Leveraging Bitstream Metadata for Fast, Accurate, Generalized Compressed Video Quality Enhancement

Max Ehrlich[1], Jon Barker[1], Namitha Padmanabhan[2], Larry Davis[2], Andrew Tao[1], Bryan Catanzaro[1], Abhinav Shrivastava[2]
[1]NVIDA [2]University of Maryland, College Park
{mehrlich, jbarker}@nvidia.com {namithap, lsdavis}@umd.edu {atao, bcatanzaro}@nvidia.com abhinav@cs.umd.edu

Abstract

Video compression is a central feature of the modern internet powering technologies from social media to video conferencing. While video compression continues to mature, for many compression settings, quality loss is still noticeable. These settings nevertheless have important applications to the efficient transmission of videos over bandwidth constrained or otherwise unstable connections. In this work, we develop a deep learning architecture capable of restoring detail to compressed videos which leverages the underlying structure and motion information embedded in the video bitstream. We show that this improves restoration accuracy compared to prior compression correction methods and is competitive when compared with recent deep-learning-based video compression methods on rate-distortion while achieving higher throughput. Furthermore, we condition our model on quantization data which is readily available in the bitstream. This allows our single model to handle a variety of different compression quality settings which required an ensemble of models in prior work.

1. Introduction

At its conception, the internet was a medium for the exchange of text data. This has rapidly changed over the last decade to focus on multimedia, and in particular, video [1], [2]. Video compression is, therefore, a critical feature of the modern internet. Even short videos are orders of magnitude larger than text data in their uncompressed form and would be impossible to transmit in a timely manner over even a broadband connection.

Although modern block-based codecs [3]–[7] are able to achieve impressive compression ratios with limited quality loss, even these codecs are challenged by bandwidth-limited scenarios which are common in third world countries, rural locations, and lower-class households [8]. Moreover, the additional transmission latency induced by a low-bandwidth internet connection is unstable: effective bandwidth can vary greatly over time. One way to overcome this limitation is to increase the aggressiveness of the video encoder creating a smaller transmission, however this comes with an associated loss in visual fidelity. We solve this fidelity loss by formulating MetaBit, a novel convolutional neural network architecture [9], [10] for restoring compressed videos. An example of this is shown in Figure 1. In effect, we are trading off the unpredictable and often extreme latency of internet transmission for the measurable and predictable latency of deep learning.

Figure 1. **Don't Spend Megabits, Use MetaBit.** Our MetaBit system takes heavily compressed frames and restores detail. The above example is stored at only 0.039bpp. For each pair, our restoration is shown with a **blue** border. Our method is able to faithfully restore natural textures (**left trees**), clothing textures/human appearance (**middle woman**), and artificial textures (**top roof**).

Metabit's design is motivated by a number of oversights we identified in prior work. In particular, Metabit is unique in the space of video quality enhancement models because it considers information present in the raw video bitstream. Since the problem we are solving is one caused by video compression, we find it natural to leverage this information

Figure 2. **Motion Vectors.** Motion vectors resemble downsampled optical flow. Left: reference image. Middle: optical flow. Right: motion vectors extracted from the video bitstream. Optical flow was computed with RAFT [11].

which describes in detail how the encoder degraded the given video. We view this metadata as a set of "hints" which aid our network in the restoration process. We leverage these hints to improve both the speed and fidelity of the restoration process by directly addressing specific design decisions in prior video quality enhancement models.

Firstly, prior works expend significant resources on either explicit [12], [13] or implicit [14]–[16] motion estimation. This is a resource intensive task for a network which can be entirely avoided by leveraging *motion vectors* from the bitstream that provide the motion information with no computation. Secondly, although Yang *et al*. [12] correctly observe that not all frames contain the same amount of information, they rely on explicit supervision, and train a discriminative model, to determine the frames with the most information which makes training cumbersome. The idea of using reference frames reappears several times in more recent works [15], [16]. In many cases what these models are using are simply *Intra-Frames* (I-frames) which the encoder intentionally stores at higher quality to use as references for decoding later frames. The position of I-frames is explicitly stored in the compressed bitstream so these high-quality frames can be identified with no computation. Additionally, by differentiating I- and P- frames, we can allocate more parameters to the I-frames and less parameters to the P-frames consequently accelerating the entire architecture. This also introduces a new restoration paradigm: where prior works are sliding-window methods which take a group of frames and produce a single output frame, our method restores blocks of frames at a time, *i.e*., in a single forward pass, our network consumes 7 degraded frames and predicts 7 high quality frames. This paradigm is much faster than the sliding window methods.

Another critical drawback of all prior works in this space, and one which makes them cumbersome to use in real scenarios, is the requirement that a new model be trained per constant *quantization parameter* (QP) setting. QP settings generally range from 0-51 meaning that in the worst case 51 different models would need to be trained. This also precludes the use of compression methods, like constant bitrate (CBR) and constant rate-factor (CRF), that allow QPs to vary over time and, potentially, space. Since these two compresion methods are in widespread use compared to the constant QP method, prior art is quite difficult to use on real videos. Luckily, the video bistream contains this quantization data as well, so we forumulate a QP cross attention model that reads the potentially time-and-space varying QP map from the bitstream and directs the restoration blocks to adapt their feature maps to varying quantization. This allows a single model, which is easy to train and deploy, to outperform the prior works which depend on an ensemble of models.

