FVC: A New Framework towards Deep Video Compression in Feature Space

Zhihao Hu
Beihang University
huzhihao@buaa.edu.cn

Guo Lu*
Beijing Institute of Technology
guo.lu@bit.edu.cn

Dong Xu
University of Sydney
dong.xu@sydney.edu.au

Abstract

Learning based video compression attracts increasing attention in the past few years. The previous hybrid coding approaches rely on pixel space operations to reduce spatial and temporal redundancy, which may suffer from inaccurate motion estimation or less effective motion compensation. In this work, we propose a feature-space video coding network (FVC) by performing all major operations (i.e., motion estimation, motion compression, motion compensation and residual compression) in the feature space. Specifically, in the proposed deformable compensation module, we first apply motion estimation in the feature space to produce motion information (i.e., the offset maps), which will be compressed by using the auto-encoder style network. Then we perform motion compensation by using deformable convolution and generate the predicted feature. After that, we compress the residual feature between the feature from the current frame and the predicted feature from our deformable compensation module. For better frame reconstruction, the reference features from multiple previous reconstructed frames are also fused by using the non-local attention mechanism in the multi-frame feature fusion module. Comprehensive experimental results demonstrate that the proposed framework achieves the state-of-the-art performance on four benchmark datasets including HEVC, UVG, VTL and MCL-JCV.

1. Introduction

There is an increasing research interest in developing the next generation video compression technologies. While the traditional video compression methods [37, 27] have achieved promising performance by using the hand-designed modules (*e.g.*, block-based motion estimation, Discrete Cosine Transform (DCT)) to reduce spatial and temporal redundancy, these modules cannot be optimized in an end-to-end fashion based on large-scale video data.

The recent deep video compression works [22, 38, 2, 10, 14, 21, 19] have achieved impressive results by applying the deep neural networks within the hybrid video compression framework. Currently, most works only rely on pixel-level operations (*e.g.,* motion estimation or motion compensation) for reducing redundancy. For example, pixel-level optical flow estimation and motion compensation are used in [22, 21, 14, 23] to reduce temporal redundancy, and pixel-level residual is further compressed by using the auto-encoder style network. However, this pixel-level paradigm suffers from the following three limitations. First, it is difficult to produce accurate pixel-level optical flow information, especially for videos with complex non-rigid motion patterns. Second, even we can extract sufficiently accurate motion information, the pixel-level motion compensation process may introduce additional artifacts. Third, it is also a non-trivial task to compress pixel-level residual information.

Given robust representation ability of deep features for various applications, it is desirable to perform motion compensation or residual compression in the feature space and more effectively reduce spatial or temporal redundancy in videos. Furthermore, the recent progress in deformable convolution [9, 35] has shown it is feasible to align two consecutive frames in the feature space. Based on the learned offset maps of convolution kernels (*i.e.*, the so-called dynamic kernels), deformable convolution can decide which positions are sampled from the input feature. By using deformable convolution based on dynamic kernels, we can better cope with more complex non-rigid motion patterns between two consecutive frames, which can thus improve the motion compensation results and also alleviate the burden for the subsequent residual compression module. Unfortunately, it is still a non-trivial task to seamlessly incorporate the feature space operations and deformable convolution into the existing hybrid deep video compression framework and train the whole network in an end-to-end fashion.

In our work, instead of following the existing works [22, 14, 3, 19] to adopt the pixel-level operations as in the traditional video codecs, we propose a new feature-space video coding network (referred to as FVC) by reducing the spatial-temporal redundancy in the feature space. Specifi-

* Guo Lu is the corresponding author.

cally, we first estimate motion information (*i.e.*, the offset maps for convolution kernels in deformable convolution), based on the extracted representations from two consecutive frames. Then the offset maps are compressed by using the auto-encoder style network and the reconstructed offset maps will be used in the subsequent deformable convolution operation to generate the predicted feature for more accurate motion compensation. After that, we adopt another auto-encoder style network to compress the residual feature between the original input feature and the predicted feature. Finally, a multi-frame feature fusion module is proposed to combine a set of reference features from multiple previous frames for better frame reconstruction.

When compared with the state-of-the-art learning based video compression methods [21, 14], we perform all operations in the feature space for more accurate motion estimation and motion compensation, in which we can seamlessly incorporate deformable convolution into the hybrid deep video compression framework. As a result, we can alleviate the errors introduced by inaccurate pixel-level operations like motion estimation/compensation and achieve better video compression performance. Our contributions are summarized as follows:

- We propose a new learning based video compression framework, which performs all operations including motion estimation, motion compensation and residual compression in the feature space.
- For effective motion compensation in the feature space, we use deformable convolution to "warp" the reference feature from the previous reconstructed frame and more accurately predict the feature of the current frame, in which the corresponding offset maps are compressed by using the auto-encoder style network.
- We propose a new multi-frame feature fusion module based on the non-local attention mechanism, which combines the features from multiple previous frames for better reconstruction of the current frame.
- Our framework FVC achieves the state-of-the-art performance on four benchmark datasets including HEVC, UVG, VTL and MCL-JCV, which demonstrates the effectiveness of our proposed framework.

2. Related Work

2.1. Image Compression

In the past decades, the traditional image compression methods like JPEG [33], JPEG2000 [28] and BPG [6] have been proposed to reduce spatial redundancy, in which the hand-crafted techniques, such as DCT, are exploited for achieving high compression performance. Recently, the learning based image compression methods have attracted increasing attention. The RNN-based image compression methods [31, 32, 15] are firstly introduced to progressively compress images. Other methods [4, 5, 25, 29] adopted the auto-encoder structure to first encode images as latent representations, and then decode the latent representations from the feature space to the pixel space, which have achieved the state-of-the-art performance for image compression.

