Deep Learning Techniques in Video Coding and Quality Analysis

Pankaj Topiwala, Madhu Krishnan, Wei Dai {pankajtva at gmail dot com}
FastVDO LLC
3097 Cortona Drive, Melbourne, FL, USA

ABSTRACT

Video coding is a powerful enabling technology for networked multimedia transmission and communication, that has been in constant improvement for decades. The upcoming VVC video codec, due in 2020, from the ITU|ISO/IEC standards committees, aims to achieve on the order of 1000:1 compression on high resolution and high dynamic range video, a stunning landmark. But the basic structure of codecs has remained largely unchanged over time, the gains obtained mainly through complexity increases. Moreover, video encoders have for decades used the same mean squared error, or sum of absolute differences, measure to optimize coding decisions. At the same time, the rapid rise of deep learning (DL) techniques poses the question: can DL fundamentally reshape how video is coded. While that question is highly complex, we first see a path for DL methods to make inroads into how video quality is measured. This in turn can also change how it is coded. In particular, we study a recently introduced video quality metric called VMAF and find ways to improve it further, which can lead to more powerful encoder designs that employ these measures in the coding decisions.

1 INTRODUCTION

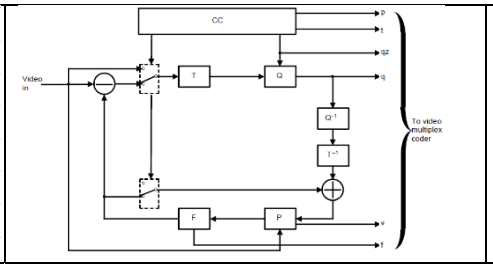
Lossy video compression is one of the most successful communications technologies that been developed in the past few decades. While lossless compression of diverse data types such as text, speech, images, and video all generally max out at under 3:1 compression, the lossy compression of multimedia data can achieve surprisingly high levels of compression, while still offering reconstruction qualities suitable for a variety of applications, most notably surveillance, communications, and consumer entertainment. While lossy coding of all media has made significant strides in recent times, the most remarkable progress has been in video. For convenience, we will summarize that development history (fig. 1).

Beginning in 1988, with the development of H.261 and then MPEG-1, the focus was on VCD (video compact disk), perhaps aimed at 25:1 compression. In 1994, H.262/MPEG-2 was focused on DVD, aimed at about 35:1 compression. In 2003, H.264/MPEG-4 AVC was focused on 1080p+ applications, and aimed at 20-200:1 compression. In 2013, H.265/MPEG-H HEVC was focused on 4K applications, and aimed at up to 500:1 compression. Finally, H.266/MPEG VVC, currently in development and due for completion in 2020, is focused on 4K/8K, High Dynamic Range (HDR) video, as well as spherical or 360-degree video, a novel application used in virtual reality. It squarely aims for up to **1000:1** compression, as the test data rates in the recent Call for Proposals (CfP) show [15]. For a review of these video coding developments, see [29-32]. In about **30 years** of development, we have increased the compression efficiency of video coding by a stunning **50:1**, so that video compressed in H.266 can earnestly aim to represent data at a rate that is one-tenth of one percent of the original video! So breathtaking is this achievement, that it was likely inconceivable to anyone when this journey began. But just as Moore's Law on compute density has marched on inexorably for decades, yet may soon be reaching its limits, so too in video compression we may be facing a similar story. While no one knows how much more efficiency can still be extracted from video, it is possible we are reaching some limits (remarkably, information theory has yet to set limits on lossy video coding efficiency).

What is also remarkable that the basic structure of the video codecs – a hybrid, motion-compensated residual coding design, first proposed in 1974 [36], with predictors, filters, transforms, quantizers, and entropy coders, has been maintained the entire time, adding only sophistication and complexity to each of its component parts. Thus, the massive coding efficiency gains have come mainly from two key factors: increasing video resolutions, and increasing operations per pixel, based on increasing compute densities available. What is further interesting is that all codec designs have essentially be structured to optimize for a single loss function – mean squared error, MSE (or just the sum of absolute differences, SAD). So much success has perhaps bred a reluctance to change the way business is done. But there is

finally a sense that to achieve further visual quality improvements, in and beyond H.266/VVC, we may perhaps benefit from improving our visual quality assessment measures as well. Thus, image quality metrics such as SSIM have been implemented in the decoder (though not yet into the encoder decisioning processing, for rate-distortion optimization, or RDO). For H.266/VVC development, a conditional weighted MSE function has been introduced (fig. 3), which is used for HDR and 360 video. In this paper, after reviewing some coding tools that have deep learning approaches, we focus on its impact on video quality assessment. We find some encouraging evidence that DL methods can be useful in this field.