Finally, prior works use a limited benchmark of only the H.265 (HEVC) [5] compression algorithm with constant QP compression (see Section 3 for a detailed discussion). The HEVC codec saw very limited use historically and is all but deprecated by more modern codecs. The legacy H.264 (AVC) codec [3] currently handles approximately 90% of internet compressed videos [1]. In addition to being more common, it generates more noticeable degradations especially when paired with CRF quantization (which is the default setting in ffmpeg and is therefore in widespread use). Since the degradations of this codec are more severe, correction models are more useful. Although we benchmark on the HEVC codec for comparison purposes (Appendix A), we additionally report results using the more common AVC codec using CRF encoding (see Section 5.4) and show that prior works in general fail to generalize to these more complex degradations.

We additionally propose new loss functions. In particular, we formulate a *Scale-Space* [17] loss that allows the network to focus on high frequency details which are removed by compression and we use a GAN [18] loss which enables the network to hallucinate plausible reconstructions. Finally, we extend our comparison to include fully deep-learning-based compression codecs and find that simply using AVC, a widely supported codec, with our restoration network is competitive in terms of rate-distortion and decoding time.

In summary, our contributions are:

1. An efficient formulation for video compression correction which leverages the underlying bitstream structure of compressed videos to achieve state-of-the-art performance.
2. A method which requires only a single model to handle a range of different quality settings
3. A more rigorous evaluation procedure which includes tests on realistic compression settings.
4. Improved loss formulations which allow the network to produce plausible reconstructions in extreme compression scenarios.

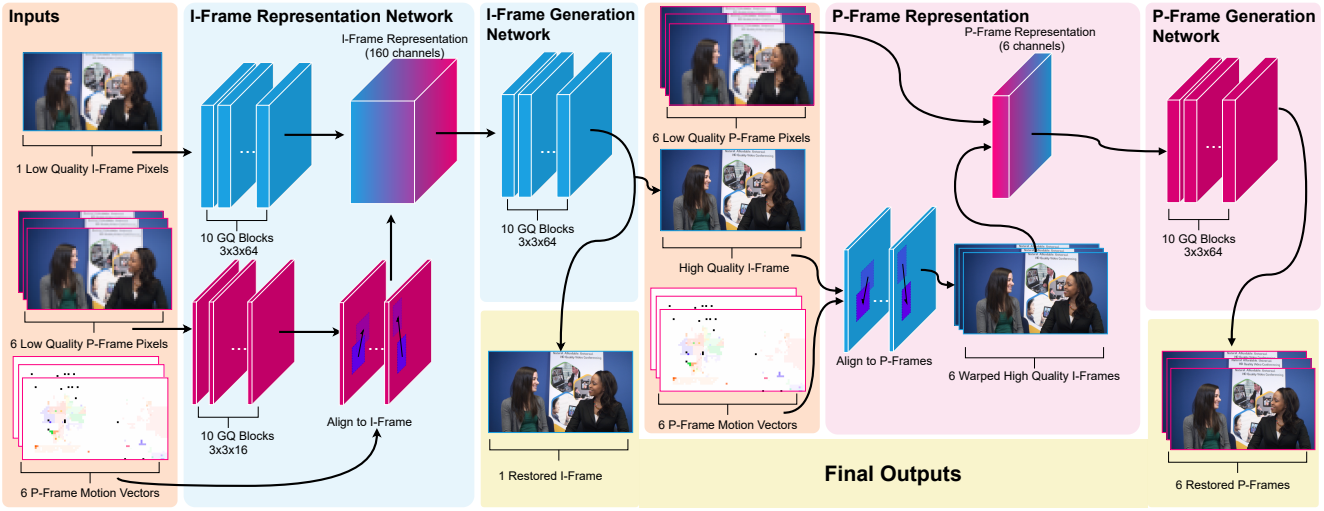

Figure 3. **MetaBit System Overview.** I-Frames are shown in **Blue** and P-Frames are shown in **Pink**. Our network takes an input (**Orange**) in the form of a low-quality Group-of-Pictures and first performs multi-frame correction on the **I-Frame**. The resulting high-quality **I-Frame** is used to guide correction of the low-quality **P-Frames**. The final output of our network (**Yellow**) is the entire high-quality Group-of-Pictures. Please see Section 3 for an overview of all terminology.

2. Prior Work

JPEG Artifact Correction The related problem of JPEG [19] artifact correction is a rich area of study with consistent progress each year. In recent years this problem is solved using convolutional neural networks [9], [10]. ARCNN [20] is the first such method which was a simple regression technique inspired by super-resolution architectures. These works were later extended to "dual-domain" methods [21]–[25] One flaw in these works was their focus on "quality-aware" formulations, in other words. This was solved by Ehrlich *et al*. [26], [27] using a formulation which was conditioned on the JPEG quantization matrix and later improved by Jiang *et al*. [28] where the network was encouraged to correctly predict the JPEG quality.

Video Restoration Video compression correction is directly related to other video restoration tasks. Toflow [29] used optical flow which is trained end-to-end with the restoration task to align frames and operated as a sliding window. EDVR [30] also operates as a sliding window but replaces the explicit motion estimation with deformable convolutions [31]. Chan *et al*. [32] analyze critical components of super-resolution and use this to design a simple, flexible architecture. Relevant here, Li *et al*. [33] consider super-resolution with common video compression settings. S2SVR [34], [35] propose a novel unsupervised sequence-to-sequence model which alleviates many of the concerns about sliding-window techniques discussed earlier. Xiao *et al*. [36] propose kernel grafts as a way to bypass complex networks by transferring their learned representation into a series of lightweight kernels. Lin *et al*. [37] improve HEVC video decoding by incorporating a superresolution network into the video decoder.