2.2. Video Compression

A number of video compression standards [37, 27] have been proposed in the past few years. Most methods follow the hybrid coding structure, where the motion compensation and residual coding techniques are used to reduce the redundancy in both spatial and temporal domains. Recently, learning based video compression [38, 22, 26, 12, 10, 19, 21, 23, 14, 11, 2, 42, 8, 39, 20, 17, 41] has become a new research direction. Following the traditional hybrid video compression framework, Lu *et al*. [22] proposed the first end-to-end optimized video compression framework, in which all the key components in H.264/H.265 are replaced with deep neural networks. Lin *et al*. [19] used multiple frames in different modules to further remove the redundancy, while Hu *et al*. [14] proposed a resolution-adaptive optical flow compression method by automatically deciding the optimal resolution for each frame and each block.

Most of the existing approaches [22, 19, 14] have to estimate the pixel-level optical flow maps and compress the corresponding pixel-level residual information. However, it may be difficult to produce reliable pixel-level flow maps or residual information, which often degrades the compression performance of learning based video codecs. To address this problem, some recent works [10, 11] proposed to calculate the feature-level residual maps for video compression. In contrast to these works [10, 11], in our work, we propose to perform all operations in the feature space, which leads to better video compression performance.

2.3. Deformable Convolution

Recently, Dai *et al*. [9] proposed to use deformable convolution together with the learnt offset maps to enhance the modeling capability of convolution neural networks, which has achieved promising results in several video-related tasks like action recognition [18] and video super-resolution [30, 35]. In contrast to these works, our work is the first to investigate how to employ deformable convolution in learning based video compression. Considering that there are multiple complex components in our hybrid video compression system, it is a non-trivial task to develop an end-to-end optimized video codec by seamlessly incorporating deformable convolution and simultaneously compressing the corresponding offset maps and other features.

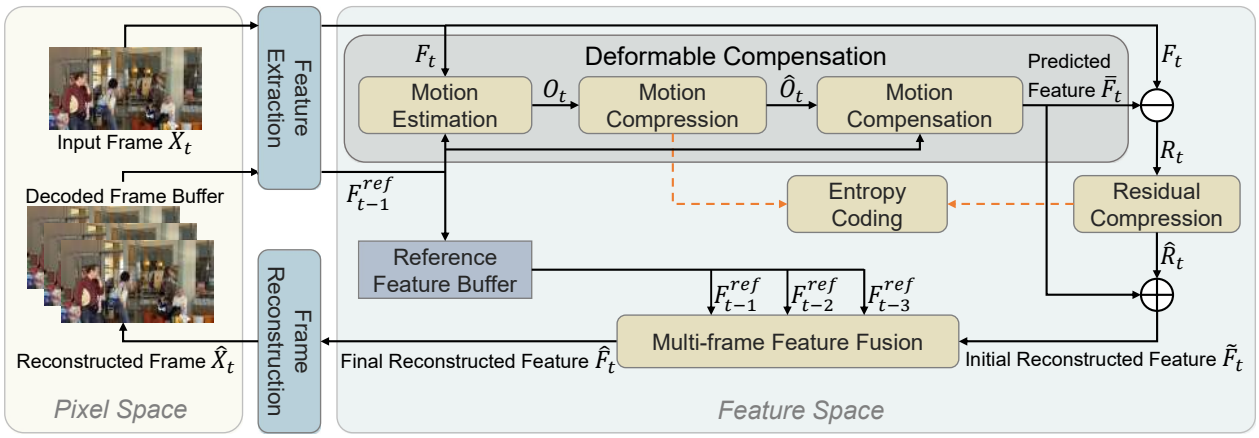


Figure 1. Overview of our proposed video compression framework. Given an input frame X_t, we first encode the input frame to produce the input feature F_t. The **Deformable Compensation** module consists of three steps (*i.e.*, motion estimation, motion compression and motion compensation). Specifically, the offset map O_t between the features F_t and F_{t-1}^{ref} from the previous reconstructed frame is estimated and then used as motion information. The offset map is then compressed and the reconstructed offset map $\hat{O}_t$ is employed for motion compensation by using the deformable convolution operation. The residual feature R_t between the input feature F_t and the predicted feature $\bar{F}_t$ is compressed in the **Residual Compression** module. Additionally, the **Multi-frame Feature Fusion** module is proposed to fuse the initial reconstructed feature $\tilde{F}_t$ and the features $F_{t-1}^{ref}, F_{t-2}^{ref}$ and F_{t-3}^{ref} from multiple previous frames to produce the final reconstructed feature $\hat{F}_t$. Finally, the reconstructed frame $\hat{X}_t$ is generated after going through the frame reconstruction module.

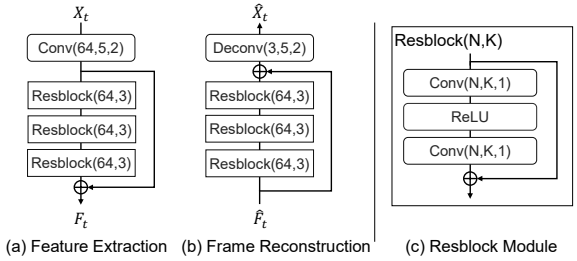


Figure 2. The network structure of (a) our Feature Extraction module and (b) our Frame Reconstruction module with the details of each Resblock are shown in (c). "Conv(N,K,S)" and "Deconv(N,K,S)" represent the convolution and deconvolution operations with the output channel, the kernel size and the stride as N, $K \times K$ and S, respectively.

3. Methodology

3.1. Overview

Let $X = \{X_1, X_2, ..., X_{t-1}, X_t, ...\}$ denote a video sequence, where X_t is the original video frame at the current time step. In our video compression system, the objective is to produce the high quality reconstructed frame $\hat{X}_t$ at any given bit-rate. As shown in Fig. 1, all the modules in our proposed framework including deformable compensation, residual compression and multi-frame feature fusion are performed in the feature space. The overview architecture of our proposed framework is summarized as follows,

Feature Extraction. To produce the representations in the feature space, the original input frame X_t and the previous reconstructed frame $\hat{X}_{t-1}$ are encoded as the feature representations F_t and F_{t-1}^{ref}, respectively. As shown in Fig. 2(a), the feature extraction module uses a convolution layer with the stride of 2, which is then followed by several residual blocks to produce the feature representation.