Standard	Date	CR	Kep App
ITU/H.261	1988	20	vid conf
ISO/MPEG-1	1992	25	VCD
H.262/MPEG-2	1994	35	DVD, TV
H.264/AVC	2003	up to 200	generic
H.265/HEVC	2013	up to 500	generic
H.266/VVC	2020	up to 1000	generic



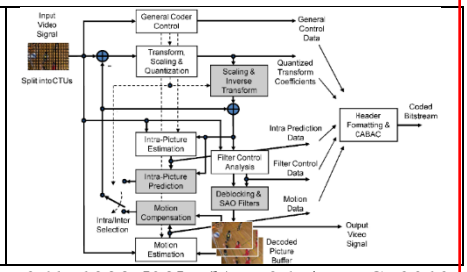


Figure 1. (a) Resume of video coding standards. Encoder block diagrams, (a) H.261, 1988 [38]; (b) H.265/HEVC, 2013 [39], copyright IEEE. All essential elements of the codec were already present in the H.261 design, including predictors, transforms, quantizers, filters, and entropy coders. These elements, scaled up, remain in use for the next gen VVC codec.

*Figure 2. FastVDO example extreme low bitrate coding, with 3 codecs (AVC/HEVC/VVC draft), on a 1080p50 sequence (Basketball Drive), with (a) objective metric Y-PSNR; (b) crop visuals at 0.8 Mb/s (that is 3000:1 compression!), using VVC, HEVC, and AVC (H.266/5/4) L to R. (a) Hor. Line suggests that as of now, 0.8 Mb/s VVC ~ 1.2 Mb/s HEVC ~ 3.2 Mb/s AVC (**VVC=1.33X HEVC = 4X AVC in coding efficiency**).*

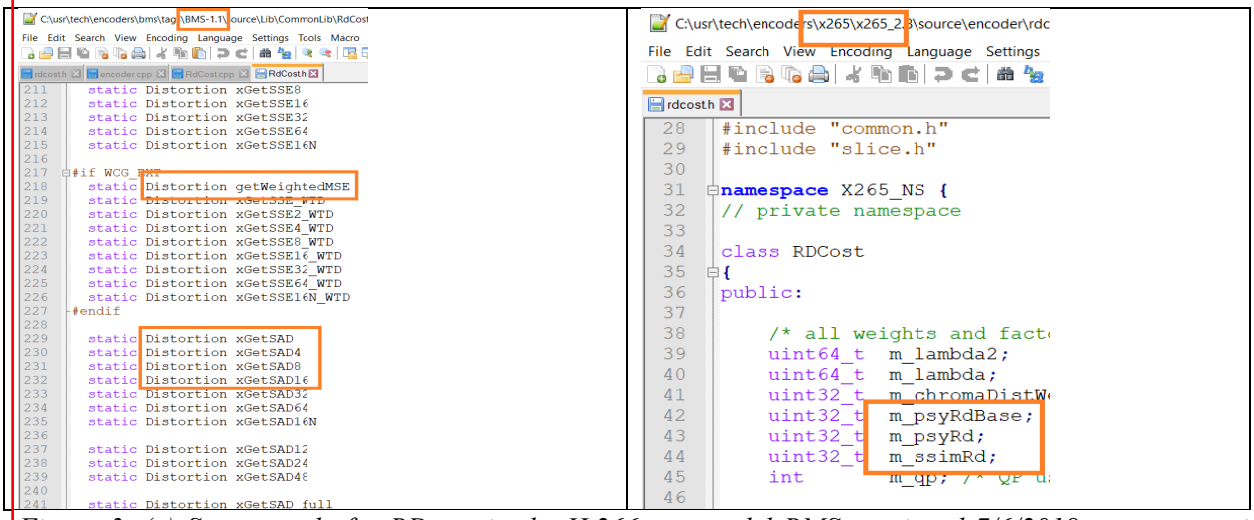


Figure 3. (a) Source code for RDcost in the H.266 test model BMS, retrieved 7/6/2018, supports weighted MSE (conditional), and SAD (default) as distortions. (b) Source code for RDcost in open source x265

(retrieved 7/6/2018), the most widely used H.265 encoder, supports SSIM, and a psyRD, though SAD is default. Models beyond SAD/PSNR are just beginning to make inroads into actual encoders in wide use.