Video Quality Enhancement Video quality enhancement, the task we solve here, was initially solved using "single-frame" enhancement methods [38], [39] which outperform image-based restoration techniques but use only a single frame at a time. Yang *et al*. [12] propose MFQE which takes multiple frames in a sliding window to correct an entire video sequence. In addition to being the first multi-frame video compression correction model, their key contribution is the concept of *Peak Quality Frames* (PQFs). These are individual frames that have a higher perceptual quality than other nearby frames, and they are identified using a manually trained SVM [40]. They combine information from nearby frames using pixel-wise motion estimation and warping. Xing *et al*. [13] extend this idea by replacing the PQF detecting SVM with a BiLSTM [41]. Deng *et al*. [14] use implicit motion compensation with deformable convolutions [31]. They show that this leads to a more accurate and faster formulation. More recently, Ding *et al*. [42] design an architecture for capturing adjacent patch information more effectively and Zhao *et al*. [15] use a recurrent hidden state and deformable attention to improve the result. Xu *et al*. [16] show that STDF performance is improved by intelligently selecting reference frames. In contrast to these techniques, our method requires no motion estimation for alignment and no supervision to determine high-information frames.

3. Background

Our method leverages concepts from video compression to improve both processing speed and restoration quality over prior works. Note that while we have selected AVC for evaluations, our method depends on information found in all codecs and is equally applicable to VP8/9, AV1, *etc*., and nothing presented in this section is codec specific.

Group of Pictures Modern video encoders pack information over time into a Group of Pictures (GOP) based on the assumption that over a small time interval motion, and therefore the difference between frames, is small. This yields two types of frames: *Intra-frames* (I-frames), used as reference images, and *Predicted-frames* (P-frames), which require a reference to decode. I-frames are so called because they can be decoded using only information contained within the frame similar to a still-image. P-frames contain two major components: *Motion Vectors* (discussed in the next section) and *Error Residuals*. Error residuals are the difference image between the motion-compensated frame and the true frame. They encode all new information that could not be modeled by motion. As I-frames contain most of the information for a GOP, we allocate more parameters to the representation and generation of the high quality I-frame and use that result to guide restoration of the low-information P-frames.

Motion Compensation Video codecs include coarse heuristic motion estimation in the encoding process. These motion vectors are computed on blocks of pixels. See Figure 2 for a visual comparison of motion vectors to optical flow. This operation alone compresses blocks of pixels into 4 tuples of source and destination while also reducing the entropy of the error residual. Our network reads and applies these coarse motions for alignment in lieu of pixelwise flows which would need to be computed.

Tuning Quality *vs*. Bitrate Modern codecs provide several methods for tuning the perceptual quality of a video stream. By removing information (which lowers the perceptual quality), the codec is able to further compress the stream resulting in a smaller file. The most common methods are *Constant Rate Factor* (CRF) and *Constant Bitrate* (CBR) with CRF being the default method in many implementations [43]. In the CRF paradigm, the user presents the encoder with an integer in $[0, 51]$ with higher numbers indicating lower quality. The CRF number is considered a "proxy" for perceptual quality. In CBR mode, the encoder is asked to target a specific bitrate in bits-per-second (BPS). This mode is commonly used if a stream is to be transmitted over a connection of known maximum bandwidth. The uncommon *Constant Quantization Parameter* (CQP) method is the mode tested by prior works. In both of the above cases, the encoder converts the user input (CRF or target bitrate) to a set of "QPs" which are used to quantize transform coefficients. These QPs generally vary over space and time. In CQP encoding, the user provides a single QP directly and the encoder uses it for all blocks in all frames without regard for information content. Use of this method is generally discouraged since the encoder is no longer making intelligent decisions about which information to keep or discard. However, this method simplifies machine learning solutions because a single QP incurs a predictable degradation. In contrast, varying QPs over space and time (as in CRF and CBR) incur different degradations even within the same frame. Since CQP encoding is so uncommon and discouraged we believe this benchmark is unrealistic and instead study the default CRF encoding.

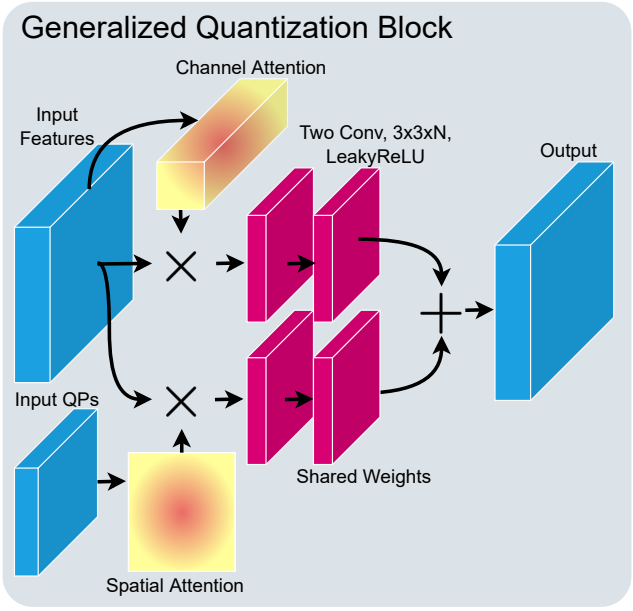

Figure 4. **Generalized Quantization Block.** Our basic block is designed to be efficient while merging information from the input feature maps and varying quantization data. The block takes the Quantization Parameters (Input QPs) and uses cross-attention to adapt features to varying levels of compression.

4. Method

Our task is to take a compressed frame and compute a restored network output which is as close as possible to the target (uncompressed) frame. We accomplish this with a novel multiframe restoration network and loss functions. One unique aspect of compression restoration, when compared to other problems such as denoising, is that we have an exact target frame and we know the exact procedure that was performed on the target frame which caused the degradation. This allows us to design a network architecture that is informed by this procedure, and in our case, incorporate metadata contained in the compressed video bitstream.

