN cannot predict accurately granular distortion (resulting from encoding with low QPs). For rate values, since these are not normalised, the generalisation and convergence of the CNN was compromised.

Chapter 4 presented two simplifications for FCN-based intra-prediction. The FCNs were trained in a self-supervised manner as the original data was used both as input and ground truth. Also, multiple modes were trained at the same time. The first simplification consists on discarding the non-linear activations and normalise the trained predictors. This approach produced BD-rate savings and reduced the computational complexity of the FCN. However, most of the prediction capabilities of the original FCN were lost. For that reason, a second approach was designed, where still the FCN was simplified by removing the non-linear activations, but the model was trained from scratch. In this scenario, the coding gains were much higher than with the previous one and the model was less complex. Moreover, the obtained gains were close to the gains of the original FCN. This final linear model allowed for further analysis and simplification.

When expressing the model using a single-layer, it was observed that most earned intra-prediction modes combined somehow horizontal, vertical and directional patterns. In addition, it was observed that in a linear FCN for intra-prediction, the magnitude of the bias or offset is inversely proportionate to the sum of the coefficients used to combine the reference samples. Namely, when the sum of the coefficients is close to one, the magnitude of the bias tends to zero. Additionally, linear model was pruned successfully without affecting the prediction capabilities too much, generating sparse predictors.

Finally, while the training methodology allowed for mode specialisation, it was difficult to make sure all modes were actually back-propagated in similar proportions. For this reason, some modes ended up being scarcely used.

Chapter 5 focused on the super pixel intra-prediction technique, which uses resolution adaptation. This technique downsamples the reference samples (named super pixels), computes the prediction in the down-sampled domain and finally applies up-sampling to produce the full size prediction. Furthermore, all these stages are learned with self-supervision as original content is used both as input and ground truth. The down-sampling and up-sampling operations are modelled using convolutional layer, shared by the different intra-prediction NN. In addition, the intra-prediction is done using the linear models described in Chapter 4. The evaluation showed the super pixel intra-prediction can predict coherent blocks, but the prediction of smooth regions results in blocking artefacts, whereas predictions of highly textured regions are noisy. Moreover, these artefacts were not only displayed in full size predictions, but down-sampled predictions as well.

6.2 Future work

Although the contributions presented in this thesis are promising, they can be improved and are subject to further developments. In the context of Chapter 3, the CNNs can be redesigned to produce more accurate results. For instance, the CNN that predicts distortion maps should be able to detect finer details in order to capture the granular distortion generated when using low QPs. Afterwards, the resulting CNN could be used as a post-processing step to enhance intra-coded blocks, before adding the frame to the picture buffer. Alternatively, the prediction of rate metrics should focus first on how to deal with label values that do not fall in a specific range. By having more accurate RD predictions, the RD model estimation would be more accurate too. This could be employed later to develop bit allocation techniques for intra-coded frames. Eventually, a similar approach could be investigated for inter-coded frames.

With respect to Chapter 4, the proposed simplification techniques can be applied to other NN architectures, and since an NN can be pruned, it might also be extended by adding extra connections during training. As all the operations are implemented VTM using floating-point arithmetic, the computational complexity could be brought down by transforming the modes so all operations use integer precision only. Though the methodology used to train the models allows for intra mode specialisation, it was difficult to ensure the update frequency was balanced among the different modes. New training methodologies could be investigated to achieve this goal.

Finally, concerning Chapter 5, the modules of the super pixel intra-prediction could be improved using four different alternatives. Firstly, each stage of the training process could be fine tuned (down-sampling, intra-prediction and up-sampling). Secondly, architecture of both the down-sampling and up-sampling modules could be redesigned. Thirdly, as these modules are common among the different modes being trained, there could be instead as many down-sampling and up-sampling variations as modes are. Fourthly, an attention module could be included in order to determine which reference lines are more suitable to be used to compute the down-sampled prediction.

Bibliography

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, S. teiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, P. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] M. Afonso, F. Zhang, and D. R. Bull. Video compression based on spatio-temporal resolution adaptation. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(1):275–280, 2019.

[3] J. I. Agbinya. Interpolation using the discrete cosine transform. *Electronics Letters*, 28(20):1927–1928, 1992.

[4] E. Agustsson and R. Timofte. NTIRE 2017 challenge on single image super-resolution: Dataset and study. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1122–1131, July 2017.

[5] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, C-23(1):90–93, 1974.