	AOM AV1 (1.0)	**H.265/HEVC HM 16.18**	**H.266/VVC BMS 1.0 (draft)**
Block Structure	10-way split (AV1) Largest block size 128x128 (superblock).	Quadtree CTU size up to 64x64	(QTBT) + Ternary Tree (TT) CTU size up to 256x256
Intra Prediction	56 intra directional modes 5 non-directional modes Recursive filt. based intra prediction Chroma from Luma Color palette based intra prediction Intra block copy	35 intra prediction modes.	65 intra prediction modes with improved intra mode coding Cross-component linear model (CCLM) prediction
Inter prediction	Single and compound prediction (similar to P and B) (VP9) Extended reference frames (3 to 7) Dynamic spatial and temporal motion vector referencing Overlapped block motion compensation Warped motion compensation Advanced compound prediction	Hierarchical weighted prediction (P, B frames) PU level motion vector prediction Motion vector difference 1/4 pel MV accuracy Block motion comp. Translation motion prediction	Hierarchical weighted prediction (P, B frames) Sub-CU based motion vector prediction Adaptive motion vector precision Affine motion prediction Decoder-side motion vector refinement
Transform	Transform blocks 4x4 up to 64x64 DCT, ADST (VP9), Flipped ADST, DST-I	Transform block size 8x8, 16x16, 32x32 DCT-II and DST-VII	Transform block sizes 4x4 up to 64x64 Adaptive multiple core transforms Mode dependent non-separable secondary transforms (4x4)
Loop filter	Constrained directional enh. filter Loop restoration filters Frame super resolution Film grain synthesis	Deblocking filter, SAO	Deblocking filter, SAO, Adaptive loop filter
Entropy Coding	Multi-symbol entropy coding Level map coefficient coding	CABAC	Modified CABAC (with Context modelling for transform coefficient levels)

Table 1. Tools Comparison for AOM AV1, HEVC HM, JVET VVC BMS1 (Draft).VVC is a standard currently in development. For many of the listed tools, there is a deep learning variant (though currently of higher complexity).

2 A PRIMER ON DEEP LEARNING METHODS

Neural networks (NNs) seem promising for many applications, but need extensive training to be useful. A breakthrough happened in 2006, when led by Canadian researchers (with G. Hinton at the center), deep networks were introduced which learned all stages through supervised training – no hand-tuning needed [LeCunHinton]. And 2014 opened the floodgates, when a Google trained network called GoogleNet (later called Inception), inspired by a previously winning effort by G. Hinton and company in 2012, finally approached human-level performance in the key task of image recognition, with a 6.7% error rate – a landmark feat [LeCunHinton]. A 2015 network by Microsoft [ResNet], with 152 layers, brought the error rate further down to a stunning 3.6%. And a combined Inception-ResNet model by Google in 2015 [Incv4] brought the error rate to 3.08%, exceeding humans on the same test data – stunning the world. A vast array of deep learning methods have appeared in just the last five years, most still having roots that trace back to the Inception model. The success of the Inception-ResNet model in image classification has been so convincing, that it has the feel of a definitive solution. Convolution layers automatically find the most useful features directly and hierarchically from the data itself, while the final, fully connected layer with a softmax activation categorizes the images into discrete classes. Differentiability assumptions on functions and the chain rule allow backpropagation training, where nodes are reweighted according to whether they contributed to (in)correct outcome, and repeated forward/backward propagation finds weights that work well. This still works when ReLU activation is used (only one non-differentiable point), and

ReLU has the advantage that its derivative is 1 for positive x, easing the well-known vanishing gradient problem when training a large system by gradient descent. The ResNet model, which adds interlayer identity maps (f(x)=x) in network architectures, provides further help, since their derivatives are also 1. We will use the following key references in our development. First, the book *Deep Learning*, by I. Goodfellow, Y. Bengio, and A. Courville, MIT Press, 2016 [DLbook], will serve as a primary source for the lay of the land. Another primary source will be the book-length article, "Learning Deep Architectures for AI," by Yoshua Bengio, 2009 [YBbook]. Finally, the expository 2015 paper, "Deep Learning," by Y. LeCun, Y. Bengio, and G. Hinton [LeCunHinton] is a powerful single source. And for computational complexity analysis, we rely on [VSze].

