Deep Video Codec Control for Vision Models

Christoph Reich[1,2,3,4] Biplob Debnath[1] Deep Patel[1] Tim Prangemeier[3]
Daniel Cremers[2,4] Srimat Chakradhar[1]

[1]NEC Laboratories America, Inc. [2]TU Munich [3]TU Darmstadt
[4]Munich Center for Machine Learning (MCML)

Abstract

*Standardized lossy video coding is at the core of almost all real-world video processing pipelines. Rate control is used to enable standard codecs to adapt to different network bandwidth conditions or storage constraints. However, standard video codecs (*e.g.*, H.264) and their rate control modules aim to minimize video distortion w.r.t. human quality assessment. We demonstrate empirically that standard-coded videos vastly deteriorate the performance of deep vision models. To overcome the deterioration of vision performance, this paper presents the first end-to-end learnable* deep video codec control *that considers both* bandwidth constraints *and* downstream deep vision performance*, while* adhering to existing standardization. *We demonstrate that our approach better preserves downstream deep vision performance than traditional standard video coding.*

1. Introduction

Video data is a major source of internet traffic [2]. A significant and increasing amount of these videos is consumed and analyzed by deep vision models [22]. Streaming videos over a network or storing these requires lossy video codecs and rate control to meet dynamic bandwidth or storage constraints, preventing video corruption or dropping [24].

Almost all real-world video processing pipelines utilize standardized video coding (*e.g.*, H.264 [52]) to ensure interoperability and low-costs [29]. While deep video codecs have demonstrated promising results and can be optimized for deep vision models, they find minimal to no application in real-world [29, 53]. This is due to the lack of standardization (ISO) and limited support for rate control (*cf*. Tab. 1) [59].

Standard video codecs are developed to minimize image distortion w.r.t. human quality assessment (*cf*. Tab. 1) [40]. We demonstrate empirically that this is suboptimal for current deep vision models. More specifically, using H.264 coded videos during inference leads to a vast deterioration in downstream deep vision performance [37].

Table 1. **High-level comparison to existing approaches.** Our deep video codec control offers both rate control and standardization while being optimized for deep vision models

	Optimize vision perf.	Rate control	ISO
Deep video codecs	✓	∼	✗
Standard video codecs (*e.g.*, H.264)	✗	✓	✓
Deep video codec control	✓	✓	✓

In this paper, we aim to optimize standard video codecs for deep vision models (*e.g.*, a semantic segmentation network) by learning a deep video codec control (*cf*. Fig. 3). For a given video clip $\mathbf{V}$, we formulate the codec control task as a constrained optimization problem

$$\begin{aligned} &\max_{\text{QP}} \text{M}(\text{DNN}(\text{H.264}(\mathbf{V}, \text{C}_\theta(\mathbf{V}, b)))) \\ &\text{s.t. } \tilde{b} \leq b. \end{aligned} \tag{1}$$

Our lightweight control network C_θ consumes both the video clip $\mathbf{V}$ and the (dynamic) target bitrate b to predict high-dimensional codec parameters QP. We aim to learn a content and bandwidth-aware prediction of QP, controlling the H.264 coding, such that we maximized the performance of a deep vision model DNN, measured by a task-specific metric M (*e.g.*, accuracy). Additionally, the resulting video bitrate $\tilde{b}$ should not exceed the target bitrate b. To the best of our knowledge, we present the first end-to-end learnable codec control taking vision performance, bandwidth conditions, and existing standardizations into account (*cf*. Tab. 1). Making our deep video codec control the first standardized approach to support real-world bandwidth conditions as well as deep vision models.

Although approaches for optimizing standard video codecs for vision models within the scope of standardization have been proposed, they entail significant limitations impeding their real-world application [11, 16, 24]. First, existing approaches heavily rely on the use of saliency maps and hand-crafted rules [11, 16]. However, saliency maps

often fail to express highly relevant regions for deep vision models due to saturating gradients [45]. Second, existing approaches only consider the preservation of vision performance and do not consider rate control [11, 16, 24]. In contrast, our deep video codec control is not limited by the concept of saliency maps, does not rely on hand-crafted rules, and considers both the preservation of downstream vision performance as well as rate control, offering a greater potential for practical application.

Contributions. Motivated by the *potential performance gains* and the *need for standardization*, we make the following contributions: *(1)* We showcase that downstream deep vision performance vastly deteriorates when standard video coding is employed. *(2)* We propose the first end-to-end learnable codec control, directly optimizing Eq. (1), while maintaining compliance with standardization. *(3)* To enable end-to-end learning, we introduce a conditional differentiable surrogate model of a standard codec, allowing gradient propagation through the non-differentiable standard codec w.r.t. high-dimensional codec parameters. We showcase the effectiveness of our deep codec control by controlling H.264 for two classical vision tasks (semantic segmentation and optical flow estimation) and on two datasets (Cityscapes [8] and CamVid [4]).

2. Related Work

Video compression. The inherent memory complexity and temporal redundancy of video data have motivated the development of numerous algorithms for compressing videos [32, 41, 54]. Significant time has been dedicated to the development of standardized video compression algorithms, examples include H.264 [52], H.265 [44], and MPEG-5 [7]. Recently, deep learning-based video compression algorithms have been introduced [53, 59]. Despite offering better preservation of the perceptual quality in the coded video and supporting custom quality objectives (*e.g.*, preserving downstream performance), deep video compression algorithms find minimal applicability, as they lack widely supported standardization, are computationally heavy, and offer limited customizability as well as control over the compression process for rate control [59].

Video compression for machines. With an increasing amount of video data being analyzed by machines, such as deep vision models, a new line of research on video compression for machines has emerged [12]. A common approach is to extract vision feature maps from an image or video and perform compression at the feature level [12, 13, 35, 56]. Another approach is to train a deep compression algorithm for a specific vision task, such as object detection [3]. While some efforts have been made towards standardizing video compression for machines, no general standard has emerged yet [12, 17, 53], and the applicability of such approaches remains very limited. We do not aim to develop a new video codec for machines, rather our aim is to control widely used standard codecs for deep vision models as content and network bandwidth changes.