[6] H. Amer, A. Rashwan, and E. Yang. Fully connected network for HEVC CU split decision equipped with laplacian transparent composite model. In *2018 Picture Coding Symposium (PCS)*, pages 189–193, 2018.

[7] Y. Bengio. Practical recommendations for gradient-based training of deep architectures. In G. Montavon, G. B. Orr, and K.-R. Müller, editors, *Neural Networks: Tricks of the Trade: Second Edition*, pages 437–478. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[8] G. Bjøntegaard. Calculation of average PSNR differences between RD-curves. Technical report, VCEG 13th Meeting, document VCEG-M33, Austin, US, April 2001.

[9] G. Bjøntegaard. Improvements of the BD-PSNR model. Technical report, VCEG 35th Meeting, document VCEG-AI11, Berlin, DE, July 2008.

[10] F. Bossen. Common test conditions and software reference configurations. Technical report, document JCTVC-L1100 WG11 Number: m28412, 12th Meeting, Geneva, CH, April 2013.

[11] Y. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2559–2566, 2010.

[12] Y. Boureau, J. Ponce, and Y. LeCun. A theoretical analysis of feature pooling in visual recognition. In *Proceedings of the 27th International Conference on Machine Learning*, ICML'10, page 111–118, 2010.

[13] V. Britanak, P. C. Yip, and K. R. Rao. *Discrete Cosine and Sine Transforms: General Properties, Fast Algorithms and Integer Approximations.* Elsevier Science, 2010.

[14] B. Bross, J. Chen, S. Liu, and Y.-K. Wang. Versatile Video Coding (Draft 9). Technical report, JVET 18th Meeting, document JVET-R2001, April 2020.

[15] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8):832, 2019.

[16] Y.-J. Chang, H.-J. Jhu, H.-Y. Jian, L. Zhao, X. Zhao, X. Li, S. Liu, B. Bross, P. Keydel, H. Schwarz, D. Marpe, and T. Wiegand. Intra prediction using multiple reference lines for the versatile video coding standard. In A. G. Tescher and T. Ebrahimi, editors, *Applications of Digital Image Processing XLII*, volume 11137, pages 302 – 309. International Society for Optics and Photonics, SPIE, 2019.

[17] J. Chen and E. Alshina. Algorithm description for Versatile Video Coding and Test Model 1 (VTM 1). document jvet-j1002-v2, 10th meeting, JVET, San Diego, US, April 2018.

[18] J. Chen, Y. Ye, and S. H. Kim. Algorithm description for Versatile Video Coding and Test Model 6 (VTM 6). Technical report, document JVET-O2002-v2, 15th Meeting, Gothenburg, SE, July 2019.

[19] Y. Chen, D. Murherjee, J. Han, A. Grange, Y. Xu, Z. Liu, S. Parker, C. Chen, H. Su, U. Joshi, C. Chiang, Y. Wang, P. Wilkins, J. Bankoski, L. Trudeau, N. Egge, J. Valin, T. Davies, S. Midtskogen, A. Norkin, and P. de Rivaz. An overview of core coding tools in the AV1 video codec. In *2018 Picture Coding Symposium (PCS)*, pages 41–45, 2018.

[20] Z. Chen, J. Shi, and W. Li. Learned fast HEVC intra coding. *IEEE Transactions on Image Processing*, 29:5431–5446, 2020.

[21] N. Choi, Y. Piao, K. Choi, and C. Kim. CE3.3 related: Intra 67 modes coding with 3MPM. Technical report, JVET 11th Meeting, document JVET-K0529-v4, Ljubljana, SI, July 2018.

[22] Cisco. Cisco visual networking index: Forecast and trends, 2017–2022. White paper, 2019.

[23] D.-A. Clevert, T. Unterthiner, and S. Hochreiter. Meta fast and accurate deep network learning by Exponential Linear Units (ELUs). In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[24] W. Cui, T. Zhang, S. Zhang, F. Jiang, W. Zuo, Z. Wan, and D. Zhao. Convolutional neural networks based intra prediction for HEVC. In *2017 Data Compression Conference (DCC)*, 2017.

[25] Y. Dai, D. Liu, Z. Zha, and F. Wu. A CNN-based in-loop filter with CU classification for HEVC. In *2018 IEEE Visual Communications and Image Processing (VCIP)*, 2018.