In particular, multiframe restoration methods are known to improve when features in different frames are aligned (see Section 5.6 for an ablation of this). Rather than compute motion estimation, we leverage the motion vectors contained in the video bitstream. Additionally, video compression allocates more information to I-frames than to P-frames (Section 3). This was reflected in the "Peak-Quality Frames" of MFQE [12], [13] which required explicit supervision. Instead of computing the locations of these frames, our method simply assumes they are I-Frames[1]. By eliminating the need

[1]Examining charts provided by Yang *et al*. [12] shows that the PQFs

to perform explicit motion estimation and detect high quality reference frames, we can reinvest the requisite parameters directly in the restoration task leading to high quality results. In the remainder of this section we describe our novel restoration procedure and loss formulations. The procedure is shown graphically in Figure 3.

4.1. Restoration Procedure

GQ Blocks We build a novel basic-block to perform restoration on video frames with varying quantization. We call this the Generalized-Quantization (GQ) block, which is illustrated in Figure 4. The design consists of two parallel branches with two convolutional layers each that share weights. The top branch uses channel self-attention to attenuate the most informative input channels. Meanwhile, the bottom branch uses cross-attention computed on the input QP map to adapt the input features to the spatially-varying quantization. The output features are summed to produce the final output of the block. These blocks are stacked to produce the larger structures in our network.

I- and P-Frame Representations Our network first performs a multi-frame restoration on I-frames. Since the I-frame itself contains most of the information in a group-of-pictures, we compute a 64-dimensional representation using 10 GQ blocks. We then compute a 16-dimensional representation of the P-frames using 10 GQ blocks. This process is shown in the top part of the blue box in Figure 3.

Motion Vector Alignment We then align the P-frame representations to the I-frame representation by warping the P-frame features using motion vectors. Each P-frame contains motion vectors that copy blocks of pixels from the previous frame into new locations in the destination frame. We reverse the direction of these vectors and copy the P-frame features backwards to align with the previous frame. Repeating this process for all motion vectors in the group-of-pictures yields a volume of P-frame features which are coarsely aligned with the I-frame.

I-Frame Generation We concatenate the I-frame features and aligned P-frame features channel-wise to yield a volume containing aligned features for the entire group-of-pictures. For a 7 frame GOP, this is a 160-dimensional representation which we project to 64 dimensions for efficiency. We then generate the high-quality I-frame using 10 more GQ blocks with the final one yielding the 3 channel output.

P-Frame Generation Finally, we use the high-quality I-frame to generate the high-quality P-frames. Each I-frame is warped using the P-frame motion vectors to yield 6 copies of the I-frame, each one aligned to one of the low-quality P-frames. These warps are concatenated channel-wise with the low-quality P-frames to create a 6 channel input. This is projected to a 64-dimensional feature space and then processed using 10 GQ blocks to yield the high-quality P-frame. This process is shown in the pink boxes of Figure 3.

they detect are likely I-Frames due to their regular spacing, although we do not analyze this here.

4.2. Loss Functions

Restoring videos which were subject to extreme compression is a challenging problem. In general, we found that traditional regression losses alone produce a blurry result. This is directly caused by lossy compression's preference for removing high frequency details which is true for both images and videos. We use two loss functions during training in order to solve this problem.

Regression Loss We use the l_1 error as our regression loss. For network output O and target frame T we compute

$$\mathcal{L}_1(O,T) = ||T - O||_1 \tag{1}$$

Scale-Space Loss We use a loss based on the Difference of Gaussians (DoG) scale space [17]. The DoG is a fast approximation to the Laplacian of Gaussians and as such functions as a band-pass filter. By isolating these frequency bands and weighting their error equally, the network is encouraged to generate images which match in more than just the low-frequency regions. Formally, given network output O and target frame T, we compute 4 scales by downsampling O, yielding the scale space

$$S = \{O, O_2, O_4, O_8\} \tag{2}$$

where entry O_s is obtained by downsampling O by the factor s in both the width and height. We then compute the DoG by convolving each entry in S with 5×5 2D gaussian kernels of increasing standard deviation σ:

$$G(\sigma)_{ij} = \frac{1}{2\pi\sigma^2} e^{-\frac{i^2+j^2}{2\sigma^2}} \tag{3}$$

for kernel offsets i, j. For each scale s we compute four filtered images

$$\begin{aligned} I_{O,s,1} &= G(1.1) * O_s \\ I_{O,s,2} &= G(2.2) * O_s \\ I_{O,s,3} &= G(3.3) * O_s \\ I_{O,s,4} &= G(4.4) * O_s \end{aligned} \tag{4}$$

where $*$ is the discrete, valid cross-correlation operator. We then compute the difference

$$\begin{aligned} B_{O,s,1} &= I_{O,s,2} - I_{O,s,1} \\ B_{O,s,2} &= I_{O,s,3} - I_{O,s,2} \\ B_{O,s,3} &= I_{O,s,4} - I_{O,s,3} \end{aligned} \tag{5}$$

to yield the per-scale frequency bands. This process is repeated for the target image yielding B_T. The final loss is then the sum of absolute error between the frequency bands

$$\mathcal{L}_{\text{DoG}}(O,T) = \sum_{s \in \{1,2,4,8\}} \sum_{b=1}^{3} ||B_{T,s,b} - B_{O,s,b}||_1 \tag{6}$$

Table 1. **Quantitative Evaluation**. We report ΔPSNR (dB) ↑ / ΔSSIM ↑ / ΔLPIPS ↓, averaged over the MFQE [12] test split.

