Video Compression with CNN-based Post Processing

Fan Zhang, *Member, IEEE*, Di Ma, Chen Feng, and David R. Bull, *Fellow, IEEE*

***Abstract*—In recent years, video compression techniques have been significantly challenged by the rapidly increased demands associated with high quality and immersive video content. Among various compression tools, post-processing can be applied on reconstructed video content to mitigate visible compression artefacts and to enhance overall perceptual quality. Inspired by advances in deep learning, we propose a new CNN-based post-processing approach, which has been integrated with two state-of-the-art coding standards, VVC and AV1. The results show consistent coding gains on all tested sequences at various spatial resolutions, with average bit rate savings of 4.0% and 5.8% against original VVC and AV1 respectively (based on the assessment of PSNR). This network has also been trained with perceptually inspired loss functions, which have further improved reconstruction quality based on perceptual quality assessment (VMAF), with average coding gains of 13.9% over VVC and 10.5% against AV1.**

***Index Terms*—CNN, VVC, AV1, video compression, perceptual quality, GAN**

I. Introduction

The importance of video compression has come to the fore over the past decade driven by the tension between the huge quantities of video content consumed everyday and the bandwidth available to transmit it. This challenge has been addressed through the creation of new video coding standards, the latest activity being by the Joint Video Exploration Team (JVET), who published the first version of H.266/Versatile Video Coding (VVC) [1] in 2020. Compared to its predecessor, H.265/High Efficiency Video Coding (HEVC), VVC has achieved up to 50% performance improvement through the adoption of numerous sophisticated coding tools, in particular with improved support for formats with high spatial resolutions, high dynamic range and spherical content. Alongside VVC, the Alliance for Open Media (AOMedia) also published its first video coding format, AOMedia Video (AV1) in 2018, which has been reported to offer comparable coding performance to VVC [2].

Machine learning, especially based on deep convolutional neural networks (CNNs), has been increasingly applied in the context of video compression and has achieved promising results both when used in conjunction with conventional coding algorithms and in the form of new end-to-end architectures. In addition to conventional normative coding tools, deep learning can also be employed at the video decoder as a post-processing stage to further reduce noticeable artefacts and enhance the visual quality of compressed content. For the state-of-the-art coding standards, such as VVC and AV1, most existing learning-based post-processing approaches can only offer evident improvement for less efficient coding configurations (e.g. intra coding), and the employed networks are normally trained to minimise average absolute pixel distortions rather than explicitly to improve perceptual quality.

Based on the Generative Adversarial Network (GAN) paradigm, we propose a novel CNN-based post-processing approach with an extension that achieves improved perceptual reconstruction quality. This approach has been evaluated on the VVC Test Model (VTM) 4.0.1 and on AV1 libaom 1.0.0, with results showing consistent improvement on standard JVET test sequences for different QP values based on different quality measurements. We have further analysed the computational complexities of different CNN structure variants and correlated them with overall coding gains.

This paper is a comprehensive extension of our previous work [3], which solely focused on the PSNR driven optimization of VVC compressed content. The primary differences are summarized below:

- The CNN model used for post-processing has been extensively upgraded with a new GAN-based perceptual training strategy, which can significantly improve the perceptual quality of the final reconstructed content compared to [3].
- This CNN-based post processing approach has been also trained and evaluated on AV1 compressed results (alongside VVC in [3]) and achieved similar coding gains.

In the remainder of the paper, we first survey the prior work in the field of deep video compression and, in particular, describe learning-based post-processing approaches. Secondly, we present our proposed CNN-based post-processing approach, describing the network architecture and how it was trained and evaluated. We then summarize and discuss the experimental results, highlighting the performance improvements over standard video codecs. Finally we conclude the paper and outline possible future work.

II. Prior Work

A. Deep video compression

In the past few years, machine learning, in particular deep neural networks, has been increasingly applied to image and video compression, demonstrating significant potential compared to conventional coding methods. Learning-based video

The authors would like to acknowledge funding from UK EPSRC (EP/L016656/1 and EP/M000885/1) and the NVIDIA GPU Seeding Grants. We also appreciate the valuable advice provided by Dr. Debargha Mukherjee on AV1 coding configuration.

Fan Zhang, Di Ma, Chen Feng, and David R. Bull are with the Bristol Vision Institute, University of Bristol, Bristol, UK.

coding algorithms can be classified into two primary groups: new end-to-end network architectures and those that enhance individual conventional coding tools.

Machine learning has been employed to refine existing coding tools within a conventional coding framework, including intra and inter prediction, transformation, quantization and in-loop filtering [4]. New coding tools have also been developed with the support of neural networks, such as CNN-based spatial resolution and bit depth adaptation [5]. Moreover, the classic hybrid video coding framework has been challenged by new deep network architectures, which enable end-to-end training and optimization [6, 7]. This latter approach often employs a general rate distortion framework with non-linear transforms, which are based on convolutional filters and nonlinear activation functions. Although these solutions show great promise as an alternative to conventional codecs, their performance cannot still compete with the latest standardized video codecs, including VVC and AV1.

B. *CNN-based post processing*

Compression processes often introduce various visible artefacts such as blocking mismatches, banding and blurring, especially when large quantization steps are employed. These unpleasant distortions can be mitigated by filtering the reconstructed frames. When this enhancement process is performed outside of the encoding loop (generally after decoding), it is referred to as post-processing.

In standardized codecs, filters have also been designed for use within the encoding loop to reduce compression artefacts. VVC employs three different types in-loop filters including de-blocking filters (DBF), sample adaptive offset (SAO) and adaptive loop filters (ALF) [1]. In AV1, in addition to deblocking filters, there are two other filtering operations: constrained directional enhancement filtering and loop restoration filtering [2].