[26] S. De-Luxán-Hernández, V. George, J. Ma, T. Nguyen, H. Schwarz, D. Marpe, and T. Wiegand. An intra subpartition coding mode for VVC. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 1203–1207, 2019.

[27] Denelson83. SMPTE color bars. `https://commons.wikimedia.org/wiki/File:SMPTE_Color_Bars.svg`, 2006. Online; accessed April 4, 2020.

[28] C. Dong, C. C. Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 184–199. Springer International Publishing, 2014.

[29] J. Dong and Y. Ye. Adaptive downsampling for high-definition video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 24(3):480–488, 2014.

[30] T. Eiter and H. Mannila. Computing the discrete Fréchet distance. Technical report, CD-TR 94/64, Christian Doppler Labor für Expertensyteme, Technische Universität Wien, Vienna, AT, April 1994.

[31] H. Everett. Generalized lagrange multiplier method for solving problems of optimum allocation of resources. *Operations Research*, 11(3):399–417, 1963.

[32] Z. Feng, P. Liu, K. Jia, and K. Duan. HEVC fast intra coding based CTU depth range prediction. In *2018 IEEE 3rd International Conference on Image, Vision and Computing (ICIVC)*, pages 551–555, 2018.

[33] A. Filippov, V. Rufitskiy, J. Chen, G. Van der Auwera, A. K. Ramasubramonian, V. Seregin, T. Hsieh, and M. Karczewicz. CE3: A combination of tests 3.1.2 and 3.1.4 for intra reference sample interpolation filter. Technical report, JVET 12th Meeting, document JVET-L0628, Macao, MO, October 2018.

[34] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, and J. Pineau. An introduction to deep reinforcement learning. *Foundations and Trends in Machine Learning*, 11(3-4):219–354, 2018.

[35] E. François, M. Kerdranvat, R. Jullian, C. Chevance, P. De Lagrange, F. Urban, T. Poirier, and T. Chen. VVC per-tool performance evaluation compared to HEVC. In *International Broadcasting Convention (IBC)*, 2020.

[36] A. Gabriellini, M. Naccari, M. Mrak, and D. Flynn. Spatial transform skip in the emerging high efficiency video coding standard. In *2012 19th IEEE International Conference on Image Processing*, pages 185–188, 2012.

[37] B. Girod. Motion-compensating prediction with fractional-pel accuracy. *IEEE Transactions on Communications*, 41(4):604–612, 1993.

[38] B. Girod. Rate-constrained motion estimation. In A. K. Katsaggelos, editor, *Visual Communications and Image Processing '94*, volume 2308, pages 1026 – 1034. International Society for Optics and Photonics, SPIE, 1994.

[39] C. A. Gomez-Uribe and B. Hunt. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems*, 6(4), 2016.

[40] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* MIT Press, 2016.

[41] V. K. Goyal. Theoretical foundations of transform coding. *IEEE Signal Processing Magazine*, 18(5):9–21, 2001.

[42] L. Guo, V. Li, J. Beckley, V. Selvaraj, and J. Watts. Netflix now streaming AV1 on Android. `https://netflixtechblog.com/netflix-now-streaming-av1-on-android-d5264a515202`, 2020. Online; accessed April 1, 2020.

[43] W.-J. Han, J. Min, I.-K. Kim, E. Alshina, A. Alshin, T. Lee, J. Chen, V. Seregin, S. Lee, Y. M. Hong, M.-S. Cheon, N. Shlyakhov, K. McCann, T. Davies, and J.-H. Park. Improved video compression efficiency through flexible unit representation and corresponding extension of coding tools. *IEEE Transactions on Circuits and Systems for Video Technology*, 20(12):1709–1720, 2010.

[44] M. M. Hannuksela, Y.-K. Wang, and M. Gabbouj. Isolated regions in video coding. *IEEE Transactions on Multimedia*, 6(2):259–267, 2004.

[45] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 1026–1034, USA, 2015. IEEE Computer Society.

[46] P. Helle, S. Oudin, B. Bross, D. Marpe, M. O. Bici, K. Ugur, J. Jung, G. Clare, and T. Wiegand. Block merging for quadtree-based partitioning in HEVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1720–1731, 2012.

[47] P. Helle, J. Pfaff, M. Schäfer, R. Rischke, H. Schwarz, D. Marpe, and T. Wiegand. Intra picture prediction for video coding with neural networks. In *2019 Data Compression Conference (DCC)*, pages 448–457, 2019.