Deformable Compensation. This procedure consists of three steps: motion estimation, motion compression and motion compensation. Specifically, based on F_t and F_{t-1}^{ref}, we perform motion estimation by using a lightweight network, and the output offset map O_t will be compressed by using the newly proposed motion compression network before being transmitted to the decoder side. Finally, given the reconstructed offset map $\hat{O}_t$ and the feature F_{t-1}^{ref}, we can generate the predicted feature $\bar{F}_t$ by using deformable convolution in the motion compensation procedure. More details can be found in Section 3.2.

Residual Compression. The residual feature R_t between the input feature F_t and the predicted feature $\bar{F}_t$ will be compressed by using an auto-encoder style network for better reconstruction. After adding the reconstructed residual feature $\hat{R}_t$ back to the predicted feature $\bar{F}_t$, we produce the initial reconstructed feature representation $\tilde{F}_t$.

Multi-frame Feature Fusion. In this procedure, the extracted feature representations F_{t-1}^{ref}, F_{t-2}^{ref} and F_{t-3}^{ref} from three previous reconstructed frames as well as the initial reconstructed feature representation $\tilde{F}_t$ are fused to produce the final reconstructed feature $\hat{F}_t$. More details will be introduced in Section 3.3.

Frame Reconstruction. As shown in Fig. 2(b), the feature decoder with several residual blocks and a deconvolution layer will transform the final reconstructed feature $\hat{F}_t$ to the reconstructed frame $\hat{X}_t$.

Entropy Coding. The quantized features from the motion compression and residual compression modules will be transformed into the bit-streams by performing entropy coding. During the training process, we use a bit estimation

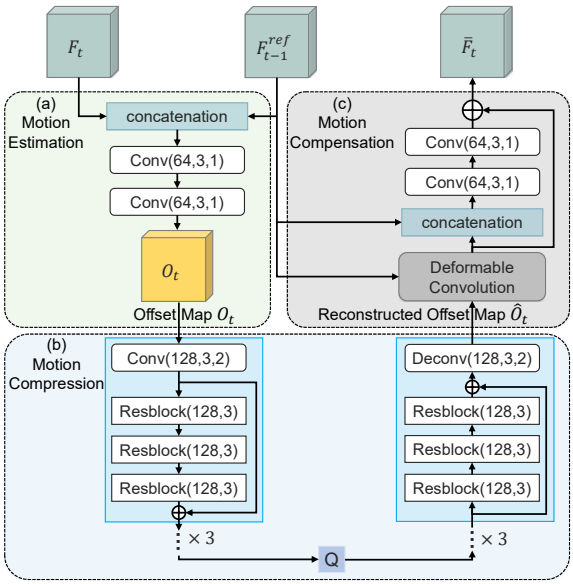


Figure 3. The network structure of our **Deformable Compensation** module, which consists of (a) the motion estimation submodule, (b) the motion compression submodule and (c) the motion compensation submodule for reducing redundancy.

network to estimate the number of bits. More details will be described in section 3.4.

3.2. Deformable Compensation

The previous video compression frameworks [22, 10, 14] rely on the optical flow estimation network and the pixel-level motion compensation network, which may lead to inaccurate frame prediction results and thus bring extra redundancy to the subsequent residual compression module. To this end, we perform motion compensation in the feature space and generate the predicted features by using the deformable convolution operation.

Given the features F_{t-1}^{ref} and F_t respectively from the previous reconstructed frame and the current frame, the deformable compensation procedure aims at generating the predicted feature $\bar{F}_t$ at the current time step. The whole network architecture is shown in Fig. 3. Specifically, we first produce the deformable offset map O_t by feeding F_{t-1}^{ref} and F_t into a 2-layer motion estimation network. Inspired by [11], we propose an auto-encoder style network to compress this offset map O_t, where both the encoder and the decoder consist of a set of Resblocks [13]. The offset map O_t is transformed to the latent space through the encoder and then quantized. After that, the decoder will convert the quantized latent representations back to the reconstructed offset map $\hat{O}_t$.

Finally, in the motion compensation procedure, the deformable convolution layer takes F_{t-1}^{ref} as the input and then performs the deformable convolution operation by using the corresponding filters with the aid of $\hat{O}_t$. In this way, we can better compress videos with complex non-rigid motion patterns by using the learned dynamic offset kernels and produce a more accurate warped feature. To generate a more accurate predicted feature, we further refine the output feature from the deformable convolution layer by using two convolution layers and eventually produce the final predicted feature $\bar{F}_t$.

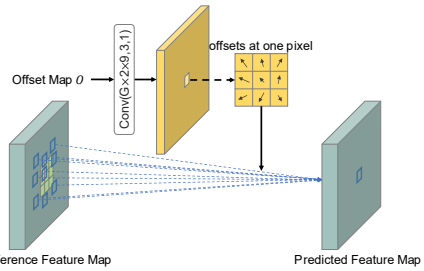


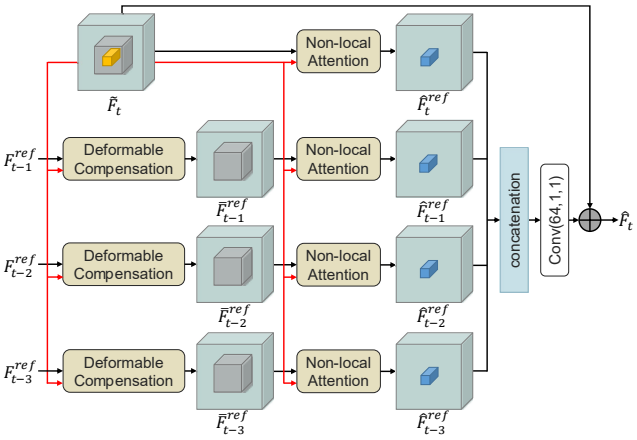
Figure 4. Illustration of Deformable Convolution. For the output channel of the convolution layer, "G", "2", "9" denote the channel group ($G = 8$), two directions (*i.e.*, the horizontal and vertical directions) of the offset map and the size of each kernel (3×3), respectively.