Method	CRF		
	35	40	50
MFQE 2.0 [13]	0.681 / 0.015 / 0.004	0.660 / 0.019 / -0.001	0.538 / 0.023 / -0.015
STDF-R1 [14]	0.862 / 0.011 / 0.032	0.814 / 0.015 / 0.030	0.632 / 0.023 / 0.013
STDF-R3L [14]	0.846 / 0.010 / 0.032	0.882 / 0.015 / 0.029	0.817 / 0.027 / 0.011
RFDA [15]	0.273 / 0.006 / 0.012	0.395 / 0.011 / 0.006	0.458 / 0.020 / -0.021
MetaBit (Ours)	**0.958 / 0.023 / -0.001**	**1.032 / 0.031 / -0.018**	**0.877 / 0.042 / -0.041**

Table 2. **Throughput.** We measure throughput (FPS) on an *NVIDIA GTX 1080 Ti* GPU. Despite having nearly twice as many parameters our network is faster than or on-par with prior works and still able to run on consumer hardware.

Method	240p	480p	720p	1080p	Parameters (M)
MFQE 2.0	25.3	8.4	3.7	1.7	0.255
STDF-R1	38.9	9.9	4.2	1.8	0.330
STDF-R3L	23.8	5.9	2.5	1.0	1.275
RFDA	24.0	6.0	2.6	1.0	1.250
MetaBit (Ours)	26.9	5.4	2.2	1.0	2.449

Table 3. **GAN Scores**. The GAN loss allows our network to generate more realistic images than the regression loss alone. Regression loss leads to worse FID scores caused by the smooth, textureless appearance of scene elements. Format is FID ↓ / LPIPS ↓

Method	CRF	
	40	50
AVC (Degraded Input)	67.07 / 0.289	152.19 / 0.511
MetaBit (Regression)	80.67 / 0.272	154.42 / 0.470
MetaBit (GAN)	**37.78 / 0.191**	**95.26 / 0.368**

GAN and Texture Losses We use the Wassertein GAN formulation $\mathcal{L}_{\text{W}}(O,T)$ [44] with a critic modeled after DCGAN [45], which we modified using the procedure in Chu *et al*. [46] to introduce temporal consistency (see Appendix B for more GAN details). We include a texture loss [26] which replaces the traditional ImageNet trained perceptual loss with a VGG [47] network trained on the MINC materials dataset [48]. Intuitively, if the images are encouraged to produce similar logits from this MINC-trained VGG, then it is likely the two images would be classified as the same material and therefore have similar textures. We compare feature maps from layer 5 convolution 3 of this VGG network. Formally:

$$\mathcal{L}_{\text{texture}}(O,T) = ||\text{MINC}_{5,3}(T) - \text{MINC}_{5,3}(O)||_1 \quad (7)$$

Composite Loss Functions This yields the following two loss functions, a regression loss

$$\mathcal{L}_R(O,T) = \alpha\mathcal{L}_1(O,T) + \beta\mathcal{L}_{DoG}(O,T) \quad (8)$$

which is used for regression-only experiments, and a GAN loss

$$\begin{aligned}\mathcal{L}_{\text{GAN}}(O,T) = \alpha\mathcal{L}_1(O,T) + \beta\mathcal{L}_{\text{DoG}}(O,T) + \\ \gamma\mathcal{L}_{\text{W}}(O,T) + \delta\mathcal{L}_{\text{texture}}(O,T)\end{aligned} \quad (9)$$

where $\alpha, \beta, \gamma, \delta$ are balancing hyperparameters.

5. Experiments and Results

The final architecture as described in Section 4 contains nearly twice the parameters of the previous state-of-the-art without performing any motion estimation or detection of high quality frames and restores 7 frame blocks. We now show empirically that this formulation works by comparing to prior works on realistic benchmarks. Our method does this while maintaining roughly the same (or better) throughput as the previous methods.

We note two things about our claims here. The first is that the architecture runs similarly-or-faster than models with lower parameter counts because it does not depend on a sliding window and does not need to allocate resources to motion estimation. The second is that this efficient architecture *allows us to allocate more parameters to the model* in order to improve benchmark performance with negligible penalties on speed. This synergizes with our novel loss function and our use of quantization parameters.

5.1. Datasets

We train on the MFQE dataset [12] training split (108 variable length sequences). We randomly crop 256×256 patches from each example and apply random horizontal and vertical flipping. We encode the resulting sequence with a 7 frame GOP and no B-frames, thus yielding one I-frame and 6 P-frames per example. We use CRF encoding with auto-variance adaptive quantization for benchmarking. Please see Appendix C for the exact compression commands we used. We evaluate on the MFQE test split. This consists of 18 variable-length sequences (7890 frames) commonly used for evaluation of compression algorithms and was proposed by the Joint Collaborative Team on Video Coding [49].

5.2. Training Procedure

Our network is implemented using PyTorch [50] and trained end-to-end for 600 epochs using the Adam optimizer [51] with a learning rate of 10^{-4}. We lower this learning rate to zero over the last 200 epochs using cosine annealing [52].

Figure 5. **Qualitative Results.** Please zoom in to view fine details. Note the increased quality of the MetaBit model over STDF and the enhanced sharpness and textures of the GAN method. This is particularly apparent on the trees (row 1), car (row 2), grass (row 3), and the wood texture (row 4). Additional qualitative results are shown in Appendix F and the attached supplement.

For quantitative benchmarks, we train using the regression loss (Equation 8) with $\alpha = 1.0, \beta = 1.0$. Note that there is not special balancing that is done in the regression loss formulation.