[48] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[49] A. Horé and D. Ziou. Image quality metrics: PSNR vs. SSIM. In *2010 20th International Conference on Pattern Recognition*, pages 2366–2369, 2010.

[50] J. Hu, W. Peng, and C. Chung. Reinforcement learning for HEVC/H.265 intra-frame rate control. In *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2018.

[51] Y. Hu, W. Yang, M. Li, and J. Liu. Progressive spatial recurrent neural network for intra prediction. *IEEE Transactions on Multimedia*, 21(12):3024–3037, 2019.

[52] D. H. Hubel. *Eye, Brain, and Vision.* Scientific American Library series. Scientific American Library, 1988.

[53] S. Huo, D. Liu, F. Wu, and H. Li. Convolutional neural network-based motion compensation refinement for video coding. In *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2018.

[54] ITS. Description of video quality metric (VQM) software. `https://www.its.bldrdoc.gov/research-topics/video-quality-research/guides-and-tutorials/description-of-vqm-tools.aspx`. Online; accessed June 27, 2021.

[55] ITU-T. ITU-T Recommendation H.261, video codec for audiovisual services at p × 64 kbits. Technical report, March 1993.

[56] ITU-T. ITU-T Recommendation H.266, versatile video coding. Technical report, August 2020.

[57] JCT-VC. HEVC test model (HM) version 16.9. `https://vcgit.hhi.fraunhofer.de/jvet/HM/-/tree/HM-16.9`, 2016. Online; accessed June 21, 2021.

[58] C. Jia, S. Wang, X. Zhang, S. Wang, J. Liu, S. Pu, and S. Ma. Content-aware convolutional neural network for in-loop filtering in high efficiency video coding. *IEEE Transactions on Image Processing*, 28(7):3343–3356, 2019.

[59] M. Karczewicz, P. Chen, R. L. Joshi, X. Wang, W. Chien, R. Panchal, Y. Reznik, M. Coban, and I. S. Chong. A hybrid video coder based on extended macroblock sizes, improved interpolation, and flexible motion representation. *IEEE Transactions on Circuits and Systems for Video Technology*, 20(12):1698–1708, 2010.

[60] A. K. Khan and H. Jamal. The intra prediction in H.264. In T. Sobh, K. Elleithy, A. Mahmood, and M. A. Karim, editors, *Novel Algorithms and Techniques In Telecommunications, Automation and Industrial Electronics*, pages 11–15, Dordrecht, 2008. Springer Netherlands.

[61] J. Kim, J. K. Lee, and K. M. Lee. Accurate image super-resolution using very deep convolutional networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1646–1654, 2016.

[62] T. King, S. Butcher, and L. Zalewski. *Apocrita - High Performance Computing Cluster for Queen Mary University of London*, March 2017.

[63] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015.

[64] O. Lahav, N. Mastronarde, and M. van der Schaar. What is interpretable? using machine learning to design interpretable decision-support systems. *CoRR*, abs/1811.10799, 2018.

[65] 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*, 22(12):1792–1801, 2012.

[66] T. Laude, F. Haub, and J. Ostermann. HEVC inter coding using deep recurrent neural networks and artificial reference pictures. In *2019 Picture Coding Symposium (PCS)*, 2019.

[67] T. Laude and J. Ostermann. Deep learning-based intra prediction mode decision for HEVC. In *2016 Picture Coding Symposium (PCS)*, 2016.

[68] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[69] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In G. B. Orr and K.-R. Müller, editors, *Neural Networks: Tricks of the Trade*, pages 9–50. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[70] Y. LeCun, J. S. Denker, and S. A. Solla. Optimal brain damage. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 598–605. Morgan-Kaufmann, 1990.

[71] J.-B. Lee and H. Kalva. Video coding techniques and standards. In B. Furht, editor, *Encyclopedia of Multimedia*, pages 899–904. Springer US, Boston, MA, 2008.

[72] B. Li, H. Li, L. Li, and J. Zhang. λ domain rate control algorithm for high efficiency video coding. *IEEE Transactions on Image Processing*, 23(9):3841–3854, 2014.

[73] J. Li, B. Li, J. Xu, and R. Xiong. Intra prediction using fully connected network for video coding. In *2017 IEEE International Conference on Image Processing (ICIP)*, 2017.

[74] J. Li, B. Li, J. Xu, R. Xiong, and W. Gao. Fully connected network-based intra prediction for image coding. *IEEE Transactions on Image Processing*, 27(7):3236–3247, 2018.