CNNs are now also playing an important role in image restoration (including super-resolution), and these approaches can also be employed for post-processing of compressed video content to improve the overall reconstruction quality. Although various researchers have implemented CNN-based post-processing approaches in the context of HEVC and VVC [8–11], most of these can only achieve coding gains for All Intra configurations, which offer lower coding efficiency compared to Random Access configurations based on hierarchical B frame structures. In addition, all the employed CNN models in these approaches were trained to optimize a simple loss function based on pixel distortions ($\ell1$ or $\ell2$ loss), which can lead to over-smoothed reconstruction results.

III. PROPOSED ALGORITHM

Figure 1 shows a high level coding workflow with a CNN-based post-processing module. This section focuses on the structure of the employed CNN architecture, and the details of network training and evaluation.

A. *The CNN architecture*

The CNN architecture used in this work is illustrated in **Figure 2**.

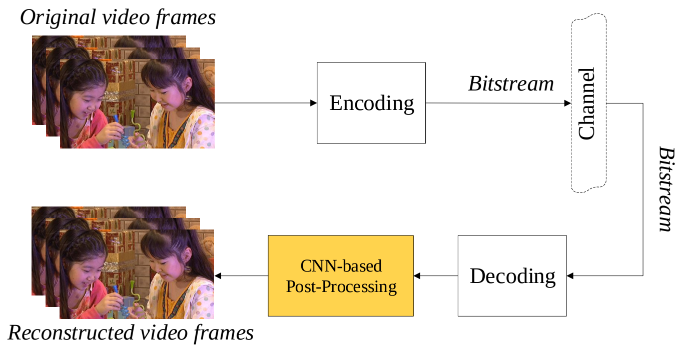


Fig. 1: A typical coding workflow with a CNN-based post-processing module.

1) The generator network: is a modified version based on the generator (SRResNet) of SRGAN [12], and this has been previously employed by the authors in video compression systems based on spatial resolution and bit depth resampling [5]. This network takes a 96×96 RGB (compressed) image block as input and produces an image block with the same format, targeting to its corresponding original (uncompressed) counterpart[1].

Residual blocks (RB) form the basic unit in this network, which contains two convolutional layers and a parametric Rectified Linear Unit (PReLU) activation function in between them. A skip connection is used between the input of each RB and the output of its second convolutional layer. The number of residual blocks is configurable and was set to 16 in this work.

The input of the network is connected to these successive RBs through a convolutional layer (also with a ReLU). Between the network output and the output of the last RB, there is also a convolution layer (output layer) followed by a Tanh activation function. An additional skip connection is used between the output of the input layer and the output of the last RB. A long skip connection is also employed between the input the first RB and the output of the output layer to produce the final output.

2) The discriminator: used in this work is similar to that in SRGAN [12], which takes the output of the generator (*fake*) and compares to its corresponding original (*real*). This network consists of one input layer (with a Leaky ReLU), seven identical convolutional layers, and two dense layers. Each of the convolutional layers is followed by a batch normalization layer and a leaky ReLU activation function. After the second dense layer, a Sigmoid activation function is employed to output a probability to predict how much the quality of the *real* image block is perceptually better than the *fake* one.

The parameters used in each convolutional layer including kernel sizes, feature map numbers and stride values are shown in Figure 2.

[1]We have used the same input/output formats as in SRResNet.