For our practical purposes, CNNs are key to image recognition tasks, exploiting spatial structures (e.g., edges, texture, color), while recurrent neural networks (RNNs) can take on tasks that involve temporal processing (such as with natural language: speech, text). These network types can also be combined, in sequence, for example to create text annotations for images and video. For segmentation in image/video data, some combination of these two architectural types is also directly merited, but more complicated than say for image annotation, where the information itself is static in an image, only the annotation requires natural language, which is not static but involves temporal processing. But in video data, the information itself has both spatial and temporal dimensions. Despite the success of the image-based methods mentioned so far, to date there has been limited work in truly video-centric processing for visual recognition tasks. We do note that the 2017 Imagenet Large Scale Visual Recognition Contest (ILSVRC 2017) explicitly had a category for object detection in video sequences [ILSVRC-vid], which we explore below. For starters, CNNs can be used for accurately recognizing digits, such as license plates and even handwritten digits such as in the MNIST database (over 99% accuracy), a test we repeated. With RNN, we've generated fake Shakespeare as well. More surprising still is that we have used CNNs backwards to deep dream, creating dreamlike images starting from captured ones, by flowing slowly in output parameter space away from a given point (fig. 4). In fact, deep NNs can not only be representative of source data, but even auto-encode – they can generate (decode) originals, or fully realistic fakes, analogous to image/video compression, where an encoder creates a sparse (but faithful) representation, and a decoder creates that reconstruction. This is also the start of Generative Adversarial Networks or GANs [DLbook] – a contest of two NNs, one creates fakes, the other judges their realism. An autoencoder NN is a sequence of nonlinear mappings, which represent source data, and a decoder segment back to the original domain, where the Lagrangian cost is minimized for fidelity of representation, but Z is lower dimensional. Video compression has now passed a staggering 1000:1 data reduction (Gb/s to Mb/s), while producing high-quality reconstructions, testing credibility. Similar reductions can be achievable in NN auto-encoders as well, and NNs can imitate all aspects of compression (e.g., predictors, filters, transforms, quantizers, entropy coders). It remains to be seen if NN-based (or assisted) frameworks for video compression can reach the state-of-the-art.

$$X \xrightarrow{f} Y \xrightarrow{g} \dots \xrightarrow{h} Z \xrightarrow{i} \dots \xrightarrow{m} X, \qquad combined\ X \xrightarrow{F} X.$$

$$L(x,z) = argmin\|x - F(x)\|^2, but\ Z\ is\ chosen\ lower\ dimensional$$

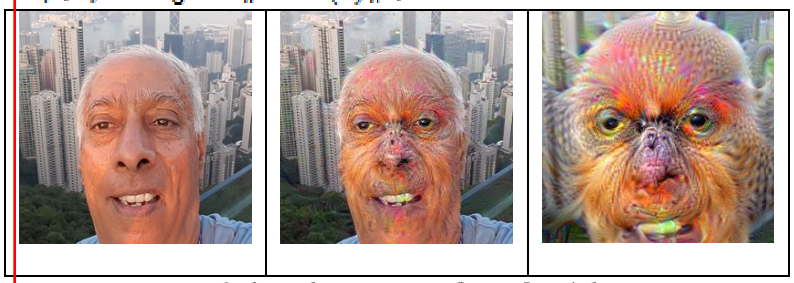


Figure 4. FastVDO deep dream, using [Incv4], w/ the presenter's image in Hong Kong as seed, a stark model of real dreams, and a visual indicator of parallels between deep NNs and animal visual cognition.