[75] L. Li, J. Heo, J. Choi, J. Choi, S. Yoo, S. Kim, and J. Lim. CE3-6.2.1: Extended MPM list. Technical report, JVET 12th Meeting, document JVET-L0165-v4, Macao, MO, October 2018.

[76] T. Li, M. Xu, and X. Deng. A deep convolutional neural network approach for complexity reduction on intra-mode HEVC. In *2017 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1255–1260, 2017.

[77] X. Li, H.-C. Chuang, J. Chen, and M. Karczewicz. Multi-type-tree. Technical report, JVET 4th Meeting, document JVET-D0117, Chengdu, CN, October 2016.

[78] X. Li, N. Oertel, A. Hutter, and A. Kaup. Laplace distribution based lagrangian rate distortion optimization for hybrid video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(2):193–205, 2009.

[79] Y. Li, B. Li, D. Liu, and Z. Chen. A convolutional neural network-based approach to rate control in HEVC intra coding. In *2017 IEEE Visual Communications and Image Processing (VCIP)*, 2017.

[80] Y. Li, L. Li, Z. Li, J. Yang, N. Xu, D. Liu, and H. Li. A hybrid neural network for chroma intra prediction. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 1797–1801, 2018.

[81] Y. Li, D. Liu, H. Li, L. Li, F. Wu, H. Zhang, and H. Yang. Convolutional neural network-based block up-sampling for intra frame coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(9):2316–2330, 2018.

[82] Z. Li, A. Aaron, I. Katsavounidis, A. Moorthy, and M. Manohara. Toward a practical perceptual video quality metric. `https://netflixtechblog.com/toward-a-practical-perceptual-video-quality-metric-653f208b9652`, 2016. Online; accessed June 27, 2021.

[83] J. Lin, Y. Chen, Y. Tsai, Y. Huang, and S. Lei. Motion vector coding techniques for HEVC. In *2011 IEEE 13th International Workshop on Multimedia Signal Processing*, 2011.

[84] J. Lin, D. Liu, H. Li, and F. Wu. Generative adversarial network-based frame extrapolation for video coding. In *2018 IEEE Visual Communications and Image Processing (VCIP)*, 2018.

[85] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *Computer Vision – ECCV 2014*, pages 740–755, 2014.

[86] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala. Video frame synthesis using deep voxel flow. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 4473–4481, 2017.

[87] Z. Liu, X. Yu, Y. Gao, S. Chen, X. Ji, and D. Wang. CU partition mode decision for HEVC hardwired intra encoder using convolution neural network. *IEEE Transactions on Image Processing*, 25(11):5088–5103, 2016.

[88] W. Lotter, G. Kreiman, and D. D. Cox. Deep predictive coding networks for video prediction and unsupervised learning. In *5th International Conference on Learning Representations (ICLR)*, 2017.

[89] G. Luz, J. Ascenso, C. Brites, and F. Pereira. Saliency-driven omnidirectional imaging adaptive coding: Modeling and assessment. In *2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP)*, 2017.

[90] C. Ma, D. Liu, X. Peng, L. Li, and F. Wu. Convolutional neural network-based arithmetic coding for HEVC intra-predicted residues. *IEEE Transactions on Circuits and Systems for Video Technology*, 2019.

[91] C. Ma, D. Liu, X. Peng, and F. Wu. Convolutional neural network-based arithmetic coding of DC coefficients for HEVC intra coding. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 1772–1776, 2018.

[92] C. Ma, D. Liu, X. Peng, Z. Zha, and F. Wu. Neural network-based arithmetic coding for inter prediction information in HEVC. In *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2019.

[93] D. Ma, F. Zhang, and D. R. Bull. Video compression with low complexity CNN-based spatial resolution adaptation, 2020.

[94] S. Ma and C.-C. H. Kuo. High-definition video coding with super-macroblocks. In C. W. Chen, D. Schonfeld, and J. Luo, editors, *Visual Communications and Image Processing 2007*, volume 6508, pages 417 – 428. International Society for Optics and Photonics, SPIE, 2007.

[95] E. Maggio and A. Cavallaro. *Video Tracking: Theory and Practice.* Wiley Publishing, 1st edition, 2011.

[96] D. Marpe, H. Schwarz, and T. Wiegand. Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):620–636, 2003.