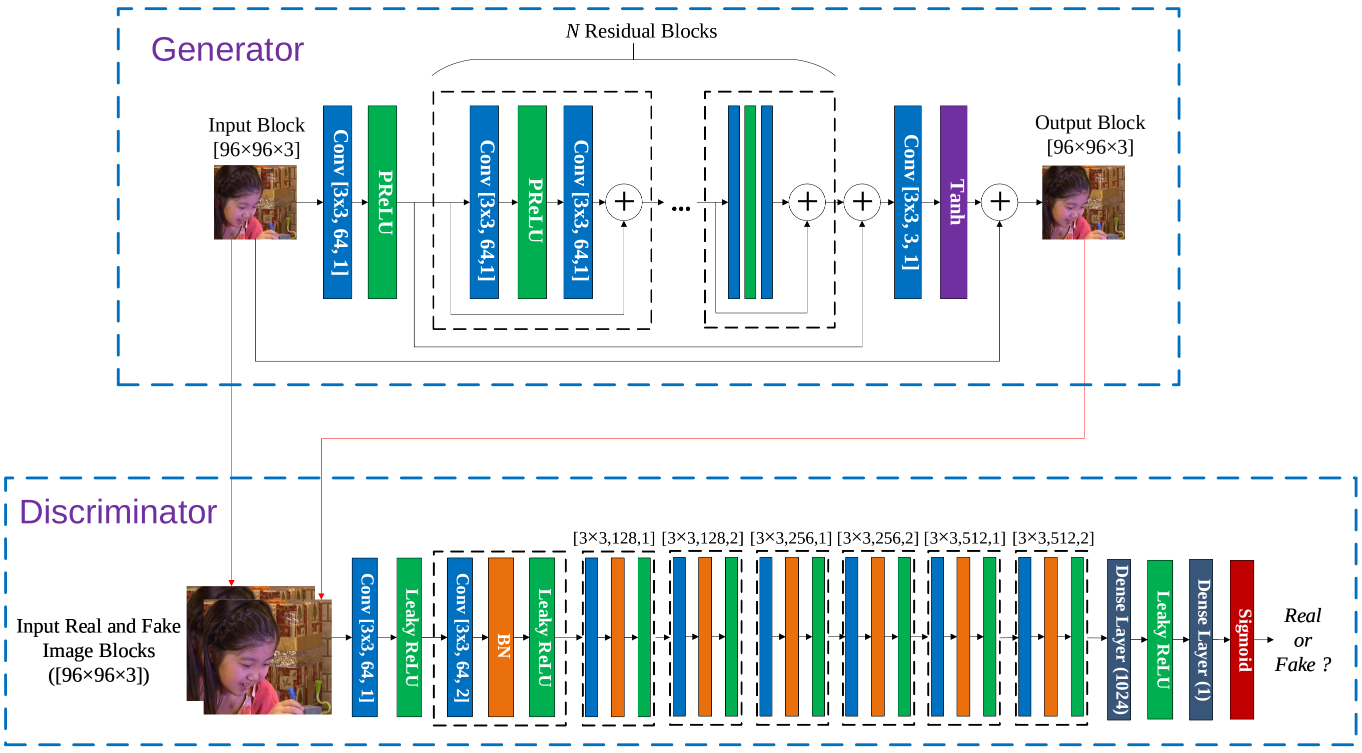


Fig. 2: The employed GAN architecture, comprising generator and discriminator stages

B. *Training database*

The training material is essential for learning-based algorithms. We need to ensure the training content is diverse and covers various texture types in order to achieve good model generalisation and avoid potential over-fitting problems. To train the employed network, we have selected 432 uncompressed video sequences from a publicly available training database, BVI-DVC [13], which was designed specifically for deep video compression. All these sequences have the same frame rate of 60 frames per second, YCbCr 4:2:0 format, and with four different spatial resolutions including 3840×2160, 1920×1080, 960×540, and 480×270. We have encoded these 432 original sequences using VVC VTM 4.0.1 and AV1 libaom 1.0.0 with the coding configurations summarized in **Table I**.

For each codec, the reconstructed video frames for each QP value and their corresponding originals were randomly selected and segmented into 96×96 colour image blocks (after converting to the RGB space from YCbCr 4:2:0). We have also rotated selected image blocks to further improve data diversity. As a result, for each video codec and QP group[2], there are over 100,000 image blocks pairs (compressed and original).

C. *Training strategy*

We trained this network using two different methodologies: (i) only train the generator using $\ell 1$ loss (mean absolute difference) (ii) jointly train both the generator and the discriminator based on perceptually-inspired loss functions. The CNN models obtained by these two training methods are used to post-process VVC and AV1 compressed content in the evaluation experiments, and their results are compared in the next section.

1) $\ell 1$ loss: is firstly employed to train the network (generator only) using the material generated for each QP group and codec. This results in a total number of nine CNN models for different evaluation scenarios:

$$\begin{cases} \mathrm{CNN}_{\mathrm{VVC,QP22}}, & \text{if} \quad \mathrm{QP}_{\mathrm{eval}} \leq 24.5 \\ \mathrm{CNN}_{\mathrm{VVC,QP27}}, & \text{if} \quad 24.5 < \mathrm{QP}_{\mathrm{eval}} \leq 29.5 \\ \mathrm{CNN}_{\mathrm{VVC,QP32}}, & \text{if} \quad 29.5 < \mathrm{QP}_{\mathrm{eval}} \leq 34.5 \\ \mathrm{CNN}_{\mathrm{VVC,QP37}}, & \text{if} \quad 34.5 < \mathrm{QP}_{\mathrm{eval}} \leq 39.5 \\ \mathrm{CNN}_{\mathrm{VVC,QP42}}, & \text{if} \quad \mathrm{QP}_{\mathrm{eval}} > 39.5 \end{cases} \tag{1}$$

$$\begin{cases} \mathrm{CNN}_{\mathrm{AV1,QP32}}, & \text{if} \quad \mathrm{QP}_{\mathrm{eval}} \leq 37.5 \\ \mathrm{CNN}_{\mathrm{AV1,QP43}}, & \text{if} \quad 37.5 < \mathrm{QP}_{\mathrm{eval}} \leq 49 \\ \mathrm{CNN}_{\mathrm{AV1,QP55}}, & \text{if} \quad 49 < \mathrm{QP}_{\mathrm{eval}} \leq 59 \\ \mathrm{CNN}_{\mathrm{AV1,QP63}}, & \text{if} \quad \mathrm{QP}_{\mathrm{eval}} > 59 \end{cases} \tag{2}$$

Here $\mathrm{QP}_{\mathrm{eval}}$ represents the base QP value employed in the evaluation phase for the two different codecs, and $\mathrm{CNN}_{c,q}$ is the CNN model trained for different codecs (VVC or AV1) and QP values.

2) Perceptual loss functions: have been employed to train the whole GAN architecture following a two stage training strategy. This was initially designed to train the Relativistic GAN for image generation [15], and has also been used to train the CNN models for spatial resolution and bit depth up-sampling [16, 17]. In the first stage, the generator is trained separately using the multi-scale structural similarity index (MS-SSIM) [18] as the loss function. The trained generator

[2]Based on our preliminary results, we note that, through QP sub-grouping, additional coding gains (approximately 0.05dB based on PSNR) can be achieved compared to using a single CNN model.

TABLE I: The coding configuration employed for VVC VTM and AV1 libaom.

Codec	Version	Configuration parameters
VVC VTM	4.0.1	Random Access configuration [14]. IntraPeriod=64, GOPSize=16, QP=22, 27, 32, 37, 42.
AV1 libaom	1.0.0-5ec3e8c (02/05/2020)	--i420 --psnr --usage=0 --verbose --cpu-used=0 --threads=0 --profile=0 --width=$w --height=$h --input-bit-depth=10 --bit-depth=10 --fps=$fps/1001 --passes=1 --kf-max-dist=64 --kf-min-dist=64 --drop-frame=0 --static-thresh=0 --arnr-maxframes=7 --arnr-strength=5 --lag-in-frames=19 --aq-mode=0 --bias-pct=100 --minsection-pct=1 --maxsection-pct=10000 --auto-alt-ref=1 --min-q=0 --max-q=63 --max-gf-interval=16 --min-gf-interval=4 --frame-parallel=0 --color-primaries=bt709 --end-usage=q --sharpness=0 --undershoot-pct=100 --overshoot-pct=100 --tile-columns=0 --cq-level={32, 43, 55, 63} w/o --enable-fwd-kf=1

model is employed as the starting point when the generator and discriminator are trained together in the second phase.