Deformable Convolution. The network structure of our deformable convolution layer [9] is shown in Fig. 4. Given the reference feature map and the corresponding offset map O, we aim to generate the predicted feature. For each kernel, the corresponding offsets are used to control the sampling location in the reference feature map, and then the feature values from the corresponding locations in the reference feature map are fused to generate one feature value in the predicted feature map. When compared with the pixel-level warping operation based on motion compensation as used in the previous video compression approach [22], our deformable compensation module provides more flexibility by using different sampling locations, which can better cope with complex non-rigid motion patterns and improve the compensation performance. In our implementation, we divide the channels of the reference feature into G groups (G=8 in this work) and use a shared offset map for each group of channels to better learn the offset maps and improve the deformable compensation results.

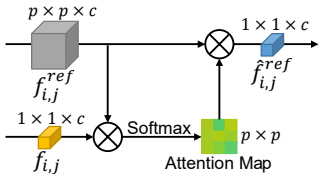
3.3. Multi-frame Feature Fusion

After the deformable compensation and residual compression procedures, we produce the initial reconstructed feature $\tilde{F}_t$ by combining the predicted feature $\bar{F}_t$ and the reconstructed residual feature $\hat{R}_t$. As shown in Fig. 1, to generate more accurate final reconstructed feature $\hat{F}_t$, a multi-frame feature fusion module is proposed to exploit temporal context information from the feature representations F_{t-1}^{ref}, F_{t-2}^{ref}, F_{t-3}^{ref} of multiple previous frames.

The detailed network architecture is shown in Fig. 5(a). Specifically, we first produce the predicted feature representations $\bar{F}_{t-1}^{ref}$, $\bar{F}_{t-2}^{ref}$ and $\bar{F}_{t-3}^{ref}$ by using the shared feature-space deformable compensation module. The non-local attention mechanism is then used to refine the predicted fea-



(a) The overall network structure for multi-frame feature fusion.



(b) Non-local attention mechanism.

Figure 5. Illustration of our multi-frame feature fusion module.

ture representation based on the similarity between the initial reconstructed feature $\tilde{F}_t$ and the predicted representations $\bar{F}_{t-1}^{ref}$, $\bar{F}_{t-2}^{ref}$ and $\bar{F}_{t-3}^{ref}$. It is worth mentioning that we also refine $\tilde{F}_t$ itself by using the non-local attention mechanism based on the so-called self-attention mechanism. Finally, we concatenate all these refined feature representations and use another convolution layer to generate the final reconstructed feature $\hat{F}_t$.

The network architecture of the deformable compensation module is the same as that in section 3.2. As for the non-local attention block, we provide the detailed architecture in Fig. 5(b). Here we take $\bar{F}_{t-1}^{ref}$ as an example to describe how to produce the refined feature $\hat{F}_{t-1}^{ref}$ after performing the attention operation guided by $\tilde{F}_t$, in which we assume the dimension of these two features $\bar{F}_{t-1}^{ref}$ and $\tilde{F}_t$ is $w \times h \times c$. For each spatial feature vector $f_{i,j}$ from $\tilde{F}_t$ at the location (i, j), whose size is $1 \times 1 \times c$, we find a collocated patch $f_{i,j}^{ref}$ with the size of $p \times p \times c$ from $\bar{F}_{t-1}^{ref}$. Then we calculate the similarity between $f_{i,j}$ and $f_{i,j}^{ref}$ by performing the convolution operation along the channel dimension and then produce the corresponding attention map after performing the softmax operation, which is further used to reweight all spatial positions in $f_{i,j}^{ref}$. After that, we generate the refined feature vector $\hat{f}_{i,j}^{ref}$ at the location (i, j). We repeat this procedure for each spatial location in $\tilde{F}_t$ and then produce the refined feature $\hat{F}_{t-1}^{ref}$. Finally, we concatenate all refined features $\hat{F}_{t-3}^{ref}$, $\hat{F}_{t-2}^{ref}$, $\hat{F}_{t-1}^{ref}$ and $\hat{F}_t^{ref}$ and perform the convolution operation to fuse these refined features and produce the final reconstructed feature $\hat{F}_t$ by adding the initial reconstructed feature $\tilde{F}_t$ back.

3.4. Residual Compression and Other Details

As shown in Fig. 1, in addition to offset map compression, we also need to compress the residual feature map. To simplify the whole system, we adopt the same network architecture as that used for the offset feature map compression (see Fig. 3(b)).

For the whole learning based video compression framework, we use the bit-rate estimation network to generate the bit-rate in the training stage. In our implementation, we employ the hyperprior entropy model in [25] for accurate bit-rate estimation. To reduce the computational complexity, the time-consuming auto-regressive model is not used in our approach.

To optimize the whole model in an end-to-end manner, it is also required to design a differentiable quantization operation. In our approach, we follow the method in [5] and approximate the quantization operation by adding the uniform noise in the training stage. In the inference stage, we directly use the *rounding* operation.

3.5. Loss Function

In our proposed framework, we optimize the following Rate Distortion (RD) trade-off:

$$RD = R + \lambda D = R_o + R_r + \lambda d(X_t, \hat{X}_t), \qquad (1)$$

where R_o and R_r denote the numbers of bits used to encode the offset map O_t and the residual feature map R_t, respectively. $d(X_t, \hat{X}_t)$ denotes the distortion between the input frame X_t and the reconstructed frame $\hat{X}_t$, where $d(\cdot)$ represents the mean square error or MS-SSIM [36]. λ is a hyper parameter used to control the rate-distortion trade-off.

4. Experiments

4.1. Experimental Setup

Training Dataset. We use the Vimeo-90K dataset [40] in the training stage, which contains 89,800 video clips with each video having 7 frames with the resolution of 448×256. We randomly crop the video sequences into the resolution of 256×256 before the training process.