[97] J. McConnell. An invisible tax on the web: Video codecs. `https://blog.mozilla.org/blog/2018/07/11/royalty-free-web-video-codecs`, 2018. Online; accessed November 15, 2020.

[98] M. Meyer, J. Wiesner, J. Schneider, and C. Rohlfing. Convolutional neural networks for video intra prediction using cross-component adaptation. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1607–1611, 2019.

[99] A. Minezawa, K. Sugimoto, and Sekiguchi S. An improved intra vertical and horizontal prediction. Technical report, JCT-VC 6th Meeting, document JCTVC-F172, Torino, IT, July 2011.

[100] R. Molina, A. K. Katsaggelos, L. D. Alvarez, and J. Mateos. Toward a new video compression scheme using super-resolution. In J. G. Apostolopoulos and A. Said, editors, *Visual Communications and Image Processing 2006*, volume 6077, pages 67 – 79. International Society for Optics and Photonics, SPIE, 2006.

[101] C. Molnar. *Interpretable Machine Learning.* 2019. `https://christophm.github.io/interpretable-ml-book/`.

[102] MPEG. ISO/IEC 11172 - coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s. Technical report, 1993.

[103] M. Mrak and J. Xu. Improving screen content coding in HEVC by transform skipping. In *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, pages 1209–1213, 2012.

[104] W. J. Murdoch, C. Singh, K. Kumbier, R. Abbasi-Asl, and B. Yu. Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, 116(44):22071–22080, 2019.

[105] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

[106] T. Nguyen, P. Helle, M. Winken, B. Bross, D. Marpe, H. Schwarz, and T. Wiegand. Transform coding techniques in HEVC. *IEEE Journal of Selected Topics in Signal Processing*, 7(6):978–989, 2013.

[107] V.-A. Nguyen, Y.-P. Tan, and W. Lin. Adaptive downsampling/upsampling for better video compression at low bit rate. In *2008 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1624–1627, 2008.

[108] Ofcom. Uk home broadband performance, measurement period may 2020 – interactive report. `https://www.ofcom.org.uk/research-and-data/telecoms-research/broadband-research/may-2020-uk-home-broadband-performance`, 2020. Online; accessed November 12, 2020.

[109] J. Ohm. *Multimedia Communication Technology: Representation,Transmission and Identification of Multimedia Signals*. Signals and Communication Technology. Springer Berlin Heidelberg, 2003.

[110] G. Pavlidis. *Mixed Raster Content: Segmentation, Compression, Transmission*. Signals and Communication Technology. Springer Singapore, 2016.

[111] J. Pfaff, P. Helle, D. Maniry, S. Kaltenstadler, W. Samek, H. Schwarz, D. Marpe, and T. Wiegand. Neural network based intra prediction for video coding. In A. G. Tescher, editor, *Applications of Digital Image Processing XLI*, volume 10752, pages 359 – 365. International Society for Optics and Photonics, SPIE, 2018.

[112] T. Phillips. *The Complete Guide to Fujifilm's X-T3 (B&W Edition)*. LULU Press, 2019.

[113] C. Poynton. *Digital Video and HD: Algorithms and Interfaces*. Morgan Kaufmann Series in Computer Graphics. Elsevier Science, 2012.

[114] S. Puri, S. Lasserre, and P. Le Callet. CNN-based transform index prediction in multiple transforms framework to assist entropy coding. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 798–802, 2017.

[115] R. Rojas. *Neural Networks: A Systematic Introduction*. Springer-Verlag, Berlin, Heidelberg, 1996.

[116] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention (MICCAI 2015)*, pages 234–241, Cham, 2015. Springer International Publishing.

[117] C. Rosewarne, B. Bross, M. Naccari, K. Sharman, and G. Sullivan. High efficiency video coding (HEVC) test model 16 (HM 16) improved encoder description update 5. Technical report, document JCTVC-W1002, 23rd Meeting, San Diego, US, February 2016.

[118] A. Said, X. Zhao, M. Karczewicz, J. Chen, and F. Zou. Position dependent prediction combination for intra-frame video coding. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 534–538, 2016.

[119] M. Santamaria. Estimation of rate control parameters for video coding using CNN. `hhttps://github.com/bbc/cnn-rate-distortion/tree/670826aff3c6232b5b4399f9b80f2078c606f41c`, 2020. GitHub repository.

[120] M. Schäfer, B. Stallenberger, J. Pfaff, P. Helle, H. Schwarz, D. Marpe, and T. Wiegand. An affine-linear intra prediction with complexity constraints. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 1089–1093, 2019.