Video codec control for vision models. A common approach for identifying regions of interest for vision tasks is to utilize local and simple heuristics [31, 57]. Hard-coded rules are used to decide how to set the codec control [31, 57]. These approaches implicitly assume static scenes and trivially fail on dynamic scenes. Other more sophisticated approaches utilize feedback from a server-side vision model during inference for controlling the encoding [10, 33, 58]. Feedback loop-based approaches entail complicated architectures, add additional points of failure, can break standardization, and often only support a specific vision task. Mandhane *et al*. [34] use the MuZero reinforcement learning algorithm to learn a codec control. This control predicts a single codec parameter for each frame under a bandwidth constraint and aims to optimize perceptual quality but does not target vision models. Itsumi *et al*. [24] propose an RL approach for finding regions relevant to a vision model. Paired with hard-coded rules they presented a codec control for H.265. While this approach considers relevant regions for a deep vision model, Itsumi *et al*. do not consider a dynamic bandwidth constraint. AccMPEG [11] and Galteri *et al*. [16] also estimate regions relevant for a deep vision model by learning to predict saliency maps [43]. These approaches also do not consider a bandwidth condition, use hard-coded rules for the control, and are limited by the concept of saliency maps [45].

Video codec surrogate. Standard video codecs, such as H.264 and H.265, are not differentiable w.r.t. the input video and codec parameters. This non-differentiability has motivated the use of deep neural networks to approximate standard codecs in a differentiable manner [23, 27, 39, 47, 61]. For instance, Tian *et al*. [47] proposed a differentiable surrogate of H.265 that predicts the coded video, for a small subset of clip-wise compression strengths. While we are not the first to learn a differentiable approximation of a standard video codec, we present the first conditional surrogate with support for fine-grain macroblock-wise quantization and offer a differentiable file size prediction.

3. H.264 and Deep Vision Performance

H.264/AVC performs efficient lossy video compression by exploiting image compression techniques and temporal redundancies [52]. The predictive coding architecture of the H.264 codec utilizes sophisticated hand-crafted transforma-

tions to analyze redundancy within videos. A macroblock-wise motion-compensated discrete cosine transform (DCT) followed by quantization and lossless coding is used to archive effective video compression.

Quantization parameter. The H.264 codec allows for a variety of different customizations to the encoding process [41]. The quantization parameter (QP) controls the quantization strength and is the key parameter to control the compression strength. QP ranges from 0 to 51 (integer range), with high values leading to stronger compression. Strong compression reduces bitrate at the cost of video distortion, measured by SSIM (structural similarity index measure) [51]. Note that, for a given set of codec parameters, the bitrate remains non-trivially dependent on the video content. For example, a video with entirely black frames requires vastly fewer bits than a natural video, while being encoded with the same codec parameters.

Macroblock-wise quantization. H.264 offers support for macroblock-wise quantization, in which regions of the video, in this case, 16×16 frame patches (macroblocks), are compressed with varying QP values. Thus, irrelevant regions can be compressed with a high QP value (strong compression) and relevant regions with a lower QP value (less compression). An example of macroblock-wise coding is given in Fig. 1. We employ macroblock-wise quantization to facilitate a fine-grain spatial and temporal control of the video distortion and bitrate [28].

Group of pictures. The group of pictures (GOP) size further influences the encoding, by controlling which frames are encoded as an I, B, or P-frame. I-frames (intra-coded frames) are only compressed by utilizing spatial redundancies (similar to image coding), whereas B-frames (bidirectional predicted frames) and P-frames (predicted frames) also use information from adjacent frames. In particular, B-frames are compressed by utilizing a previous and a subsequent I- or P-frame. For compressing P-frames only a single previous I- or P-frame is used.

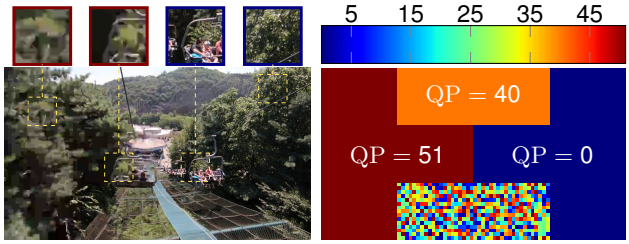

Figure 1. **H.264 macroblock-wise encoding example.** Effect of different macroblock-wise quantization parameters on the visual quality. Video frame (left) is encoded with QP map (right) as an I-frame. Video data from REDS dataset [36].

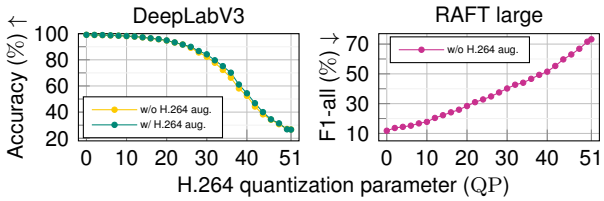

Figure 2. **Vision performance *vs*. compression.** Cityscapes segmentation accuracy and optical flow estimation performance, measured by the average endpoint error (AEPE), for different H.264 quantization parameters between the raw clip predictions (pseudo label) and the coded clip predictions. QP is applied uniformly.

Deep vision performance on H.264 coded videos. We empirically showcase that lossy compressing videos with H.264 can severely deteriorate downstream performance. When uniformly (for all macroblocks) increasing QP, both segmentation accuracy (DeepLabV3 [6]) and optical flow estimation (RAFT [46]) performance decreased, with reference to the prediction obtained on the original uncompressed video. Otani *et al*. [37] observed the same for action recognition. Using H.264 augmented frames during training cannot help to overcome this issue. We trained a DeepLabV3 model (w/ ResNet-18 [21] backbone) on H.264 coded frames (H.264 aug.). While consuming coded frames during training performance is still highly affected by the compression. Fig. 2 visualizes the results of these experiments. We aim to learn the allocation of QP such that downstream performance is better preserved and a target bandwidth is met. Additional experiments and experimental details are in the supplement.

4. Method

Here we introduce our deep video codec control (*cf*. Fig. 3). Before we describe our novel deep video codec control we present our H.264 codec surrogate model. Surrogate modeling the H.264 in a differentiable manner is required to enable our end-to-end codec control training.

4.1. Differentiable H.264 surrogate model

The H.264 video codec is not differentiable due to discreet operations (non-differentiable) and quantizations (gradient zero or undefined). To enable a gradient flow from the vision model and the generated bandwidth to the codec control network we aim to build a conditional differentiable surrogate model of H.264.

We consider H.264 coding (encoding & decoding) as a continuous[1] black-box function mapping the original (raw) video $\mathbf{V}$ conditioned on the macroblock-wise quantization

[1]Standard H.264 maps from discreet input to discreet outputs. To enable continuous differentiation we extend the H.264 surrogate mapping to real-valued input and output videos.