Testing Datasets. To evaluate the performance of our FVC, we use the video sequences in four datasets including the HEVC [27], UVG [24], MCL-JCV [34] and VTL [1]. The HEVC datasets contain 16 videos in Class B, C, D and E with different resolutions from 416×240 to 1920×1080. The UVG dataset contains seven high frame rate videos with the resolution of 1920×1080, in which the difference between the neighboring frames is small. The MCL-JCV dataset is widely used for video quality evaluation, which consists of thirty 1080p video sequences. For the VTL

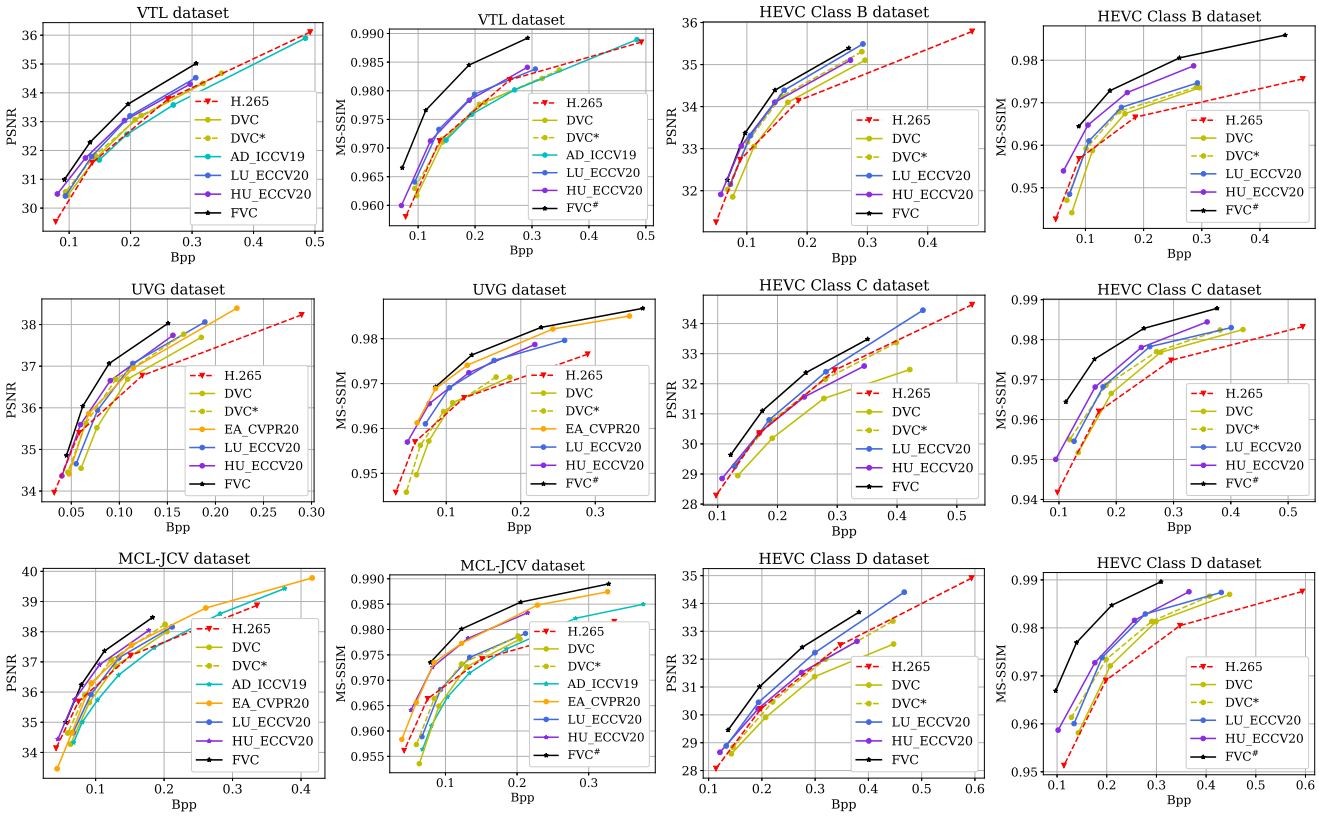


Figure 6. The experimental results on the VTL, UVG, MCL-JCV and HEVC Class B, Class C, Class D datasets. When using MS-SSIM for performance evaluation, we additionally perform the fine-tune operation in our FVC$^{\#}$ by using MS-SSIM as the distortion loss.

dataset, we use the first 300 frames from the high-resolution video sequences with the resolution of 352 × 288 for performance evaluation.

Evaluation Metrics. We use bpp (bit per pixel) to measure the average number of bits used for motion coding and residual coding for one pixel in each frame. We use PSNR and MS-SSIM[36] to evaluate the distortion between the reconstructed frame and the ground-truth frame and the PSNR/MS-SSIM of each video sequence is produced by calculating the average PSNR/MS-SSIM over all reconstructed frames.

Implementation Details. We train four models with different λ values (λ=256, 512, 1024, 2048). A two-stage training scheme is used to train our model. In the first training stage, we train the model without using the multi-frame feature fusion module for 2,000,000 steps at the learning rate of 5e-5. In the second stage, the multi-frame feature fusion module is included and we first train the overall framework with the learning rate of 5e-5 for 400,000 steps, and then optimize the model with the reduced learning rate of 5e-6 for 100,000 steps. When using MS-SSIM for performance evaluation, we further fine-tune the model from Stage 2 for 80,000 steps by using MS-SSIM as the distortion loss to achieve better MS-SSIM results. Our method is implemented based on PyTorch with CUDA support. All

Table 1. BDBR(%) results of DVC [22], EA [2], LU [21], HU [14] and our proposed method FVC when compared with H.265 [27] on different datasets. Negative values in BDBR indicate bit-rate savings while positive values indicate more bit-rate costs.

	DVC	EA	LU	HU	FVC
HEVC Class B	2.97	-	-15.92	-14.91	**-23.75**
HEVC Class C	20.65	-	-3.78	1.76	**-14.18**
HEVC Class D	14.08	-	-8.29	-1.77	**-18.39**
UVG	8.45	-9.75	-7.34	-13.27	**-28.71**
VTL	-10.92	-	-16.85	-20.17	**-28.10**
MCL-JCV	13.94	-1.52	4.75	-13.71	**-22.48**

the experiments are conducted on the machine with a single NVIDIA 2080TI GPU (11GB memory). We set the batch size as 4 and use the Adam optimizer [16]. It takes about 4 days, 3 days and 12 hours for the first training stage, the second training stage and the fine-tuning stage, respectively.