For GAN training, we begin with regression weights and fine-tune the entire network using our GAN loss (Equation 9) with $\alpha = 0.01.\beta = 0.01, \gamma = 0.005, \delta = 1$. We train for an additional 200 epochs with a learning rate of 10^{-5} and the RMSProp optimizer. Please see Appendix B for additional details on the GAN architecture. In this case we use the balancing parameters only to keep the GAN loss from becoming unstable and diverging, this was done by fixing δ, lowering α, β together by one order of magnitude, and then lower γ until training consistently converged.

For regression we report the change in PSNR, SSIM [53], and LPIPS [54] averaged over all frames in the test set. For GAN evaluation, we report the average FID [55] of the compressed and restored frames.

5.3. Quantitative Results

We compare to MFQE 2.0 [13], STDF [14], and RFDA [15]. These methods were all re-trained using publicly available code. We do not compare with single-frame video restoration methods or image restoration methods which were found to have objectively worse performance than the multiframe methods. Please see Appendix A for more details about compared methods and how we chose them. Table 1 shows our quantitative results. We test STDF in both the R1 (3-frame sliding window) and the R3 (7-frame sliding window) setting. Of note here is the RFDA result which despite being the most recent compared method made limited gains. This appears to be caused by the larger temporal range

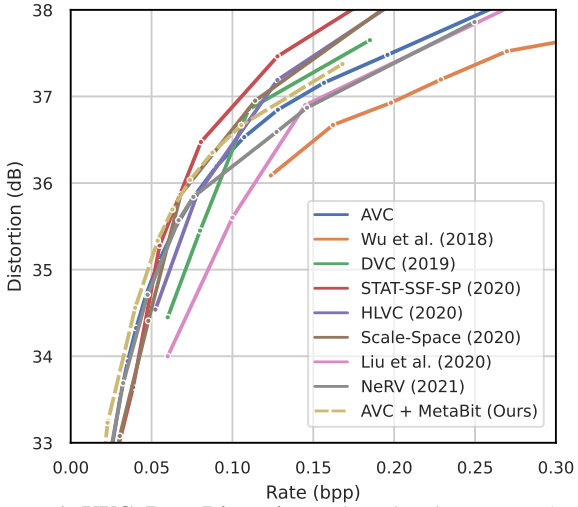

Figure 6. **UVG Rate-Distortion.** Distortion is measured with PSNR. Using AVC + MetaBit method surpasses recent fully deep-learning codecs at low bitrates. As expected, the improvement reduces as bitrate increases. Rate here is "bits-per-pixel" (bpp).

Table 4. **Compression Throughput.** We measure FPS on an *NVIDIA GTX 2080 Ti* GPU compared to recent deep-learning-based codecs on 1080p frames. For encoding, our method uses AVC and there is no GPU requirement. NeRV [56] encoding speed is not directly reported but requires training a unique network per video.

	Wu *et al*. [57]	DVC [58]	Liu *et al*. [59]	NeRV [56]	**MetaBit (Ours)**
Decoding 10^{-3}	1.8	3	**12.5**	3.42	
Encoding 2.4	1.5	2	-	**52**	

that RFDA uses as hidden states are accumulated over time. The temporally varying quality of CRF encoding became a confounding factor.

Overall MetaBit makes an advancement in all metrics over prior works, often by a large margin. This is particularly noteworthy because we train only a single model to produce all the results in Table 1 whereas all prior works required a model per CRF setting. Since our model is conditioned on the QP map, it is able to adapt to spatially and temporally varying degradations in ways that prior works cannot. In particular, MetaBit was the only method which improves perceptual quality (LPIPS) on all three tested CRF values.

We also provide throughput results in Table 2. Note that despite having more parameters and better restoration results, our method achieves similar throughput to STDF, even exceeding it in some cases. We are also comparable in speed to MFQE 2.0 which has 9 times fewer parameters. These tests were performed in like-conditions to those reported by prior works to control for the compute environment.

5.4. GAN Correction

While the results on CRF values $\{40, 50\}$ show an improvement in quantitative metrics, these settings represent extreme compression. We found that the regression result of our network is not visually pleasing despite the improvement, so we additionally show results using our GAN procedure. Quantitatively, FID scores in Table 3 show a significant improvement in realism with LPIPS scores similarly indicating a significant improvement in perceptual similarity. We show qualitative results in Figure 5, note the significantly improved textures, sharp edges, and additional detail introduced by the GAN loss, particularly compared to STDF.

5.5. Comparison to Learned Video Compression

One application for this work is as a stopgap technology between classical compression and fully deep-learning-based compression. This allows for the speed, memory consumption, and technical debt associated with classical compression algorithms to sustain, with bitstreams fully decodable by users who lack the computational resources for deep models. In Table 4, we compare the frame-rate of our method against recently published learned video compression algorithms, and in Figure 6, we compare rate-distortion on the UVG [60] dataset. Our method is only bested by the recent NeRV [56] for throughput (which we note has extremely long encoding times) and achieves better rate-distortion results than many compared methods especially at low bitrates.

Table 5. **Ablation.** Inference FPS is computed on an *NVIDIA GTX 1080 Ti* for 240p frames, PSNR is computed for H.264 CRF 35.

Property	Option	Result	
		ΔPSNR (dB)	FPS
Parameter Distribution	Favors I-Frames	**0.954**	**26.9**
	Even	0.938 (-1.6%)	24.3 (-9.7%)
Motion Compensation	Motion Vectors	0.948	26.9
	Optical Flow	**0.952 (+0.4%)**	17.0 (-36.8%)
Loss	l_1 and Scale-Space	**0.954**	-
	l_1 Only	0.900 (-5.6%)	-

5.6. Ablation

We ablate our design in Table 5, showing impact on throughput and reconstruction accuracy. Note that the first row in each section is the "reference" method, *i.e*., the final model tested in previous sections.