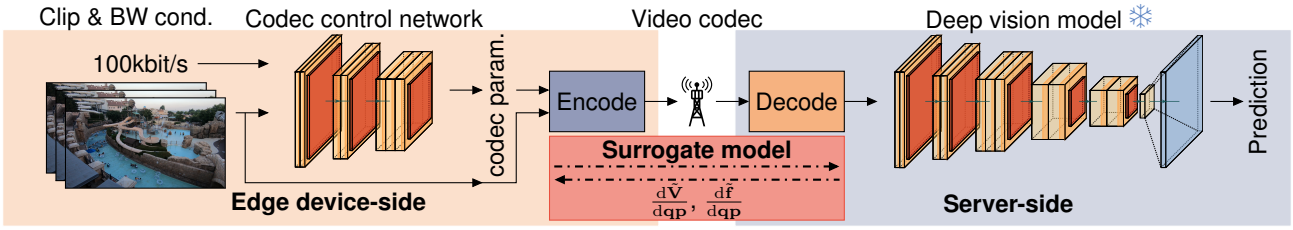

Figure 3. **Deep video codec control pipeline.** The control network predicts high-dimensional codec parameters for an input clip and a given dynamic network bandwidth (BW) condition. The video clip is encoded using the predicted codec parameters, sent over the network to the server-side, decoded, and analyzed by a deep vision model (*e.g.*, segmentation model). At training, the pre-trained server-side model is fixed and a differentiable surrogate model of the standard codec is used to propagate gradients from the server-side model and the file size prediction to the control network. During inference, the surrogate model is not used. Video frames from [36].

parameters $\mathbf{QP}$ to the encoded and decoded video $\hat{\mathbf{V}}$ as well as the encoded per-frame file size $\mathbf{f}$

$$\begin{aligned} \text{H.264}(\mathbf{V}, \mathbf{QP}) = (\hat{\mathbf{V}}, \mathbf{f}),\ \mathbf{V}, \hat{\mathbf{V}} \in \mathbb{R}^{3\times \mathrm{T}\times \mathrm{H}\times \mathrm{W}}, \\ \mathbf{QP} \in \{0, \dots, 51\}^{\mathrm{T}\times \mathrm{H}/16\times \mathrm{W}/16},\ \mathbf{f} \in \mathbb{R}_{+}^{\mathrm{T}}. \end{aligned} \tag{2}$$

T indicates the number of frames and H $\times$ W the spatial dimensions of the RGB video. Other H.264 parameters are considered to be constant. In particular, we consider a GOP of 8 (thus, T = 8) and a default preset [48].

Intuitively, our surrogate model fulfills two tasks during the codec control training (*cf*. Fig. 3). First, it allows our codec control network to consume gradient-based feedback from the downstream model regarding its performance. Second, the codec control can also get gradient-based feedback w.r.t. the generated and required bandwidth/file size through our differentiable file size prediction.

Surrogate model architecture. Our H.264 surrogate model architecture (*cf*. Fig. 4) is motivated by computational and memory efficiency and encodes inductive biases from the original (non-diff) H.264 coding approach. In general, our surrogate model entails an encoder-decoder architecture with a bottleneck stage [42]. For computational efficiency, we constrain the encoder and decoder to the frame level. To efficiently learn temporal interactions, we present an aligned convolutional gated recurrent unit (AGRU) for each frame type (I-, P-, and B-frame) [1]. By using the optical flow prediction of a pre-trained (small) RAFT model [46], we align adjacent frames used to compress P- and B-frames in each AGRU iteration. Based on the output features of the AGRU we regress the file size on a per-frame level. Our file size head utilizes Multi-Head Attention [49] to perform cross-attention between learnable query tokens $\mathbf{t} \in \mathbb{R}^{3\times \mathrm{C_t}}$ and the AGRU output features to regress the per-frame file size. We utilize individual tokens for each frame type (I-, P-, and B-frame) and repeat the query tokens for each frame.

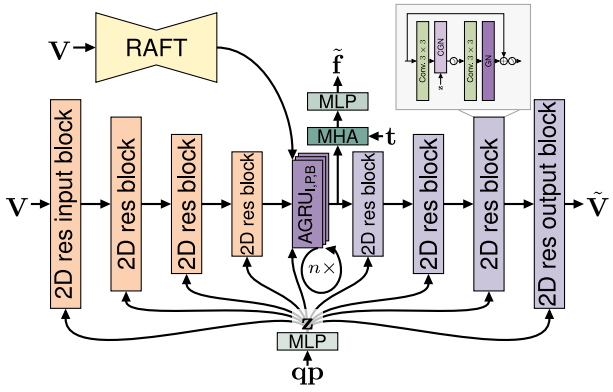

Figure 4. **Surrogate model architecture**. Our model is composed of a 2D encoder (orange), a 2D decoder (blue), an MHA-based file size head, and three AGRU bottleneck blocks. We use RAFT to compute the optical flows for the AGRU blocks. For embedding the $\mathbf{qp}$ we use an MLP. Skip-connections omitted for simplicity.

To condition both the encoder, decoder, and bottleneck blocks on QP we use a multilayer perceptron (MLP) to embed QP into a latent vector $\mathbf{z} \in \mathbb{R}^{\mathrm{C_z}\times \mathrm{T}\times \mathrm{H}/16\times \mathrm{W}/16}$. To incorporate $\mathbf{z}$ into the surrogate we used conditional group normalization (CGN) [9]. The CGN layer combines a spatial feature transform layer [50] followed by a group normalization layer [55] (w/o learnable parameters). Note our surrogate model uses one-hot encoded quantization parameters, denoted as $\mathbf{qp} \in \{0, 1\}^{52\times \mathrm{T}\times \mathrm{H}/16\times \mathrm{W}/16}$. This allows us to later formulate the prediction of the integer-valued QP as a classification problem. Our resulting surrogate is fully differentiable w.r.t. both the input video clip $\mathbf{V}$ and $\mathbf{qp}$.

Aligned convolutional gated recurrent unit. Taking inspiration from the motion-compensated and GOP-based compression performed by H.264 (and other standard codecs) we propose to utilize AGRUs in the bottleneck stage of our surrogate model. Our AGRU aims to approximate H.264 compression in an iterative fashion in latent space. Through an alignment in the latent space temporal interactions between the frame to be compressed and the reference frames are efficiently modeled. In particular, we

utilize separate AGRUs for each frame type.