[121] R. Schäfer and T. Sikora. Digital video coding standards and their role in video communications. *Proceedings of the IEEE*, 83(6):907–924, 1995.

[122] R. Schäfer, T. Wiegand, and H. Schwarz. The emerging H.264/AVC standard. *EBU Technical Review*, 2003.

[123] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.

[124] M. Sharma. Compression using Huffman coding. *International Journal of Computer Science and Network Security (IJCSNS)*, 10(5):133–141, 2010.

[125] Y. Shoham and A. Cersho. Efficient codebook allocation for an arbitrary set of vector quantizers. In *ICASSP '85. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 10, pages 1696–1699, 1985.

[126] A. Skodras, C. Christopoulos, and T. Ebrahimi. The JPEG 2000 still image compression standard. *IEEE Signal Processing Magazine*, 18(5):36–58, 2001.

[127] N. Song, Z. Liu, X. Ji, and D. Wang. CNN oriented fast PU mode decision for HEVC hardwired intra encoder. In *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 239–243, 2017.

[128] R. Song, D. Liu, H. Li, and F. Wu. Neural network-based arithmetic coding of intra prediction modes in HEVC. In *2017 IEEE Visual Communications and Image Processing (VCIP)*, 2017.

[129] Spin Digital Video Technologies GmbH. Next generation video. `https://spin-digital.com/technology/next-generation-video`. Online; accessed November 9, 2020.

[130] 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*, 22(12):1649–1668, 2012.

[131] G. J. Sullivan and T. Wiegand. Rate-distortion optimization for video compression. *IEEE Signal Processing Magazine*, 15(6):74–90, 1998.

[132] V. Sze, M. Budagavi, and G. J. Sullivan. *High Efficiency Video Coding (HEVC): Algorithms and Architectures*. Springer International Publishing, 2014.

[133] T. K. Tan, R. Weerakkody, M. Mrak, N. Ramzan, V. Baroncini, J. Ohm, and G. J. Sullivan. Video quality evaluation methodology and verification testing of HEVC compression performance. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(1):76–90, 2016.

[134] D. Taubman and M. Marcellin. *JPEG2000: Image Compression Fundamentals, Standards and Practice*. Springer Publishing Company, Incorporated, 2013.

[135] The European Commission's science and knowledge service. Coronavirus pandemic reveals large differences in the prevalence of telework across the eu. `https://ec.europa.eu/jrc/en/news/coronavirus-pandemic-reveals-large-differences-prevalence-telework-across-eu`, 2020. Online; accessed November 8, 2020.

[136] R. Timofte, E. Agustsson, L. Van Gool, M.-H. Yang, L. Zhang, B. Lim, et al. Ntire 2017 challenge on single image super-resolution: Methods and results. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2017.

[137] F. Toutounchi. *Super-Resolution in Still Images and Videos via Deep Learning*. PhD thesis, Queen Mary University of London, 2019.

[138] F. Toutounchi and E. Izquierdo. Advanced super-resolution using lossless pooling convolutional networks. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1562–1568, 2019.

[139] P. N. Tudor. MPEG-2 video compression. *Electronics Communication Engineering Journal*, 7(6):257–264, 1995.

[140] Twitch Tracker. Twitch statistics & charts. `https://twitchtracker.com/statistics`. Online; accessed November 15, 2020.

[141] K. Ugur, A. Alshin, E. Alshina, F. Bossen, W. Han, J. Park, and J. Lainema. Interpolation filter design in HEVC and its coding efficiency - complexity analysis. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1704–1708, 2013.

[142] K. Ugur, A. Alshin, E. Alshina, F. Bossen, W. Han, J. Park, and J. Lainema. Motion compensated prediction and interpolation filter design in H.265/HEVC. *IEEE Journal of Selected Topics in Signal Processing*, 7(6):946–956, 2013.

[143] K. Ugur and J. Lainema. Updated results on HEVC still picture coding performance. Technical report, JCT-VC 13th Meeting, document JCTVC-M0041, Incheon, KR, April 2013.

[144] M. Wang, K. N. Ngan, and H. Li. An efficient frame-content based intra frame rate control for high efficiency video coding. *IEEE Signal Processing Letters*, 22(7):896–900, July 2015.