4.2. Experimental Results

Settings of the Baseline Methods. In this section, we provide the experimental results to demonstrate the effectiveness of our proposed method FVC on four benchmark datasets HEVC [27], UVG [24], VTL [1] and MCL-JCV [34]. We compare our method with other state-of-the-art methods including the traditional methods [37, 27] and recently proposed learning based methods (DVC [22], AD_ICCV19 [10], EA_CVPR20 [2], LU_ECCV20 [21] and

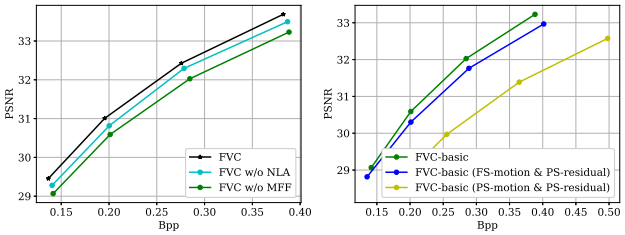
HU_ECCV20 [14]). For the traditional method H.265 [27], we follow the command line in [14] and use FFmpeg with the *default* mode. For fair comparison, we additionally report the results of "DVC*", which is an enhanced version of DVC by using the same motion and residual encoder/decoder modules as those in our deformable compensation module (see Fig. 3). Following the previous methods [22, 14, 21], we set the GoP size as 10 for the HEVC datasets and 12 for other datasets. We use the same I-frame compression method as in H.265 to reconstruct I-frame. For our "FVC$^{\#}$", we further perform the fine-tune operation according to the MS-SSIM based rate-distortion trade-off.

Results. In Table 1, we report the BDBR [7] results of different video compression methods when compared with H.265 on the HEVC, UVG, VTL and MCL-JCV datasets. Specifically, our approach saves more than 18% bit rate in terms of the overall results on all benchmark datasets. It is obvious that our approach outperforms DVC and other recent state-of-the-art learning based video compression approaches. For example, when compared with H.265, our proposed FVC saves 18.39% bit-rate on the HEVC Class D dataset while the corresponding bit-rate savings are 8.29% and 1.77% for the recent approaches LU_ECCV20 [21] and HU_ECCV20 [14].

We provide the RD curves of different compression methods in Fig. 6, it is noted that our method outperforms the baseline method DVC [22] and the enhanced version DVC* by a large margin on all datasets. We would like to highlight that DVC* directly compresses the pixel-level optical flow maps and the pixel-level residual maps by using the same compression network as our proposed method FVC, so the performance improvement clearly demonstrates the effectiveness of our proposed method. When compared with the traditional method H.265, our method achieves better results in terms of PSNRs at all bit-rates. Our method also outperforms all other baseline methods in terms of MS-SSIM.

4.3. Ablation Study and Analysis

Effectiveness of Different Components. As shown in Fig. 7, we take the HEVC Class D dataset as an example to demonstrate the effectiveness of different modules in our proposed framework FVC. As shown in Fig. 7(a), to demonstrate the effectiveness of the non-local attention (NLA) module, we remove NLA in the multi-frame feature fusion (MFF) module and the results of FVC w/o NLA drop nearly 0.2dB when compared with our complete model (*i.e.*, FVC). We also introduce the basic model of our approach (*i.e.*, FVC w/o MFF or FVC-basic), where the multi-frame feature fusion module is removed in our complete model. When compared with our complete model (*i.e.*, FVC), the result of our basic model FVC-basic after removing the MFF module drops 0.5dB at 0.3bpp (see the green and black curves). These experimental results demonstrate the effectiveness of our newly proposed MFF module equipped with the NLA mechanism for fusing the features from multiple previous frames.



(a) (b)

Figure 7. Ablation study on the HEVC Class D dataset. (1) **FVC**: Our proposed method FVC. (2) **FVC w/o NLA**: FVC without adopting the non-local attention (NLA) mechanism in the multi-frame feature fusion (MFF) module. (3) **FVC w/o MFF (also referred to as FVC-basic)**: FVC without using the whole MFF module. (4) **FVC-basic (FS-motion & PS-residual)** and (5) **FVC-basic (PS-motion & PS-residual)** are two variants of FVC-basic by only performing residual compression at the pixel space and by performing both motion-related operations and residual compression at the pixel space, respectively. Note the motion-related operations and residual compression in FVC-basic are both performed in the feature space, which are referred to as FS-motion and FS-residual, respectively. We also refer to the corresponding pixel-space operations in DVC* as PS-motion and PS-residual.

To verify the effectiveness of our proposed feature-level operations, in Fig. 7(b) we report the results of two variants of our FVC-basic. **(1) FVC-basic (FS-motion & PS-residual)**. The predicted feature $\bar{F}_t$ after our deformable compensation module is converted as the predicted frame by using our frame reconstruction module, which is then used as the input to the pixel-space residual compression module in DVC*. **(2) FVC-basic (PS-motion & PS-residual**. We perform both motion-related operations and residual compression at the pixel space, which is conceptually the same as DVC* except some subtle differences in implementation details.

It is noted that FVC-basic outperforms FVC-basic (FS-motion & PS-residual) by 0.3dB at 0.38bpp, which indicates it is beneficial to perform residual compression at the feature space. Furthermore, FVC-basic (FS-motion & PS-residual) achieves 1.2dB improvement at 0.4bpp when compared with FVC-basic (PS-motion & PS-residual), which demonstrates it is also necessary to perform motion compensation at the feature space.