The B-frame AGRU is described by

$$\begin{aligned}
\mathbf{Z}_{t+1} &= \sigma(\mathrm{CGN}(\mathrm{C}_{3\times3}(\mathbf{H}_t) + \mathrm{C}_{3\times3}([\hat{\mathbf{A}}_t, \tilde{\mathbf{A}}_t]), \mathbf{z})) \\
\mathbf{R}_{t+1} &= \sigma(\mathrm{GN}(\mathrm{C}_{1\times1}(\mathbf{H}_t) + \mathrm{C}_{1\times1}([\hat{\mathbf{A}}_t, \tilde{\mathbf{A}}_t]))) \\
\overline{\mathbf{H}}_{t+1} &= \tanh(\mathrm{GN}(\mathrm{C}_{1\times1}(\mathbf{R}_{t+1} \odot \mathbf{H}_t) + \mathrm{C}_{1\times1}([\hat{\mathbf{A}}_t, \tilde{\mathbf{A}}_t]))) \\
\mathbf{H}_{t+1} &= (1 - \mathbf{Z}_{t+1}) \odot \mathbf{H}_t + \mathbf{Z}_{t+1} \odot \overline{\mathbf{H}}_{t+1},
\end{aligned} \tag{3}$$

where $\mathrm{C}_{3\times3}$ and $\mathrm{C}_{1\times1}$ denote a 3×3 and a 1×1 2D convolution, respectively. We do not share parameters between convolutions. $\mathbf{H}_t$ are the latent features of the B-frame. $\hat{\mathbf{A}}_t$ and $\tilde{\mathbf{A}}_t$ are the aligned previous and subsequent frame features used for compression based on the GOP structure. We align the features of the frames used for compression by $\hat{\mathbf{A}}_t = \overleftarrow{w}(\hat{\mathbf{H}}_t, \mathrm{RAFT}(\mathbf{V}_t, \hat{\mathbf{V}}_t))$. We backward warp ($\overleftarrow{w}$) the unaligned features $\hat{\mathbf{H}}_t$ based on the optical flow between the frame to be compressed $\mathbf{V}_i$ and the reference frame $\hat{\mathbf{V}}_j$, using RAFT. We downsample the optical flow to match the spatial dimension of the latent features. For P-frames, we utilize one reference frame. In the case of an I-frame we fully omit the conditioning in the AGRU. Note the reference frames for B- and P-frame compression can be obtained by the known GOP structure [52].

Surrogate model training. In general, we are interested that our surrogate model approximates both the H.264 function ($\hat{\mathbf{V}} \approx \tilde{\mathbf{V}}$, $\mathbf{f} \approx \tilde{\mathbf{f}}$, *cf*. Eq. (2)) and provide smooth gradients w.r.t. the quantization parameters ($\frac{\mathrm{d}\tilde{\mathbf{V}}}{\mathrm{d}\mathbf{qp}}$, $\frac{\mathrm{d}\tilde{\mathbf{f}}}{\mathrm{d}\mathbf{qp}}$). Based on the control variates theory, the surrogate model can become a low-variance and continuous gradient estimator of Eq. (2) if *(1)* the difference between the output of the surrogate and the true H.264 function is minimized and *(2)* the two output distributions are maximizing the correlation coefficients ρ [19, 20]. We enforce both requirements for $\tilde{\mathbf{V}}$ and $\tilde{f}$ by minimizing $\mathcal{L}_\mathrm{s} = \mathcal{L}_{\mathrm{s}_v} + \mathcal{L}_{\mathrm{s}_f}$ during training. $\mathcal{L}_{\mathrm{s}_v}$ supervises the per-frame file size prediction and is defined as

$$\mathcal{L}_{\mathrm{s}_f} = \alpha_{\rho_f}\, \mathcal{L}_{\rho_f} + \alpha_{\mathrm{L1}}\, \mathcal{L}_{\mathrm{L1}}. \tag{4}$$

$\mathcal{L}_{\rho_f}$ is the correlation coefficient loss [47] between the true file size $\mathbf{f}$ and the predicted file size $\tilde{\mathbf{f}}$. $\mathcal{L}_{\mathrm{L1}}$ is used to minimize the distance between $\mathbf{f}$ and $\tilde{\mathbf{f}}$. Note that we learn $\tilde{\mathbf{f}}$ in $\log_{10}$-space due to the large range of file sizes.

To learn the prediction of the coded video $\tilde{\mathbf{V}}$ we minimize

$$\mathcal{L}_{\mathrm{s}_v} = \alpha_{\rho_v}\, \mathcal{L}_{\rho_v} + \alpha_{\mathrm{SSIM}}\, \mathcal{L}_{\mathrm{SSIM}} + \alpha_{\mathrm{FF}}\, \mathcal{L}_{\mathrm{FF}}. \tag{5}$$

$\mathcal{L}_{\rho_f}$ is the correlation coefficient loss for the coded video prediction. We minimize the distance between $\hat{\mathbf{V}}$ and $\tilde{\mathbf{V}}$ both in pixel space and frequency space. In particular, we use the structural similarity measure (SSIM) loss [60] for minimizing the error in pixel space. Taking inspiration from the frequency-based compression utilized by H.264, we utilize the focal frequency loss [26], minimizing prediction error in the frequency space. α is used to denote the different weighting parameters.

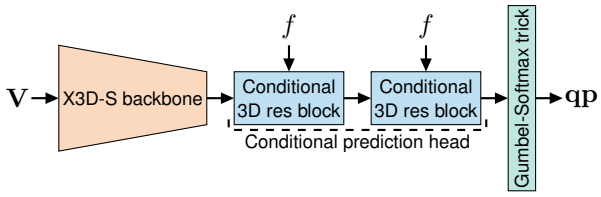

Figure 5. **Control network architecture.** We use a pre-trained X3D-S followed by two conditional 3D residual blocks. The $\mathbf{qp}$ one-hot vector is obtained by using the Gumbel-Softmax trick.

4.2. Deep video codec control

We aim to learn a deep codec control solving our constrained optimization (*cf*. Eq. (1)). In particular, given a video clip and a dynamic target bandwidth our deep codec control should predict high-dimensional codec parameters (QP), such that downstream performance is preserved while staying within the target bandwidth. Our general architecture is depicted in Fig. 3. Note, while we demonstrate our general codec control on the task of semantic segmentation and optical flow estimation in this paper, the general training formulation (*cf*. Eq. (6)) is agnostic to the vision task to be performed. Our approach only assumes the differentiability of the server-side vision model.

Control network. Our control network (*cf*. Fig. 5) consumes a original video clip $\mathbf{V}$ as well as a target bandwidth f and predicts the macroblock-wise QP. To facilitate the deployment on a standard edge device, such as the NVIDIA Jetson Nano, we use a lightweight architecture. In particular, we utilize X3D-S [15] as our control network. To input the bandwidth condition to the network we omit the X3D classification head and use two residual blocks with conditional normalization [9].

Due to the discreet nature of QP (integer-valued), we formulate the QP prediction as a classification. Our control network learns to predict a logit vector over all possible QP values. During end-to-end training, the Gumbel-Softmax trick [25] is used to produce a differentiable one-hot $\mathbf{qp}$ vector based on the predicted logits. During inference, when used as an input to the original H.264 codec and not to the surrogate, we apply the $\arg\max$ function.