The generator is trained using a combined loss function, $\mathcal{L}_{\text{gen}}$ in the second stage:

$$\mathcal{L}_{\text{gen}} = \mathcal{L}_{\text{SSIM}} + \alpha \cdot \mathcal{L}_{\ell 1} + \beta \cdot \mathcal{L}_G^a \tag{3}$$

in which $\mathcal{L}_{\text{SSIM}}$ stands for the SSIM [19] loss (1-SSIM) between the generator output and the target, while $\mathcal{L}_{\ell 1}$ is the $\ell 1$ loss between them. $\mathcal{L}_G^a$ is defined as the adversarial loss for the generator:

$$\begin{aligned}\mathcal{L}_G^a = &- E_{I_r}[\ln(1-(\text{Sig}(O_d(I_r) - E_{I_f}[O_d(I_f)])))] \\ &- E_{I_f}[\ln(\text{Sig}(O_d(I_f) - E_{I_r}[O_d(I_r)]))]\end{aligned} \tag{4}$$

Here I_r and I_f are denoted as the *real* and *fake* image blocks respectively. $E_{I_r}[\cdot]$ represents the mean operation for all the *real* (*fake* if I_r is replaced by I_f) image blocks, and $O_d(\cdot)$ is the output of the discriminator. 'Sig' represents the Sigmoid function.

For the discriminator, the loss function $\mathcal{L}_D$ is given by (5):

$$\begin{aligned}\mathcal{L}_D = &- E_{I_r}[\ln(\text{Sig}(O_d(I_r) - E_{I_f}[O_d(I_f)]))] \\ &- E_{I_f}[\ln(1-(\text{Sig}(O_d(I_f) - E_{I_r}[O_d(I_r)])))]\end{aligned} \tag{5}$$

3) The training configuration: is summarized as follows. We implemented all networks based on the TensorFlow 1.8.0 framework, and set the learning rate and weight decay to 0.0001 and 0.1 (for every 100 epochs) respectively for both training stages. The total number of training epochs are 200. We have used Adam optimisation algorithm during the training with the hyper parameters of $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The two weights α and β in equation (3) are set up to 0.025 and 5×10^{-3} respectively.

D. Evaluation operation

In the evaluation stage, when we use the trained CNN models to enhance compressed video frames, each frame is segmented into 96×96 overlapping image blocks with an overlap of 4 pixels. These blocks are converted to RGB color space as the input of CNN models, and the output image blocks are then aggregated in the same way through simple blending to form the final video frame. Here only the generator network is used in the evaluation (the discriminator is for training only).

IV. RESULTS AND DISCUSSION

The proposed post-processing approach has been utilised to enhance both VVC and AV1 compressed content. We have used all 19 test sequences from the JVET CTC standard dynamic range (SDR) testset. None of these sequences were included in the CNN training database.

VVC compressed content was generated using VTM 4.0.1 with the Random Access configuration. AV1 libaom 1.0.0 was used to produce AV1 content, with similar coding parameters to those for VVC. The employed configurations for both codecs are summarised in Table I. The tested QP values are 22, 27, 32, 37 and 42 for VVC, and 32, 43, 55 and 63 for AV1.

The final video quality was evaluated using two assessment methods, Peak Signal-to-Noise ratio (PSNR) and Video Multimethod Assessment Fusion (VMAF) [20]. PSNR is the most commonly used quality metric in the video compression community, while VMAF is a machine learning based metric, which combines multiple existing quality metrics and a video feature using a Support Vector Machine regression approach. Compared to PSNR, VMAF has been reported to provide more accurate prediction of perceptual quality. The performance of the proposed approach has been compared with two original codecs using the Bjøntegaard Delta rate measurements (BD-rate) [21]. For VVC, the compression performance is evaluated for both low (22-37) and high QP (27-42) ranges.

In order to further benchmark the performance of the proposed algorithm, another two state-of-the-art CNN-based post-processing approaches, denoted as JVET-N0254 [10] and JVET-O0079 [11], are also compared here in the context of VVC (for low QP range only). Results of both are based on the RA configuration and were submitted to MPEG JVET meetings as VVC proposals.

A. Compression performance

Table II and **III** summarize the compression performance of the proposed method when it is applied to VVC and AV1 compressed content. For $\ell 1$ trained CNNs, we note that the average bit-rate savings according to PSNR are 3.9% and 5.8% against the original VVC and AV1 respectively. If the perceptual quality metric, VMAF, is used to assess video quality, the coding gains are 4.2% over VVC and 2.7% over AV1. When we use perceptual loss function trained models for post-processing, the coding gains appear much more significant based on the assessment of VMAF – 13.9% and 10.5% over VVC and AV1 respectively. This is particularly

TABLE II: The compression performance of the proposed method benchmarked on the original VVC VTM 4.0.1. Negative BD-rate values indicate coding gains.