Parameter Distribution Our architecture allocates more parameters to the I-frame representation than the P-frame representation (64- *vs*. 16- dimensional). Here, we compare with an "even" distribution that allocates a 32-dimensional representation to both. This performs worse in all regards.

Motion Compensation We use video motion vectors to perform alignment. A natural comparison is using per-pixel optical flow. For optical flow, we use a pre-trained RAFT [11] model and find that indeed the fine motion detail does improve performance but at a significant throughput penalty.

Loss We claimed in Section 4.2 that our scale-space loss helps ensure a correct reconstruction of higher frequency information leading to better reconstruction accuracy. We test this and find that it indeed leads to a 5.6% improvement over not using a scale-space loss.

6. Conclusion and Future Work

We presented a novel formulation for video compression correction. Our network leverages the structure of the compressed bitstream to outperform prior works while still being extremely efficient. We proposed and tested an improved benchmark with wider applicability. This work has the potential to help people in bandwidth-constrained environments by allowing heavily compressed bitstreams to be viewable.

We hope our work will inspire additional research. Particularly: high-resolution video is slow to process and time-varying compression artifacts can introduce slight temporal inconsistencies (see video examples in the attached supplement). While our parameter count is modest in the space of deep learning models and the method runs on consumer hardware, a practical solution will need to be both smaller and faster. Nevertheless, we believe that this technology is an important stopgap between classical compression and fully deep-learning compression.

References

[1] A. Oentoro, *Video marketing statistics and strategy 2021*, Oct. 2021. [Online]. Available: https://breadnbeyond.com/video-marketing/video-marketing-strategies-statistics/.

[2] M. Duggan, *Photo and video sharing grow online*, May 2020. [Online]. Available: https://www.pewresearch.org/internet/2013/10/28/photo-and-video-sharing-grow-online/.

[3] D. Marpe, T. Wiegand, and G. J. Sullivan, "The h. 264/mpeg4 advanced video coding standard and its applications," *IEEE communications magazine*, vol. 44, no. 8, pp. 134–143, 2006.

[4] D. Mukherjee, J. Han, J. Bankoski, *et al.*, "A technical overview of vp9—the latest open-source video codec," *SMPTE Motion Imaging Journal*, vol. 124, no. 1, pp. 44–54, 2015.

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

[6] Y. Chen, D. Murherjee, J. Han, *et al.*, "An overview of core coding tools in the av1 video codec," in *2018 Picture Coding Symposium (PCS)*, IEEE, 2018, pp. 41–45.

[7] J. Bankoski, P. Wilkins, and Y. Xu, "Technical overview of vp8, an open source video codec for the web," in *2011 IEEE International Conference on Multimedia and Expo*, IEEE, 2011, pp. 1–6.

[8] B. Auxier and M. Anderson, *Social media use in 2021*, Apr. 2021. [Online]. Available: https://www.pewresearch.org/internet/2021/04/07/social-media-use-in-2021/.

[9] Y. LeCun, B. E. Boser, J. S. Denker, *et al.*, "Handwritten digit recognition with a back-propagation network," in *Advances in neural information processing systems*, 1990, pp. 396–404.

[10] I. Sutskever, G. E. Hinton, and A. Krizhevsky, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[11] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *European conference on computer vision*, Springer, 2020, pp. 402–419.

[12] R. Yang, M. Xu, Z. Wang, and T. Li, "Multi-frame quality enhancement for compressed video," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2018, pp. 6664–6673, ISBN: 978-1-5386-6420-9. DOI: 10.1109/CVPR.2018.00697. [Online]. Available: https://ieeexplore.ieee.org/document/8578795/.

[13] Q. Xing, Z. Guan, M. Xu, R. Yang, T. Liu, and Z. Wang, "Mfqe 2.0: A new approach for multi-frame quality enhancement on compressed video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 3, pp. 949–963, Mar. 2021, arXiv: 1902.09707, ISSN: 0162-8828, 2160-9292, 1939-3539. DOI: 10.1109/TPAMI.2019.2944806.

[14] J. Deng, L. Wang, S. Pu, and C. Zhuo, "Spatio-temporal deformable convolution for compressed video quality enhancement," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 0707, pp. 10 696–10 703, Apr. 2020, ISSN: 2374-3468. DOI: 10.1609/aaai.v34i07.6697.

[15] M. Zhao, Y. Xu, and S. Zhou, "Recursive fusion and deformable spatiotemporal attention for video compression artifact reduction," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021, pp. 5646–5654.

[16] Y. Xu, M. Zhao, J. Liu, *et al.*, "Boosting the performance of video compression artifact reduction with reference frame proposals and frequency domain information," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 213–222.

[17] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the seventh IEEE international conference on computer vision*, Ieee, vol. 2, 1999, pp. 1150–1157.

[18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[19] G. K. Wallace, "The jpeg still picture compression standard," *IEEE transactions on consumer electronics*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[20] C. Dong, Y. Deng, C. Change Loy, and X. Tang, "Compression artifacts reduction by a deep convolutional network," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 576–584.

[21] X. Liu, X. Wu, J. Zhou, and D. Zhao, "Data-driven sparsity-based restoration of jpeg-compressed images in dual transform-pixel domain," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5171–5178.

[22] X. Zhang, W. Yang, Y. Hu, and J. Liu, "Dmcnn: Dual-domain multi-scale convolutional neural network for compression artifacts removal," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, IEEE, 2018, pp. 390–394.