Self-supervised control training. We propose a self-supervised training strategy to train our control network. To utilize end-to-end gradient-based learning we reformulate the constrained optimization problem (*cf*. Eq. (1)) as a continuous optimization task. In particular, our control loss $\mathcal{L}_\mathrm{c}$

consists of three terms - the bandwidth loss $\mathcal{L}_\mathrm{b}$, a performance loss $\mathcal{L}_\mathrm{p}$, and a bandwidth regularizer $\mathcal{L}_\mathrm{r}$,

$$\mathcal{L}_\mathrm{c} = \alpha_\mathrm{b}\,\mathcal{L}_\mathrm{b} + \alpha_\mathrm{p}\,\mathcal{L}_\mathrm{p} + \alpha_\mathrm{r}\,\mathcal{L}_\mathrm{r}. \tag{6}$$

where each individual loss is weighted by a separate regularization parameter α.

The bandwidth loss $\mathcal{L}_\mathrm{b}$ is used to enforce that the deep codec control satisfies the network bandwidth condition (*cf*. Eq. (1)) and is defined as

$$\mathcal{L}_\mathrm{b} = \max\big(0, \tilde{b} - b\,(1-\epsilon_\mathrm{p})\big), \tag{7}$$

where b is the maximum available bandwidth (bandwidth condition). $\tilde{b}$ denotes the bandwidth prediction computed based on the surrogate model file size prediction $\tilde{\mathbf{f}}$. We convert the per-frame file size (in B) to the bandwidth (in bit/s), with the known frame rate (fps), the number of video frames T, and the temporal stride Δt, by $\tilde{b} = \frac{8\,\sum_{i=1}^{\mathrm{T}} \tilde{f}_i\,\mathrm{fps}}{\mathrm{T}\,\Delta t}$, assuming a constant stream for the duration of the input clip. We use a small ϵ_B in order to enforce the generated bandwidth to be just below the available bandwidth.

The performance loss enforces the preservation of downstream vision performance. In the case of optical flow estimation, we use a scaled absolute error loss $\mathrm{H}(b - \tilde{b}\,(1+\epsilon_\mathrm{p}))\|\mathbf{O} - \tilde{\mathbf{O}}\|_1$. $\mathbf{O}$ denotes the optical flow prediction for the coded video clip and $\tilde{\mathbf{O}}$ represents the optical flow prediction based on the original (raw) video clip, used as a pseudo label. We scale the absolute error with a Heaviside function H, only considering the server-side model's performance if the target bandwidth is met with a small tolerance ϵ_p. For semantic segmentation, we replace the absolute error with the Kullback–Leibler divergence. Note, using a different server-side model (*e.g*. object detection model) can require adapting $\mathcal{L}_\mathrm{p}$ to the new task.

During preliminary experiments, we observed the control network can struggle to use the whole QP value range. Motivated by this observation, we regularize the control network toward generating a bitrate close to the target by

$$\mathcal{L}_\mathrm{r} = \big|\min\big(0, \tilde{b} - b\,(1-\epsilon_\mathrm{r})\big)\big|. \tag{8}$$

This regularization loss penalizes the control network if the bandwidth prediction $\tilde{b}$ is far away from the target bandwidth b. We utilize $\epsilon_\mathrm{B} < \epsilon_\mathrm{r}$ to not push the generated bandwidth above the target bandwidth.

Training schedule We train both our deep codec control and surrogate in an alternating fashion. To ensure a stable training of our control network from the beginning on, we pre-train the surrogate model before fine-tuning it during the control training. The codec control training is depicted in pseudocode in the supplement. Note, that our H.264 surrogate is only required for training, not for inference. For inference, the standard H.264 codec is used. We maintain an exponential moving average of the control network (w/ a decay of 0.99) to combat the noise in our learning signal introduced by our surrogate and the Gumbel-Softmax trick.

Discussion. We propose to learn high-dimensional codec parameters using end-to-end self-supervised learning, directly optimizing our control objective (Eq. (1)). However, learning codec parameters could theoretically also be achieved using reinforcement learning (RL) [34]. We argue that using RL is infeasible due to the high-dimensional action space (codec parameters) and the complex loss surface of downstream vision model [5, 30]. Note that our deep codec control training can be also viewed as a kind of knowledge distillation [17]. We learn our control network with knowledge distilled from the downstream model.

5. Experiments

Dataset. We perform experiments on Cityscapes [8] and CamVid [4]. We utilize the unlabeled sequences of Cityscapes including 30-frame video clips ($\sim$ 10k frames total) with 2967 training and 498 validation clips. The frame rate is 17fps. Similarly for Camvid, we use the available videos (at 15fps) composed of $\sim$ 29k frames in total. We use three videos for training and one video for validation. For codec control training & validation, we sample clips of 8 frames with a temporal stride of 3. For the surrogate model pre-training, we vary the temporal stride randomly between 1, 2, and 3.

Baseline. Since we are not aware of an existing general or task-specific approach to directly solve our constrained optimization problem (*cf*. Eq. (1)) within the scope of standardization, we utilize the generic H.264 rate control. In particular, we follow similar work and use 2-pass average bitrate (ABR) control [34, 48]. 2-pass ABR is a competitive baseline, since, in contrast to our control, 2-pass ABR consumes the video clip twice. The first pass gathers information about the motion and prediction error. The second pass uses this information to set QP for meeting a target bandwidth while minimizing distortion. Note that 2-pass ABR does not guarantee that the target bandwidth is met.

Control validation. We aim to validate the codec controls directly on the constrained optimization problem (*cf*. Eq. (1)) by measuring a task-specific metric, considering clip dropping. In a real-world deployment, exceeding the available network bandwidth can result in block noises, frame skipping, or even clip dropping (worst case) [24]. For validating segmentation performance we use the pixel-wise accuracy $\mathrm{acc}_\mathrm{seg}$ (in %) considering clip-dropping if bandwidth is exceeded. If a clip is dropped $\mathrm{acc}_\mathrm{seg}$ is considered

Figure 6. **Qualitative surrogate model results.** On the left frame coded by H.264 for the given QP map (right). In the middle is the coded video prediction of our surrogate model. We show the first frame of the clip. CamVid dataset used.

to be zero for the respective clip. For optical flow estimation, we use the F1-all metric, reporting the outlier ratio (in %) of pixels exceeding a threshold of 3 pixels or 5% w.r.t. the optical flow (pseudo) label. If bandwidth is exceeded we set F1-all to 100% for the respective clip. Note we employ the prediction on the original (uncomp.) clip as a pseudo label. In addition to the task-specific metric, we also measure the bandwidth condition accuracy $\mathrm{acc}_{\mathrm{bw}}$ (in %). Following similar work, we measure both the task-specific metric ($\mathrm{acc}_{\mathrm{seg}}$ or F1-all) and $\mathrm{acc}_{\mathrm{bw}}$ with a tolerance of zero ($\Delta 0\%$), five ($\Delta 2\%$), and ten ($\Delta 5\%$) percent on the bandwidth condition [34].