Method	VTM-PP ($\ell1$ loss)				VTM-PP (Perceptual loss)			
Metric	PSNR		VMAF		PSNR		VMAF	
QP Range	H-QPs	L-QPs	H-QPs	L-QPs	H-QPs	L-QPs	H-QPs	L-QPs
Class-Sequence	BD-rate	BD-rate	BD-rate	BD-rate	BD-rate	BD-rate	BD-rate	BD-rate
A1-Campfire	-3.3%	-2.3%	-5.6%	-4.6%	+0.2%	-0.4%	-10.4%	-10.7%
A1-FoodMarket4	-2.6%	-2.0%	-3.8%	-3.0%	-0.0%	+0.1%	-8.4%	-7.4%
A1-Tango2	-3.3%	-2.9%	-3.4%	-3.0%	-1.1%	-0.6%	-7.8%	-9.3%
A2-CatRobot1	-5.2%	-5.2%	-4.6%	-4.4%	-0.6%	-1.1%	-14.4%	-17.8%
A2-DaylightRoad2	-6.0%	-7.1%	-6.8%	-7.2%	-1.1%	-2.1%	-19.5%	-23.4%
A2-ParkRunning3	-0.8%	-0.4%	-2.3%	-0.2%	+2.1%	+2.4%	-11.7%	-11.1%
Class A	-3.5%	-3.3%	-4.4%	-3.7%	-0.1%	+0.3%	-10.1%	-13.3%
B-BQTerrace	-2.2%	-1.0%	-6.1%	-1.1%	+2.0%	+0.3%	-23.3%	-28.8%
B-BasketballDrive	-3.4%	-3.1%	-1.8%	+2.7%	-0.9%	-0.9%	-7.9%	-8.7%
B-Cactus	-3.4%	-3.0%	-5.1%	-4.4%	+0.2%	-0.2%	-15.8%	-17.1%
B-MarketPlace	-2.6%	-2.3%	-4.8%	-4.0%	+1.2%	+0.3%	-17.7%	-18.2%
B-RitualDance	-3.8%	-3.5%	-4.6%	-2.6%	-1.1%	-1.2%	-11.7%	-11.2%
Class B	-3.1%	-2.6%	-4.5%	-1.9%	+0.3%	-0.3%	-15.3%	-16.8%
C-BQMall	-5.6%	-5.6%	-4.7%	-6.8%	-2.1%	-2.5%	-14.3%	-13.6%
C-BasketballDrill	-3.9%	-3.6%	-3.8%	-2.8%	-1.4%	-1.6%	-12.3%	-11.7%
C-PartyScene	-4.1%	-4.3%	-5.9%	-4.1%	-0.4%	-1.4%	-16.1%	-13.8%
C-RaceHorses	-3.1%	-2.1%	-3.4%	+1.2%	+0.2%	-0.4%	-11.9%	-10.4%
Class C	-4.2%	-3.9%	-4.5%	-3.1%	-0.9%	-1.5%	-13.7%	-12.4%
D-BQSquare	-8.7%	-9.6%	-10.1%	-11.6%	-4.0%	-4.5%	-16.6%	-19.5%
D-BasketballPass	-6.1%	-5.6%	-5.4%	-4.0%	-3.0%	-2.8%	-9.8%	-8.1%
D-BlowingBubbles	-3.7%	-3.8%	-4.8%	-3.8%	-0.5%	-1.3%	-16.1%	-14.5%
D-RaceHorses	-4.8%	-4.2%	-5.2%	-1.0%	-1.6%	-1.9%	-11.8%	-10.5%
Class D	-5.8%	-5.8%	-6.4%	-5.1%	-2.3%	-2.6%	-13.6%	-13.2%
Overall	-4.0%	-3.8%	-4.9%	-3.4%	-0.6%	-1.2%	-13.5%	-14.2%
	BD-rate=-3.9%		BD-rate=-4.2%		BD-rate=-0.9%		BD-rate=-13.9%	

evident for test sequences, such as ParkRunning3, BQTerrace and BQSquare, which present a large number of sharp edges. We have also compared the proposed method with two JVET proposals, JVET-N0254 [10] and JVET-O0079 [11], in **Table IV**, where the $\ell1$ trained CNNs provides superior enhancement performance to these two works for all resolution classes according to PSNR.

B. Subjective comparison

Figure 3 and **4** provide subjective comparisons between the reconstructed frames generated by the original VVC/AV1, $\ell1$ trained CNNs and perceptual loss function trained models. We can observe that the reconstructed blocks of the proposed method (for both $\ell1$ and perceptually trained models) exhibit fewer noticeable blocking artefacts compared to the anchor codecs. In addition, the CNN models trained using perceptual loss functions produce results with slightly more textural detail and higher contrast than those generated by $\ell1$ trained networks.

C. Complexity analysis

The relative computational complexity of the proposed method, benchmarked on the original VVC and AV1 codecs, is presented in **Table V**. We have used a shared cluster computer at the University of Bristol, to execute all the computations. This computer contains multiple nodes with 2.4GHz Inter CPUs, 138GB RAM and NVIDIA P100 GPU devices. We note that the average decoding complexity is 56.3 and 70.0 times compared to the original VVC VTM 4.0.1 and AV1 respectively, due to the employment of CNN-based post-processing at the decoder. This is for a configuration with 16 residual blocks in the generator. This figure is slightly higher than those of two JVET proposals O0079 [11] and N0254 [10], in which the decoder complexities (inc. the CNN-based PP module) are 45.6 and 35.2 times that of the original VVC decoder (based on the whole JVET dataset).

Moreover, we have further investigated the relationship between the number of residual blocks and compression performance. **Figure 5** shows the coding gains (in terms of PSNR) and algorithm relative complexity using CNN models with different numbers of residual blocks (N=4, 8, 12, 16, 20, 24, 28 and 32) to process VVC VTM QP 42 compressed content. Again the relative complexity here is benchmarked on the original VVC decoder. We observe that, when the number of residual blocks (N) increases from 4 to 16, the PSNR gain relative to the original VVC content increases in a linear

TABLE III: The compression performance of the proposed method benchmarked on the original AV1 1.0.0. Negative BD-rate values indicate coding gains.