Reinforcement Learning. A powerful new idea in the field of AI, reinforcement learning (RL), allows AI systems to learn on their own, entirely from experience, by learning a policy of action that maximizes a reward. This works well in

playing games (e.g., AlphaGo, Google's machine that beat the human master) [deepmindRL] where one learns successful moves simply by playing millions of games, and rewards (win or lose) are used to modify the approach to playing. Just as backprop supplies a neural weight gradient according to its partial contribution to success, RL supplies a policy gradient, say in a strategy, by its partial value in winning. Since policy choices are often discrete, but we work with smooth functions, we assign probabilities for the various policy choices, and adjust these. To play a game, we choose values according to the probabilities. War games have long been used precisely for the same reason: to find winning strategies by fighting in sims, rather than thinking them up. In a sense, RL generalizes gradient-descent learning by backprop; if we view the action of NNs with given weights for performing an action (e.g., classification) as policies, then policy gradients can reduce to actual gradients. But that would only repeat existing DL methods, whereas the RL framework is far more powerful, and can even be used to design DL architectures for given problems, called AutoML [aml-aml3], with leading performance (fig. 5).

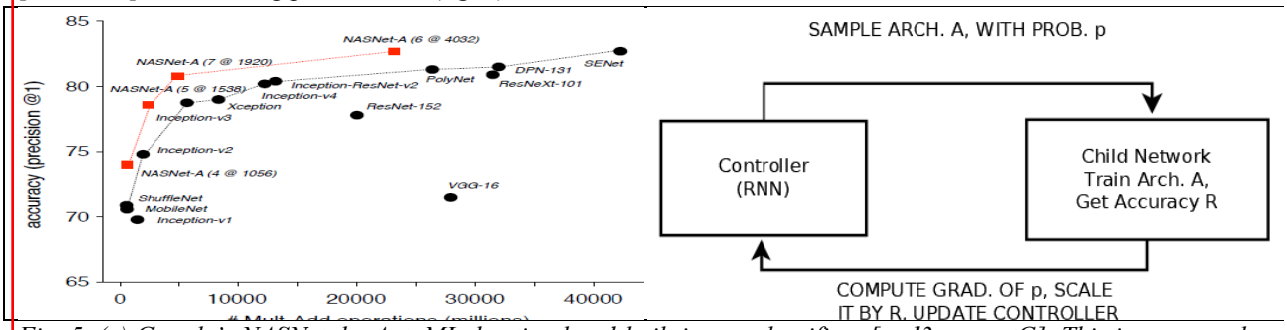


Fig. 5. (a) Google's NASNet, by AutoML, beating hand-built image classifiers [aml3, nasnetG]. This is a game-changer (code public). Since machine vision is built on imageClassifiers, this is a 2nd revolution. These results are for the top-1 classification; similar results also hold for top-5, what is often reported. (b) *The neural arch. search process (NAS) [neuArchSearchG], which can create good NNs, like NASNet. Graphic (a) is from cited Google Blog.*

The Neural Architecture Search process [neuArchSearchG, nasnetG] works as follows. A controller NN (an RNN) samples an architecture A with probability p (in reality, each component has a probability), thus creating an instance child network, whose performance on a given task, say image classification, results in accuracy R. In the next iteration, the probability p is adjusted by scaling its gradients by control parameters by the accuracy R, more accurate models increase in probability. The recurrent neural network controller learns over time which architectural components were the most successful. Since it is designed to increase the reward (success) it eventually develops a powerful, successful, and compact design that work on the initial test problem. Finally, the test set can be changed, and the process restarted. Surprisingly, what Google has found is that a network that works well on the tiny CIFAR-10 dataset (60k images of size 32x32), upon replication into a number of layers, also works well on the massive ImageNet dataset (14M large images, and growing). While this process is not yet fully automated (e.g., guessing a generic structure as starting point), it is a strong indication of the trend. Indeed, if component tools in a NN are listed, a system can test all combinations from scratch, and be fully automatic.