Implementation details. We implement our surrogate model and codec control pipeline using PyTorch [38] and Lightning [14]. For the macroblock-wise H.264 compression, we rely on a modified FFmpeg implementation of AcMPEG [11, 48]. We pre-train our surrogate model for 45k iterations. Our deep codec control is trained for just 6.0k iterations. For both, we utilize two NVIDIA A6000 (48GB) GPUs. Surrogate pre-training takes approximately one day, whereas the codec control training requires 16h to complete. We sample the network bandwidth condition b from a log uniform distribution $\mathcal{U}_{\log}(30\mathrm{kbit/s}, 0.9\mathrm{Mbit/s})$ capturing the working range of the H.264 codec. We sample from a log-uniform distribution to explore QP uniformly during training. For more implementation details and hyperparameters please refer to the supplement.

Surrogate model results. In Fig. 6, we showcase qualitative results of our codec surrogate model in approximating H.264 video distortion. Our conditional surrogate can adapt to different macroblock-wise quantization parameters and accurately approximate video distortion introduced by H.264. For low QP regions, our conditional surrogate maintains details of the video frame. For high QP regions, details and high-frequencies are discarded and the video is distorted similarly to the original coded video.

We also validate the effectiveness of our conditional H.264 surrogate model qualitatively (*cf*. Tab. 2). In addition, we also ablate the effect of our AGRU bottleneck block. For comparison, we train a conditional surrogate model with three standard 3D residual blocks, instead of our AGRU, in the bottleneck stage. Our conditional surrogate model is able to approximate the codec video of standard H.264 well. Additionally, our surrogate model also accurately predicts the generated file size. Using our AGRU improves over the 3D convolutional baseline. Note we offer additional results of our surrogate in the supplement.

Table 2. **Conditional surrogate model validation results.** We report SSIM and L1 scores for the distorted video prediction and the relative error for the file size prediction. Note a pixel range of $[0, 255]$ is used for validation. We show results averaged over the full QP range. We validate the surrogate model once on each possible QP value using a uniform QP (for all macroblocks).

		$\tilde{\mathbf{V}}$		$\tilde{\mathbf{f}}$
Method	AGRU	SSIM ↑	L1 ↓	Rel. error (%) ↓
		Cityscapes [8]		
Conditional surrogate	✗	0.949	3.892	6.719
Conditional surrogate	✓	0.964	2.552	5.150
		CamVid [4]		
Conditional surrogate	✓	0.958	2.876	2.246

Codec control results: semantic segmentation. We compare our deep codec control to 2-pass ABR on the downstream task of semantic segmentation. As the downstream model, we utilize a DeepLabV3 model (w/ ResNet-18 backbone) trained on the respective dataset. Our obtained results are depicted in Tab. 3. On both Cityscapes and CamVid, our deep codec control strongly outperforms the 2-pass ABR control. Our deep codec control better preserves downstream performance (with no bandwidth tolerance) and predicts codec parameters such that the dynamic bandwidth condition is met. In contrast, 2-pass ABR tends to overshoot the bandwidth condition. This can be observed by the vastly increasing $\mathrm{acc}_{\mathrm{bw}}$ if a bandwidth tolerance is

Table 3. **Semantic segmentation validation results.** BW ($\mathrm{acc}_{\mathrm{bw}}$) & segmentation accuracies ($\mathrm{acc}_{\mathrm{seg}}$) for difference BW tolerances reported. Metrics averaged over ten BW conditions.

	$\mathrm{acc}_{\mathrm{bw}}$ (%) ↑			$\mathrm{acc}_{\mathrm{seg}}$ (%) ↑		
Method	$\Delta 0\%$	$\Delta 2\%$	$\Delta 5\%$	$\Delta 0\%$	$\Delta 2\%$	$\Delta 5\%$
			Cityscapes [8]			
2-pass ABR (H.264)	68.13	74.98	82.31	64.29	70.57	77.07
Deep codec control	**96.22**	**97.05**	**97.91**	**84.79**	**85.50**	**86.28**
			CamVid [4]			
2-pass ABR (H.264)	63.91	74.43	85.36	54.06	62.49	**71.53**
Deep codec control	**94.64**	**95.61**	**96.46**	**65.70**	**62.52**	59.01

Table 4. **Optical flow validation results.** We report bandwidth accuracy (acc_{bw}) and F1-all scores for difference tolerances on the BW conditions. Metrics averaged over ten BW conditions.

Method	acc_{bw} (%) ↑			F1-all (%) ↓		
	Δ0%	Δ2%	Δ5%	Δ0%	Δ2%	Δ5%
Cityscapes [8]						
2-pass ABR (H.264)	69.60	76.48	83.09	41.99	36.37	31.18
Deep codec control	**98.05**	**98.29**	**98.86**	**27.57**	**27.42**	**27.03**
CamVid [4]						
2-pass ABR (H.264)	63.99	73.93	85.34	43.54	34.73	26.31
Deep codec control	**97.41**	**98.73**	**98.09**	**21.06**	**20.55**	**20.08**

permitted. This behavior is also reflected in an increased segmentation accuracy (considering drops) of 2-pass ABR if a bandwidth tolerance is permitted. However, in the most representative case, if no bandwidth tolerance is permitted, our deep codec control leads to vastly better downstream results. In particular, on Cityscapes our deep codec control leads to improvements in acc_{seg} of 20.5% and on CamVid to 11.6% over 2-pass ABR considering no BW tolerance.

Codec control results: optical flow estimation. We also analyze the performance of deep codec control and our 2-pass ABR baseline on optical flow estimation as the downstream vision task. As the downstream model, we utilize RAFT large, trained on synthetic data and driving data from KITTI [18, 46]. Tab. 4 presents our codec control results for optical flow estimation. Similar to semantic segmentation, our codec control also outperforms 2-pass ABR in controlling H.264 for optical flow estimation. Our deep codec control better preserves downstream performance (w/ no bandwidth tolerance) and better follows dynamic bandwidth conditions. Since the 2-pass ABR control often overshoots the bandwidth condition downstream performance is increased if a bandwidth tolerance is permitted. If no bandwidth tolerance is permitted, the most representative setting, our deep codec control leads to more than 10% fewer outliers (F1-all) in the optical flow prediction than 2-pass ABR.

Codec control transfer results In Tab. 5, we report results when transferring our deep codec control between downstream tasks during inference. When transferring our deep codec control trained to preserve optical flow performance to semantic segmentation we observe a drop in performance. This demonstrates both the ability of our codec control to learn a task-specific behavior and showcases the effectiveness of the surrogate model's gradients.