Method	AV1-PP ($\ell 1$ loss)		AV1-PP (Perceptual loss)	
Metric	PSNR	VMAF	PSNR	VMAF
Class-Sequence	BD-rate	BD-rate	BD-rate	BD-rate
A1-Campfire	-5.0%	-7.3%	-1.1%	-11.8%
A1-FoodMarket4	-4.0%	-8.9%	-1.1%	-9.6%
A1-Tango2	-4.6%	-8.2%	-1.9%	-10.7%
A2-CatRobot1	-5.9%	-5.9%	-1.9%	-15.0%
A2-DaylightRoad2	-7.7%	-5.0%	-3.1%	-17.2%
A2-ParkRunning3	-1.9%	-2.2%	+0.4%	-11.3%
Class A	-4.9%	-6.3%	-1.5%	-12.6%
B-BQTerrace	-4.5%	+1.4%	-1.3%	-10.2%
B-BasketballDrive	-6.0%	-3.0%	-2.9%	-7.1%
B-Cactus	-3.8%	-3.3%	-0.7%	-11.9%
B-MarketPlace	-3.0%	-4.5%	-0.5%	-17.4%
B-RitualDance	-4.7%	-5.0%	-2.3%	-11.1%
Class B	-4.4%	-2.9%	-1.5%	-11.5%
C-BQMall	-6.3%	-2.4%	-2.9%	-9.4%
C-BasketballDrill	-6.8%	-2.4%	-3.1%	-9.9%
C-PartyScene	-7.8%	+2.3%	-3.3%	-9.1%
C-RaceHorses	-4.1%	-3.8%	-1.8%	-8.4%
Class C	-6.3%	-1.6%	-2.8%	-9.2%
D-BQSquare	-16.1%	+11.2%	-8.4%	-4.5%
D-BasketballPass	-7.0%	-3.8%	-3.9%	-8.2%
D-BlowingBubbles	-6.2%	+2.0%	-2.8%	-9.7%
D-RaceHorses	-5.6%	-3.0%	-3.0%	-7.7%
Class D	-8.7%	+1.6%	-4.5%	-7.5%
Overall	-5.8%	-2.7%	-2.4%	-10.5%

TABLE IV: Comparison between the proposed method and two existing CNN-based PP approaches for VVC (low QP range).

Sequence (Class)	O0079 [11]	N0254 [10]	Proposed Method ($\ell 1$)
	BD-rate (PSNR)	BD-rate (PSNR)	BD-rate (PSNR)
Class A (2160p)	-1.3%	-1.7%	**-3.5%**
Class B (1080p)	-1.5%	-1.1%	**-3.1%**
Class C (480p)	-3.3%	-1.4%	**-4.2%**
Class D (240p)	-5.0%	-1.4%	**-5.8%**
Overall	-2.6%	-1.4%	**-4.0%**

TABLE V: Relative Complexity of the proposed method benchmarked on original VVC and AV1 decoders.

Hose Codec	VVC	AV1
Class A (2160p)	18.6×	23.2×
Class B (1080p)	36.8×	38.3×
Class C (480p)	76.5×	83.3×
Class D (240p)	116.9×	166.7×
Average	56.3×	70.0×

fashion. However when the number of residual blocks exceeds 20, the overall coding gain starts to decrease. This provides evidence that the residual block number in the proposed work is an optimal selection.

V. Conclusions

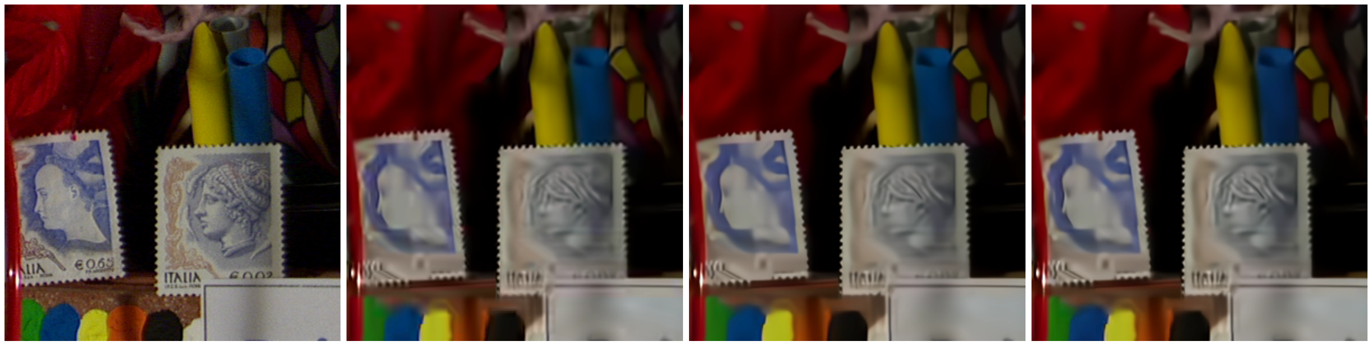
In this paper, we presented a CNN-based post-processing approach, which achieves evident and consistent coding gains over standardized video codecs, VVC and AV1. The employed CNN model was trained using both $\ell 1$ and perceptually inspired methodologies. We would like to recommend future work focusing on computational complexity reduction and further improvement on the training methodology.