[23] B. Zheng, Y. Chen, X. Tian, F. Zhou, and X. Liu, "Implicit dual-domain convolutional network for robust color image compression artifact reduction," *IEEE Transactions on Circuits and Systems for Video Technology*, 2019.

[24] Z. Wang, D. Liu, S. Chang, Q. Ling, Y. Yang, and T. S. Huang, "D3: Deep dual-domain based fast restoration of jpeg-compressed images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2764–2772.

[25] P. Liu, H. Zhang, K. Zhang, L. Lin, and W. Zuo, "Multi-level wavelet-cnn for image restoration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 773–782.

[26] M. Ehrlich, L. Davis, S.-N. Lim, and A. Shrivastava, "Quantization guided jpeg artifact correction," *Proceedings of the European Conference on Computer Vision*, 2020.

[27] M. Ehrlich, L. Davis, S.-N. Lim, and A. Shrivastava, "Analyzing and mitigating jpeg compression defects in deep learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2357–2367.

[28] J. Jiang, K. Zhang, and R. Timofte, "Towards flexible blind jpeg artifacts removal," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 4997–5006.

[29] T. Xue, B. Chen, J. Wu, D. Wei, and W. T. Freeman, "Video enhancement with task-oriented flow," *International Journal of Computer Vision*, vol. 127, no. 8, pp. 1106–1125, Aug. 2019, arXiv: 1711.09078, ISSN: 0920-5691, 1573-1405. DOI: 10.1007/s11263-018-01144-2.

[30] X. Wang, K. C. Chan, K. Yu, C. Dong, and C. Change Loy, "Edvr: Video restoration with enhanced deformable convolutional networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[31] J. Dai, H. Qi, Y. Xiong, *et al.*, "Deformable convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 764–773.

[32] K. C. Chan, X. Wang, K. Yu, C. Dong, and C. C. Loy, "Basicvsr: The search for essential components in video super-resolution and beyond," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4947–4956.

[33] Y. Li, P. Jin, F. Yang, C. Liu, M.-H. Yang, and P. Milanfar, "Comisr: Compression-informed video super-resolution," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2021, pp. 2543–2552.

[34] J. Lin, Y. Cai, X. Hu, *et al.*, "Flow-guided sparse transformer for video deblurring," in *ICML*, 2022.

[35] J. Lin, X. Hu, Y. Cai, *et al.*, "Unsupervised flow-aligned sequence-to-sequence learning for video restoration," in *International Conference on Machine Learning*, PMLR, 2022, pp. 13 394–13 404.

[36] J. Xiao, X. Jiang, N. Zheng, *et al.*, "Online video super-resolution with convolutional kernel bypass grafts," *IEEE Transactions on Multimedia*, 2023.

[37] H. Lin, X. He, L. Qing, Q. Teng, and S. Yang, "Improved low-bitrate hevc video coding using deep learning based super-resolution and adaptive block patching," *IEEE Transactions on Multimedia*, vol. 21, no. 12, pp. 3010–3023, 2019.

[38] T. Wang, M. Chen, and H. Chao, "A novel deep learning-based method of improving coding efficiency from the decoder-end for hevc," in *2017 Data Compression Conference (DCC)*, IEEE, 2017, pp. 410–419.

[39] R. Yang, M. Xu, T. Liu, Z. Wang, and Z. Guan, "Enhancing quality for hevc compressed videos," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 7, pp. 2039–2054, 2018.

[40] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[41] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[42] Q. Ding, L. Shen, L. Yu, H. Yang, and M. Xu, "Patch-wise spatial-temporal quality enhancement for hevc compressed video," *IEEE Transactions on Image Processing*, vol. 30, pp. 6459–6472, 2021.

[43] S. Tomar, "Converting video formats with ffmpeg," *Linux Journal*, vol. 2006, no. 146, p. 10, 2006.

[44] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*, PMLR, 2017, pp. 214–223.

[45] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[46] M. Chu, Y. Xie, J. Mayer, L. Leal-Taixé, and N. Thuerey, "Learning temporal coherence via self-supervision for gan-based video generation," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 75–1, 2020.

[47] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[48] S. Bell, P. Upchurch, N. Snavely, and K. Bala, "Material recognition in the wild with the materials in context database," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3479–3487.

[49] J.-R. Ohm, G. J. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards—including high efficiency video coding (hevc)," *IEEE Transactions on circuits and systems for video technology*, vol. 22, no. 12, pp. 1669–1684, 2012.

[50] A. Paszke, S. Gross, F. Massa, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, pp. 8026–8037, 2019.

[51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[52] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.

[53] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[54] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595.

[55] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems*, vol. 30, 2017.

[56] H. Chen, B. He, H. Wang, Y. Ren, S.-N. Lim, and A. Shrivastava, "Nerv: Neural representations for videos," *arXiv preprint arXiv:2110.13903*, 2021.

[57] C.-Y. Wu, N. Singhal, and P. Krahenbuhl, "Video compression through image interpolation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 416–431.

[58] G. Lu, W. Ouyang, D. Xu, X. Zhang, C. Cai, and Z. Gao, "Dvc: An end-to-end deep video compression framework," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 006–11 015.

[59] J. Liu, S. Wang, W.-C. Ma, *et al.*, "Conditional entropy coding for efficient video compression," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*, Springer, 2020, pp. 453–468.

[60] A. Mercat, M. Viitanen, and J. Vanne, "Uvg dataset: 50/120fps 4k sequences for video codec analysis and development," in *Proceedings of the 11th ACM Multimedia Systems Conference*, 2020, pp. 297–302.