Additional results. We refer the reader to the supplement for additional (qualitative & quantitative) results of both our deep video codec control and conditional surrogate model.

Table 5. **Transfer results of our codec control from optical flow estimation to semantic segmentation on Cityscapes.** For reference, we also report results when directly trained on semantic segmentation.

Training task	acc_{bw} ↑			acc_{seg} ↑		
	Δ0%	Δ2%	Δ5%	Δ0%	Δ2%	Δ5%
Optical flow estimation	97.79	98.31	98.90	75.03	75.37	75.76
Semantic segmentation	96.22	97.05	97.91	84.79	85.50	86.28

6. Discussion

We demonstrated the general feasibility of our deep codec control in overcoming the limitations of the standard H.264 codec and conserving downstream performance. Our experiments demonstrate compelling results for two classical downstream tasks: semantic segmentation and optical flow estimation. However, currently, we train a deep codec control for a specific downstream task and model. Training a deep control for every task and model might not be practical. An avenue for future investigation lies in exploring how control networks can be transferred across different downstream tasks and models. Showcasing the potential application to control other video codecs, such as H.265, also represents an interesting direction for future research. Our deep codec control pipeline offers a solid foundation for future research toward cross-task support, a generalized deep codec control, and support for other standard video codecs. We hope that our contribution opens up new avenues in video coding and enables the effective use of standard codecs in state-of-the-art computer vision pipelines.

7. Conclusion

We presented the first fully end-to-end learnable deep codec control for standard video codecs, to conserve downstream deep vision performance in the face of dynamic bandwidth conditions. Our novel conditional differentiable codec surrogate model enables us to learn a content and network bandwidth-dependent codec control using self-supervised learning. We empirically demonstrate that our deep codec control can control the standard H.264 codec and is able to meet dynamic bandwidth conditions, while better preserving the downstream performance of a deep vision model than a standard baseline (H.264 2-pass ABR control). Our deep video codec control not only offers an alternative approach for optimizing video codecs for deep vision models but we can also conserve current standards.

Acknowledgements. We thank Simone Schaub-Meyer for feedback on the initial version of the paper. We also thank Heinz Koeppl for the general support.

References

[1] Nicolas Ballas, Li Yao, Chris Pal, and Aaron C. Courville. Delving deeper into convolutional networks for learning video representations. In *ICLR*, 2016. 4

[2] Thomas Barnett, Shruti Jain, Usha Andra, and Taru Khurana. Cisco visual networking index (VNI) complete forecast update. *Americas/EMEAR CKN Presentation*, pages 1–30, 2018. 1

[3] Florian Beye, Hayato Itsumi, Charvi Vitthal, and Koichi Nihei. Recognition-aware deep video compression for remote surveillance. In *ICIP*, pages 1986–1990, 2022. 2

[4] Gabriel J. Brostow, Julien Fauqueur, and Roberto Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognit. Lett.*, 30(2):88–97, 2009. 2, 6, 7, 8

[5] Lucian Buşoniu, Tim de Bruin, Domagoj Tolić, Jens Kober, and Ivana Palunko. Reinforcement learning for control: Performance, stability, and deep approximators. *Annu. Rev. Control.*, 46:8–28, 2018. 6

[6] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv:1706.05587 [cs.CV]*, 2017. 3

[7] Kiho Choi, Jianle Chen, Dmytro Rusanovskyy, Kwang-Pyo Choi, and Euee S Jang. An overview of the MPEG-5 essential video coding standard. *IEEE Signal Process. Mag.*, 37 (3):160–167, 2020. 2

[8] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes dataset for semantic urban scene understanding. In *CVPR*, pages 3213–3223, 2016. 2, 6, 7, 8

[9] Harm de Vries, Florian Strub, Jeremie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C Courville. Modulating early visual processing by language. In *NIPS*, 2017. 4, 5

[10] Kuntai Du, Ahsan Pervaiz, Xin Yuan, Aakanksha Chowdhery, Qizheng Zhang, Henry Hoffmann, and Junchen Jiang. Server-driven video streaming for deep learning inference. In *SIGCOMM*, pages 557–570, 2020. 2

[11] Kuntai Du, Qizheng Zhang, Anton Arapin, Haodong Wang, Zhengxu Xia, and Junchen Jiang. AccMPEG: Optimizing video encoding for accurate video analytics. In *MLSys*, pages 450–466, 2022. 1, 2, 7

[12] Lingyu Duan, Jiaying Liu, Wenhan Yang, Tiejun Huang, and Wen Gao. Video coding for machines: A paradigm of collaborative compression and intelligent analytics. *IEEE Trans. Image Process.*, 29:8680–8695, 2020. 2

[13] John Emmons, Sadjad Fouladi, Ganesh Ananthanarayanan, Shivaram Venkataraman, Silvio Savarese, and Keith Winstein. Cracking open the DNN black-box: Video analytics with dnns across the camera-cloud boundary. In *MobiCom HotEdgeVideo Workshop*, pages 27–32, 2019. 2

[14] William Falcon and The PyTorch Lightning Team. PyTorch Lightning. https://github.com/Lightning-AI/lightning, 2019. 7

[15] Christoph Feichtenhofer. X3D: Expanding architectures for efficient video recognition. In *CVPR*, pages 203–213, 2020. 5

[16] Leonardo Galteri, Marco Bertini, Lorenzo Seidenari, and Alberto Del Bimbo. Video compression for object detection algorithms. In *ICPR*, pages 3007–3012, 2018. 1, 2

[17] Wen Gao, Shan Liu, Xiaozhong Xu, Manouchehr Rafie, Yuan Zhang, and Igor Curcio. Recent standard development activities on video coding for machines. *arXiv:2105.12653 [cs.CV]*, 2021. 2, 6

[18] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *Int. J. Robot. Res.*, 32(11):1231–1237, 2013. 8

[19] Peter W Glynn and Roberto Szechtman. Some new perspectives on the method of control variates. In *Monte Carlo and Quasi-Monte Carlo Methods 2000*, pages 27–49. Springer, 2002. 5

[20] Will Grathwohl, Dami Choi, Yuhuai Wu, Geoff Roeder, and David Duvenaud. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. In *ICLR*, 2018. 5

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 3

[22] Miao Hu, Zhenxiao Luo, Amirmohammad Pasdar, Young Choon Lee, Yipeng Zhou, and Di Wu. Edge-based video analytics: A survey. *arXiv:2303.14329 [cs.DC]*, 2023. 1

[23] Berivan Isik, Onur G Guleryuz, Danhang Tang, Jonathan Taylor, and Philip A Chou. Sandwiched video compression: Efficiently extending the reach of standard codecs with neural wrappers. *arXiv:2303.11473 [eess.IV]*, 2023. 2