VI. Acknowledgment

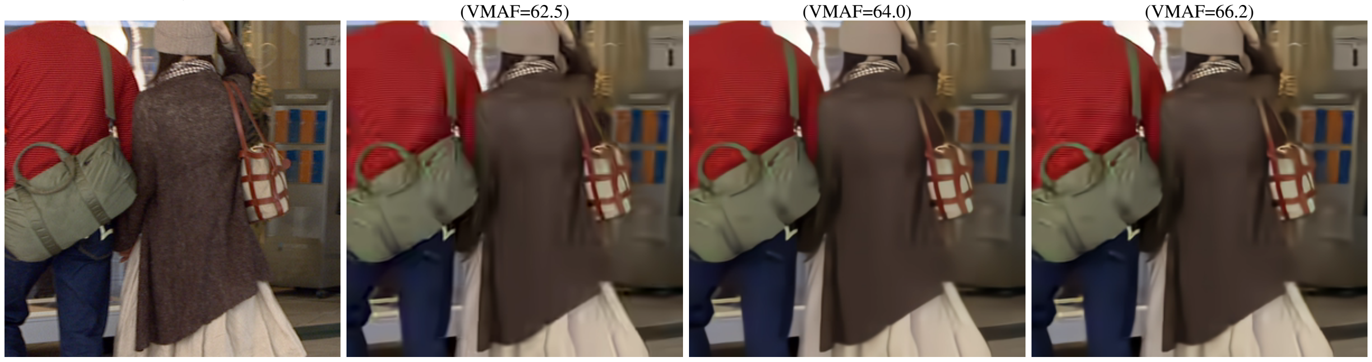
We would like to acknowledge funding from UK EPSRC (EP/L016656/1 and EP/M000885/1) and the NVIDIA GPU Seeding Grants. We also appreciate the valuable advice provided by Dr. Debargha Mukherjee on AV1 coding configuration.

References

[1] ITU-T Rec. H.266, *Versatile video coding*, ITU-T Std., 2020.



(a) Original (b) VTM 4.0.1, QP=42 (VMAF=62.5) (c) VTM-PP: $\ell 1$, QP=42 (VMAF=64.0) (d) VTM-PP: GAN, QP=42 (VMAF=66.2)



(e) Original (f) VTM 4.0.1, QP=42 (VMAF=70.4) (g) VTM-PP: $\ell 1$, QP=42 (VMAF=72.2) (h) VTM-PP: GAN, QP=42 (VMAF=74.3)

Fig. 3: Example blocks of the reconstructed frames for the anchor VTM 4.0.1 and the proposed approach. These are from the 91st and 320th frames of *Cactus* and *BQMall* sequences respectively. We can note that the results produced by the perceptual loss function trained models exhibit more textural detail and higher contrast than those generated by $\ell 1$ trained networks (the difference is more evident within the regions with the stamps in *Cactus*, and the areas with the bags and the white skirt in *BQMall*).

[2] Y. Chen, D. Mukherjee, J. Han, A. Grange, Y. Xu, S. Parker, C. Chen, H. Su, U. Joshi, C.-H. Chiang *et al.*, "An overview of coding tools in av1: the first video codec from the alliance for open media," *APSIPA Transactions on Signal and Information Processing*, vol. 9, 2020.

[3] F. Zhang, C. Feng, and D. R. Bull, "Enhancing VVC through CNN-based post-processing," in *Proc. IEEE Int Conf. on Multimedia & Expo*, 2020.

[4] D. Liu, Y. Li, J. Lin, H. Li, and F. Wu, "Deep learning-based video coding: a review and a case study," *ACM Computing Surveys (CSUR)*, vol. 53, no. 1, pp. 1–35, 2020.

[5] F. Zhang, M. Afonso, and D. R. Bull, "ViSTRA2: Video coding using spatial resolution and effective bit depth adaptation," *arXiv preprint arXiv:1911.02833 (submitted to Elsevier Signal Processing: Image Communication)*, 2019.

[6] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *International Conference on Learning Representations*, 2017.

[7] O. Rippel, S. Nair, C. Lew, S. Branson, A. G. Anderson, and L. Bourdev, "Learned video compression," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 3454–3463.

[8] L. Ma, Y. Tian, P. Xing, and T. Huang, "Residual-based post-processing for hevc," *IEEE MultiMedia*, vol. 26, no. 4, pp. 67–79, 2019.

[9] H. Zhao, M. He, G. Teng, X. Shang, G. Wang, and Y. Feng, "A CNN-based post-processing algorithm for video coding efficiency improvement," *IEEE Access*, vol. 8, pp. 920–929, 2019.

[10] Y. Wang, Z. Chen, Y. Li, L. Zhao, S. Liu, and X. Li, "Ce13: Dense residual convolutional neural network based in-loop filter (ce13-2.2 and ce13-2.3)," in *the JVET meeting*, no. JVET-N0254. ITU-T, ISO/IEC, 2019.

[11] S. Wan, M.-Z. Wang, H. Gong, C.-Y. Zou, Y.-Z. Ma, J.-Y. Huo, Y.-F. Yu, and Y. Liu, "CE10: Integrated in-loop filter based on CNN (Tests 2.1, 2.2 and 2.3)," in *the JVET meeting*, no. JVET-O0079. ITU-T, ISO/IEC, 2019.