Analysis of Deformable Compensation. For better comparison of the motion compensation module in the feature space and the pixel space, we also take the predicted feature $\bar{F}_t$ as the input of our frame reconstruction module to produce the predicted frame and evaluate the PSNR of the predicted frame and the corresponding bpp for compressing motion information. As shown in Fig. 8, the predicted

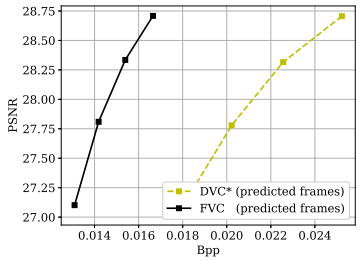
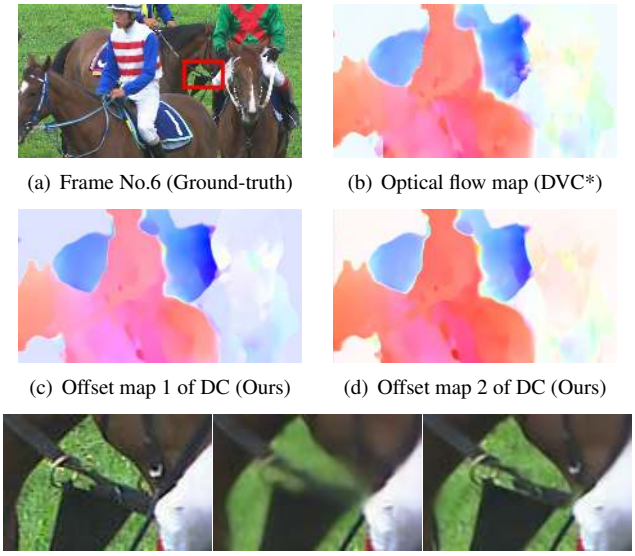


Figure 8. Average PSNR(dB) over all predicted frames and bpps used to encode motion information of our FVC and DVC* on the HEVC Class C dataset.



(a) Frame No.6 (Ground-truth) (b) Optical flow map (DVC*)

(c) Offset map 1 of DC (Ours) (d) Offset map 2 of DC (Ours)

(e) One patch (GT) (f) Flow (26.68/0.0346) (g) DC (26.95/0.0223)

Figure 9. Visualization of the results by using optical flow based motion compensation (b,f) and our deformable compensation (c,d,g) for the 6th frame (a) in the *RaceHorses* sequence from the HEVC Class C dataset. (b): the reconstructed optical flow map from the motion compression network of DVC*. (c,d): two representative reconstructed offset maps used for deformable convolution (DC) in our deformable compensation module. Visualization results of one zoomed-in patch from: the ground-truth (e), the flow based motion compensation module from DVC* (f) and our deformable compensation module (g), in which both PSNR and bpps for coding motion information are reported.

frames of our proposed FVC achieves 1.75dB improvement at 0.017bpp on the HEVC Class C dataset when compared with the predicted frames of DVC*. It should be mentioned that DVC* uses the same compression method based on the same auto-encoder style network as our proposed method FVC, so the result clearly demonstrates the effectiveness of our proposed deformable compensation module.

Visualization of Deformable Compensation Results. In Fig. 9, we take the 6th frame in the *RaceHorses* sequence from the HEVC Class C dataset as an example and visualize motion information and the motion compensation results. We visualize the optical flow map used for motion compensation in DVC* (see Fig. 9(b)) and two representative offset maps (from the total number of 72 offset maps) used for deformable convolution in our work (see Fig. 9(c) and Fig. 9(d)). It can be observed that the learned offset maps in our work encode similar motion patterns as the optical flow map from DVC*. We also observe that the motion compensation result for one patch after using our proposed deformable compensation (DC) module are visually more similar to the ground-truth patch when compared with that by using the optical flow based motion compensation method in DVC*. In practice, we can achieve 0.27dB gain by only using 65% bpp for motion coding (see Fig. 9(e), Fig. 9(f) and Fig. 9(g)).

Running Time and Model Complexity. The total number of parameters in our proposed framework is about 26M, in which the parameters from the compression network used for offset maps and residual feature maps take more than 24M. We use the videos with the resolution of 1920×1080 to evaluate the inference time on the machine with a single 2080TI GPU (11GB Memory). The coding time of our proposed frameworks FVC and FVC-basic as well as DVC and DVC* are 548ms, 201ms, 460ms and 709ms, and the corresponding BDBRs on the HEVC Class D dataset by using H.265 as the anchor method are -18.39%, -7.09%, 14.08% and 4.31%. From the results, we observe that our FVC-basic outperforms DVC and DVC* in terms of both efficiency and effectiveness. In addition, our proposed framework FVC achieves the best BDBR result with 347ms used for the multi-frame feature fusion module, which can be accelerated by using a simpler fusion module in our future work.

5. Conclusion

In this work, we have proposed a new framework FVC for deep video compression in the feature space, which consists of the deformable compensation module, the feature-level residual compression module and the multi-frame feature fusion module. The deformable compensation module first predicts the offset maps as motion information, which are then compressed by using the newly proposed motion compression module and the reconstructed offset maps will be finally used in deformable convolution for motion compensation. The proposed multi-frame feature fusion module takes multiple feature representations from the current frame and the previous frames and uses the deformable compensation and the non-local attention mechanism to refine the initial reconstructed feature for better frame reconstruction. By performing all the operations in the feature space, our framework achieves promising results on the HEVC, UVG, VTL and MCL-JCV datasets.

Acknowledgement This work was supported by the National Key Research and Development Project of China (No. 2018AAA0101900).

References

[1] Video trace library. http://trace.eas.asu.edu/yuv/index.html, 2001. Accessed: 2020-11-11. 5, 6

[2] Eirikur Agustsson, David Minnen, Nick Johnston, Johannes Ballé, Sung Jin Hwang, and George Toderici. Scale-space flow for end-to-end optimized video compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8503–8512, 2020. 1, 2, 6

[3] Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool. Generative adversarial networks for extreme learned image compression. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 221–231, 2019. 1

[4] Johannes Ballé, Valero Laparra, and Eero P Simoncelli. End-to-end optimized image compression. *International Conference on Learning Representations (ICLR)*, 2017. 2

[5] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. *International Conference on Learning Representations (ICLR)*, 2018. 2, 5

[6] Fabrice Bellard. BPG image format. *URL https://bellard.org/bpg*, 2015. 2

[7] Gisle Bjontegaard. Calculation of average PSNR differences between RD-curves. *VCEG-M33*, 2001. 7

[8] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Learning image and video compression through spatial-temporal energy compaction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10071–10080, 2019. 2