[24] Hayato Itsumi, Florian Beye, Vitthal Charvi, and Koichi Nihei. Learning important regions via attention for video streaming on cloud robotics. In *IROS*, 2022. 1, 2, 6

[25] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-Softmax. In *ICLR*, 2017. 5

[26] Liming Jiang, Bo Dai, Wayne Wu, and Chen Change Loy. Focal frequency loss for image reconstruction and synthesis. In *ICCV*, pages 13919–13929, 2021. 5

[27] Jan P Klopp, Keng-Chi Liu, Liang-Gee Chen, and Shao-Yi Chien. How to exploit the transferability of learned image compression to conventional codecs. In *CVPR*, pages 16165–16174, 2021. 2

[28] Christoph H. Lampert. Machine learning for video compression: Macroblock mode decision. In *ICPR*, pages 936–940, 2006. 3

[29] Stefan Lederer. 2019 video developer report – The future of video: AV1 codec, AI & machine learning, and low latency. https://bitmovin.com/bitmovin-2019-video-developer-report-av1-codec-ai-machine-learning-low-latency/, 2019. 1

[30] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *NeurIPS*, 2018. 6

[31] Yuanqi Li, Arthi Padmanabhan, Pengzhan Zhao, Yufei Wang, Guoqing Harry Xu, and Ravi Netravali. Reducto: On-camera filtering for resource-efficient real-time video analytics. In *SIGCOMM*, pages 359–376, 2020. 2

[32] Dong Liu, Yue Li, Jianping Lin, Houqiang Li, and Feng Wu. Deep learning-based video coding: A review and a case study. *ACM Comput. Surv.*, 53(1):1–35, 2020. 2

[33] Luyang Liu, Hongyu Li, and Marco Gruteser. Edge assisted real-time object detection for mobile augmented reality. In *ACM MobiCom*, pages 1–16, 2019. 2

[34] Amol Mandhane, Anton Zhernov, Maribeth Rauh, Chenjie Gu, Miaosen Wang, Flora Xue, Wendy Shang, Derek Pang, Rene Claus, Ching-Han Chiang, et al. MuZero with self-competition for rate control in vp9 video compression. *arXiv:2202.06626 [eess.IV]*, 2022. 2, 6, 7

[35] Yoshitomo Matsubara, Sabur Baidya, Davide Callegaro, Marco Levorato, and Sameer Singh. Distilled split deep neural networks for edge-assisted real-time systems. In *MobiCom HotEdgeVideo Workshop*, pages 21–26, 2019. 2

[36] Seungjun Nah, Sungyong Baik, Seokil Hong, Gyeongsik Moon, Sanghyun Son, Radu Timofte, and Kyoung Mu Lee. NTIRE 2019 challenge on video deblurring and super-resolution: Dataset and study. In *CVPRW*, 2019. 3, 4

[37] Aoi Otani, Ryota Hashiguchi, Kazuki Omi, Norishige Fukushima, and Toru Tamaki. Performance evaluation of action recognition models on low quality videos. *IEEE Access*, 10:94898–94907, 2022. 1, 3

[38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. *NeurIPS*, 32, 2019. 7

[39] Kaitian Qiu, Lu Yu, and Daowen Li. Codec-simulation network for joint optimization of video coding with pre- and post-processing. *IEEE Open J. Circuits Syst.*, 2:648–659, 2021. 2

[40] Iain E Richardson. *H.264 and MPEG-4 video compression: Video coding for next-generation multimedia*. John Wiley & Sons, 2004. 1

[41] Iain E Richardson. *The H. 264 Advanced Video Compression Standard*. John Wiley & Sons, 2011. 2, 3

[42] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pages 234–241, 2015. 4

[43] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *ICLR*, 2014. 2

[44] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (HEVC) standard. *IEEE Trans. Circuits Syst. Video Technol.*, 22(12):1649–1668, 2012. 2

[45] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, pages 3319–3328, 2017. 2

[46] Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *ECCV*, pages 402–419, 2020. 3, 4, 8

[47] Yuan Tian, Guo Lu, Xiongkuo Min, Zhaohui Che, Guangtao Zhai, Guodong Guo, and Zhiyong Gao. Self-conditioned probabilistic learning of video rescaling. In *ICCV*, pages 4490–4499, 2021. 2, 5

[48] Suramya Tomar. Converting video formats with FFmpeg. *Linux J.*, 2006(146):10, 2006. 4, 6, 7

[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017. 4

[50] Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Recovering realistic texture in image super-resolution by deep spatial feature transform. In *CVPR*, pages 606–615, 2018. 4

[51] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4): 600–612, 2004. 3

[52] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra. Overview of the H.264/AVC video coding standard. *IEEE Trans. Circuits Syst. Video Technol.*, 13(7):560–576, 2003. 1, 2, 5

[53] Daniel Wood. Task oriented video coding: A survey. *arXiv:2208.07313 [eess.IV]*, 2022. 1, 2

[54] Dapeng Wu, Yiwei Thomas Hou, Wenwu Zhu, Ya-Qin Zhang, and Jon M Peha. Streaming video over the internet: Approaches and directions. *IEEE Trans. Circuits Syst. Video Technol.*, 11(3):282–300, 2001. 2

[55] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, pages 3–19, 2018. 4

[56] Sifeng Xia, Kunchangtai Liang, Wenhan Yang, Ling-Yu Duan, and Jiaying Liu. An emerging coding paradigm vcm: A scalable coding approach beyond feature and signal. In *ICME*, pages 1–6, 2020. 2

[57] Ben Zhang, Xin Jin, Sylvia Ratnasamy, John Wawrzynek, and Edward A. Lee. AWStream: Adaptive wide-area streaming analytics. In *SIGCOMM*, page 236–252, 2018. 2

[58] Wuyang Zhang, Zhezhi He, Luyang Liu, Zhenhua Jia, Yunxin Liu, Marco Gruteser, Dipankar Raychaudhuri, and Yanyong Zhang. ELF: Accelerate high-resolution mobile deep vision with content-aware parallel offloading. In *ACM MobiCom*, pages 201–214, 2021. 2

[59] Yun Zhang, Linwei Zhu, Gangyi Jiang, Sam Kwong, and C-C Jay Kuo. A survey on perceptually optimized video coding. *ACM Comput. Surv.*, 55(12):1–37, 2023. 1, 2

[60] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. Loss functions for image restoration with neural networks. *IEEE Trans. Comput. Imaging.*, 3(1):47–57, 2016. 5

[61] Lijun Zhao, Huihui Bai, Anhong Wang, and Yao Zhao. Learning a virtual codec based on deep convolutional neural network to compress image. *J. Vis. Commun. Image Represent.*, 63:102589, 2019. 2