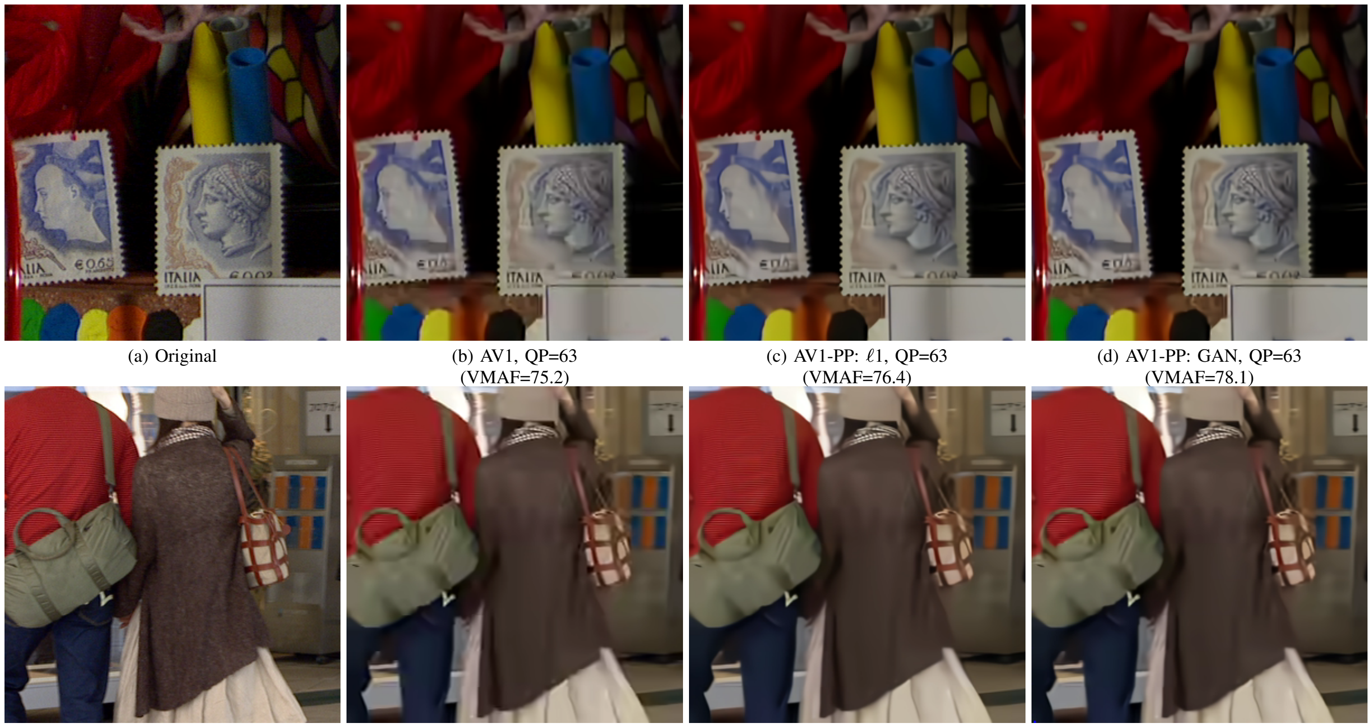
[12] C. Ledig and *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 105–114.

[13] D. Ma, F. Zhang, and D. R. Bull, "BVI-DVC: A training database for deep video compression," *arXiv preprint arXiv:2003.13552 (submitted to IEEE Trans. on Multimedia)*, 2020.

[14] F. Bossen, J. Boyce, X. Li, V. Seregin, and K. Sühring, "JVET common test conditions and software reference configurations for SDR video," in *the JVET meeting*, no. JVET-M1001. ITU-T and ISO/IEC, 2019.

[15] A. Jolicoeur-Martineau, "The relativistic discriminator: a key element missing from standard gan," *arXiv preprint arXiv:1807.00734*, 2018.

[16] D. Ma, F. Zhang, and D. R. Bull, "GAN-based effective bit depth adaptation for perceptual video compression,"



(a) Original (b) AV1, QP=63 (VMAF=75.2) (c) AV1-PP: $\ell1$, QP=63 (VMAF=76.4) (d) AV1-PP: GAN, QP=63 (VMAF=78.1)

(e) Original (f) AV1, QP=63 (VMAF=83.7) (g) AV1-PP: $\ell1$, QP=63 (VMAF=84.9) (h) AV1-PP: GAN, QP=63 (VMAF=86.2)

Fig. 4: Example blocks of the reconstructed frames for the anchor AV1 and the proposed approach. These are from the 91st and 320th frames of *Cactus* and *BQMall* sequences respectively. The results produced by the perceptual loss function trained models exhibit more textural detail and higher contrast than those generated by $\ell1$ trained networks (the difference is more evident within the regions with the stamps in *Cactus*, and the areas with the bags and the white skirt in *BQMall*).

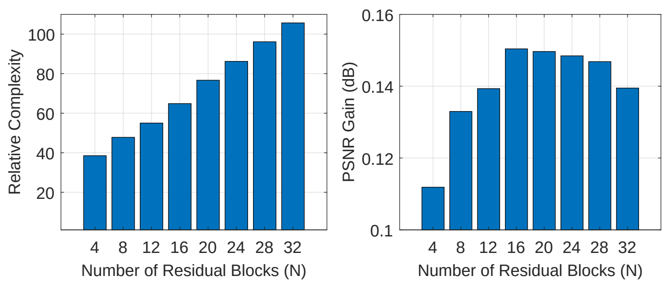


Fig. 5: (Left) Relative complexity for different number of residual blocks. (Right) PSNR gains for different number of residual blocks.

in *Proc. IEEE Int Conf. on Multimedia & Expo*, 2020.

[17] D. Ma, M. F. Afonso, F. Zhang, and D. R. Bull, "Perceptually-inspired super-resolution of compressed videos," in *Applications of Digital Image Processing XLII*, vol. 11137. International Society for Optics and Photonics, 2019, p. 1113717.

[18] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multi-scale structural similarity for image quality assessment," in *Proc. Asilomar Conference on Signals, Systems and Computers*, vol. 2. IEEE, 2003, p. 1398.

[19] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, pp. 600–612, 2004.

[20] Z. Li, A. Aaron, I. Katsavounidis, A. Moorthy, and M. Manohara, "Toward a practical perceptual video quality metric," *The Netflix Tech Blog*, 2016.

[21] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," in *13th VCEG Meeting*, no. VCEG-M33. Austin, Texas, USA: ITU-T, April 2001.