[145] T. Wang, M. Chen, and H. Chao. A novel deep learning-based method of improving coding efficiency from the decoder-end for HEVC. In *2017 Data Compression Conference (DCC)*, pages 410–419, 2017.

[146] Y. Wang, X. Fan, C. Jia, D. Zhao, and W. Gao. Neural network based inter prediction for HEVC. In *2018 IEEE International Conference on Multimedia and Expo (ICME)*, 2018.

[147] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[148] Z. Wang, S. Wang, X. Zhang, S. Wang, and S. Ma. Fast QTBT partitioning decision for interframe coding with convolution neural network. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 2550–2554, 2018.

[149] R. Weerakkody and M. Mrak. High efficiency video coding for ultra high definition television. In *NEM Summit*, pages 9–14, 2013.

[150] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):560–576, 2003.

[151] M. Wien. *High Efficiency Video Coding: Coding Tools and Specification.* Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.

[152] S. Winkler, M. Kunt, and C. J. van den Branden Lambrecht. Vision and video: Models and applications. In C. J. van den Branden Lambrecht, editor, *Vision Models and Applications to Image and Video Processing*, pages 201–229. Springer US, Boston, MA, 2001.

[153] A. Wright. YouTube for Android TV adopts AV1 video codec in certain devices. `https://www.xda-developers.com/youtube-for-android-tv-adopts-av1-video-codec-in-certain-devices`, 2020. Online; accessed June 13, 2020.

[154] B. Xu, X. Pan, Y. Zhou, Y. Li, D. Yang, and Z. Chen. CNN-based rate-distortion modeling for H.265/HEVC. In *2017 IEEE Visual Communications and Image Processing (VCIP)*, 2017.

[155] M. Xu, T. Li, Z. Wang, X. Deng, R. Yang, and Z. Guan. Reducing complexity of HEVC: A deep learning approach. *IEEE Transactions on Image Processing*, 27(10):5044–5059, 2018.

[156] N. Yan, D. Liu, H. Li, B. Li, L. Li, and F. Wu. Invertibility-driven interpolation filter for video coding. *IEEE Transactions on Image Processing*, 28(10):4912–4925, 2019.

[157] N. Yan, D. Liu, H. Li, and F. Wu. A convolutional neural network approach for half-pel interpolation in video coding. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2017.

[158] N. Yan, D. Liu, H. Li, T. Xu, F. Wu, and B. Li. Convolutional neural network-based invertible half-pixel interpolation filter for video coding. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 201–205, 2018.

[159] R. Yang, M. Xu, T. Liu, Z. Wang, and Z. Guan. Enhancing quality for HEVC compressed videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 2018.

[160] H. Zhang, L. Song, Z. Luo, and X. Yang. Learning a convolutional neural network for fractional interpolation in HEVC inter coding. In *2017 IEEE Visual Communications and Image Processing (VCIP)*, 2017.

[161] T. Zhang and S. Mao. An overview of emerging video coding standards. *GetMobile: Mobile Comp. and Comm.*, 22(4):13–20, 2019.

[162] Y. Zhang, T. Shen, X. Ji, Y. Zhang, R. Xiong, and Q. Dai. Residual highway convolutional neural networks for in-loop filtering in HEVC. *IEEE Transactions on Image Processing*, 27(8):3827–3841, 2018.

[163] L. Zhao, S. Wang, X. Zhang, S. Wang, S. Ma, and W. Gao. Enhanced motion-compensated video coding with deep virtual reference frame generation. *IEEE Transactions on Image Processing*, 28(10):4832–4844, 2019.

[164] L. Zhao, X. Zhao, S. Liu, X. Li, J. Lainema, G. Rath, F. Urban, and F. Racapé. Wide angular intra prediction for versatile video coding. In *2019 Data Compression Conference (DCC)*, pages 53–62, 2019.

[165] Z. Zhao, S. Wang, S. Wang, X. Zhang, S. Ma, and J. Yang. Enhanced bi-prediction with convolutional neural network for high-efficiency video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(11):3291–3301, 2019.

[166] Y. Zheng, M. Coban, and M. Karczewicz. Simplified intra smoothing. Technical report, JCT-VC 3rd Meeting, document JCTVC-C234 WG11 Number: M18274, Guangzhou, CN, October 2010.

[167] I. Zupancic, E. Izquierdo, M. Naccari, and M. Mrak. Two-pass rate control for UHDTV delivery with HEVC. In *2016 Picture Coding Symposium (PCS)*, 2016.