[9] Jifeng Dai, Haozhi Qi, Y. Xiong, Y. Li, Guodong Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 764–773, 2017. 1, 2, 4

[10] Abdelaziz Djelouah, Joaquim Campos, Simone Schaub-Meyer, and Christopher Schroers. Neural inter-frame compression for video coding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6421–6429, 2019. 1, 2, 4, 6

[11] Runsen Feng, Yaojun Wu, Zongyu Guo, Zhizheng Zhang, and Zhibo Chen. Learned video compression with feature-level residuals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 120–121, 2020. 2, 4

[12] Amirhossein Habibian, Ties van Rozendaal, Jakub M Tomczak, and Taco S Cohen. Video compression with rate-distortion autoencoders. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7033–7042, 2019. 2

[13] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 4

[14] Zhihao Hu, Zhenghao Chen, Dong Xu, Guo Lu, Wanli Ouyang, and Shuhang Gu. Improving deep video compression by resolution-adaptive flow coding. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1, 2, 4, 6, 7

[15] Nick Johnston, Damien Vincent, David Minnen, Michele Covell, Saurabh Singh, Troy Chinen, Sung Jin Hwang, Joel Shor, and George Toderici. Improved lossy image compression with priming and spatially adaptive bit rates for recurrent networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4385–4393, 2018. 2

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference for Learning Representations*, 2015. 6

[17] Jan P Klopp, Liang-Gee Chen, and Shao-Yi Chien. Utilising low complexity cnns to lift non-local redundancies in video coding. *IEEE Transactions on Image Processing*, 2020. 2

[18] Jun Li, Xianglong Liu, Mingyuan Zhang, and Deqing Wang. Spatio-temporal deformable 3d convnets with attention for action recognition. *Pattern Recognition.*, 98, 2020. 2

[19] Jianping Lin, Dong Liu, Houqiang Li, and Feng Wu. M-LVC: Multiple frames prediction for learned video compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3546–3554, 2020. 1, 2

[20] Salvator Lombardo, Jun Han, Christopher Schroers, and Stephan Mandt. Deep generative video compression. In *Advances in Neural Information Processing Systems*, pages 9287–9298, 2019. 2

[21] Guo Lu, Chunlei Cai, Xiaoyun Zhang, Li Chen, Wanli Ouyang, Dong Xu, and Zhiyong Gao. Content adaptive and error propagation aware deep video compression. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1, 2, 6, 7

[22] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao. DVC: An end-to-end deep video compression framework. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11006–11015, 2019. 1, 2, 4, 6, 7

[23] Guo Lu, Xiaoyun Zhang, Wanli Ouyang, Li Chen, Zhiyong Gao, and Dong Xu. An end-to-end learning framework for video compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, in Press:1–1, 2020. 1, 2

[24] Alexandre Mercat, Marko Viitanen, and Jarno Vanne. UVG dataset: 50/120fps 4k sequences for video codec analysis and development. *Proceedings of the 11th ACM Multimedia Systems Conference*, 2020. 5, 6

[25] David Minnen, Johannes Ballé, and George D Toderici. Joint autoregressive and hierarchical priors for learned image compression. In *Advances in Neural Information Processing Systems*, pages 10771–10780, 2018. 2, 5

[26] Oren Rippel, Sanjay Nair, Carissa Lew, Steve Branson, Alexander G Anderson, and Lubomir Bourdev. Learned video compression. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3454–3463, 2019. 2

[27] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on circuits and systems for video technology*, 22(12):1649–1668, 2012. 1, 2, 5, 6, 7

[28] David S Taubman and Michael W Marcellin. JPEG2000: Standard for interactive imaging. *Proceedings of the IEEE*, 90(8):1336–1357, 2002. 2

[29] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. *International Conference for Learning Representations*, 2017. 2

[30] Yapeng Tian, Yulun Zhang, Yun Fu, and Chenliang Xu. TDAN: Temporally-deformable alignment network for video super-resolution. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3357–3366, 2020. 2

[31] George Toderici, Sean M O'Malley, Sung Jin Hwang, Damien Vincent, David Minnen, Shumeet Baluja, Michele Covell, and Rahul Sukthankar. Variable rate image compression with recurrent neural networks. *International Conference for Learning Representations*, 2017. 2

[32] George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor, and Michele Covell. Full resolution image compression with recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5306–5314, 2017. 2

[33] Gregory K Wallace. The JPEG still picture compression standard. *IEEE transactions on consumer electronics*, 38(1):xviii–xxxiv, 1992. 2

[34] Haiqiang Wang, Weihao Gan, Sudeng Hu, Joe Yuchieh Lin, Lina Jin, Longguang Song, Ping Wang, Ioannis Katsavounidis, Anne Aaron, and C-C Jay Kuo. MCL-JCV: a JND-based H.264/AVC video quality assessment dataset. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 1509–1513. IEEE, 2016. 5, 6

[35] Xintao Wang, Kelvin C. K. Chan, K. Yu, C. Dong, and Chen Change Loy. EDVR: Video restoration with enhanced deformable convolutional networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1954–1963, 2019. 1, 2

[36] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. IEEE, 2003. 5, 6

[37] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on circuits and systems for video technology*, 13(7):560–576, 2003. 1, 2, 6

[38] Chao-Yuan Wu, Nayan Singhal, and Philipp Krahenbuhl. Video compression through image interpolation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 416–431, 2018. 1, 2

[39] Yi Xu, Longwen Gao, Kai Tian, Shuigeng Zhou, and Huyang Sun. Non-local convlstm for video compression artifact reduction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7043–7052, 2019. 2

[40] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. Video enhancement with task-oriented flow. *International Journal of Computer Vision*, 127(8):1106–1125, 2019. 5

[41] Ren Yang, Fabian Mentzer, Luc Van Gool, and Radu Timofte. Learning for video compression with hierarchical quality and recurrent enhancement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6628–6637, 2020. 2

[42] Ren Yang, Fabian Mentzer, Luc Van Gool, and Radu Timofte. Learning for video compression with recurrent auto-encoder and recurrent probability model. *arXiv preprint arXiv:2006.13560*, 2020. 2