3 RESUME OF APPLICATIONS OF DEEP LEARNING TO VIDEO CODING

We restate that a NN can indeed be deployed to perform all the same functional tasks that are performed in the design of a modern video codec, such as predictors, filters, transforms, quantizers, and even entropy coders. For example, predictors such as motion estimators can be designed using NNs [nnme], though highly complex, while intra-predictors are quite similar. As for filtering, numerous papers have addressed this topic; we cite [cnnf]. DCT-like transforms can be constructed with CNNs [nndct]. For quantization, since this is a simple classification problem, this has been known for decades, and even vector quantizers can be represented [nnvq]. Indeed, a full end-to-end image codec can be built by using neural networks [vrRnn], which moreover allows to adjust the data rate. We wish now to focus on video coding. One important observation is that, if considered as a restoration problem post compression, we note that we have both

distorted and original content at hand. So a NN can be trained to modify the distorted imagery (block-by-block) in the direction of the original content. This was initially successfully done in the context of JPEG compression [arcnn], and then later in the context of intra in video [vrcnn]. In particular, this concept can be viewed in two contexts: as a post-loop filter [cnnf] in the decoder, and as a post-decoding restoration processing [dcad, mfqe]; see the many references cited in [mfqe]. One interesting observation is that while these methods provide a modest impact on image quality (about 0.5 dB gain), that difference is in fact visible in close-up views (where it significantly reduces artifacts such as blurring, ringing…).

Meanwhile, at least two papers, both from April, 2018, have directly addressed the problem of building a full video codec from DL alone [nnfullvid, voxelcnn]. Both report performance comparable to H.264/AVC, our 2003 standard. Meanwhile, our coding algorithms have seriously improved (see fig. 2). Nevertheless, the fact that this can be done at all, with reasonable success, is stunning, and there is little doubt this nascent field will advance rapidly in the coming years.

4 VIDEO QUALITY ANALYSIS – A RESEARCH PROGRAM

While DL can fill in for many of the tools used in video compression (or even potentially the entire chain), we focus now on a specific aspect that seems especially promising – measuring video quality. Digital video services, encompassing everything from broadcast TV and streaming to video chat, is a massive worldwide industry estimated to reach $120B by 2022 [3]. Critical to the success of this industry is providing the highest quality video afforded by receivers and channel capacities. But measuring that video quality, called video quality assessment (VQA), remains a fine art, best done by subjective testing, impossible when you have millions of streams like Netflix, YouTube. Instead, this industry has long used an objective metric called peak-signal-to-noise ratio (PSNR), developed more for computational ease than reliability. Recently, new objective metrics such as Structural SIMiliary (SSIM) and Visual Information Fidelity (VIF) have been introduced, which have made some inroads. But a fundamental need remains to have an objective metric which is both easy to compute and has predictive power for subjective quality with very high reliability. The availability of such a reliable (and computable) metric would be a bonanza. Our research program: construct it by combining image-based measures with motion-based measures, fused if needed through machine learning methods. Netflix has proposed just such a fused metric called VMAF (and a simple linear SVM fusion method), which is apparently the state-of-the-art for its application (streaming by http/tcp) today. While image-based measures are well-represented in it, we find the motion-based measure used inadequate, and see a path to improving it. Our intuition is that improving the motion representation will help.

In fact, motion is key to understanding vision, and thus video (a feature film is a **motion picture**). Yet for over 30 years, full reference video quality analysis has principally relied on image quality analysis, applied per frame, then averaging over frames, ignoring the fundamental role of motion in video. This is clearly incomplete. “Gordon Lynn Walls, a comparative anatomist, observed, “If asked what aspect of vision means the most to them, a watchmaker may answer ‘acuity,’ a night flier ‘sensitivity,’ and an artist ‘color.’ But to the animals which invented the vertebrate eye, and hold the patents on most of the features of the human model, the visual registration of ***movement*** was of the greatest importance” (Walls, 1942)” [1]. In a predator/prey world, it is a vital survival sense, but also a key to video quality. Video quality analysis is itself an industry on its own, with standardization efforts running for over two decades in the Video Quality Experts Group (VQEG) of the International Telecommunications Union (ITU, a UN-chartered organization based in Geneva) [21], a sister organization to the Video Coding Experts Group (VCEG) of ITU that we have served in for ~20 years. US govt agencies such as the National Telecomm. Information Admin. (NTIA), and the National Institute for Standards and Technology (NIST), both of the Dept. of Commerce. In fact, a leading VQA, called VQM-VFD, was developed at NTIA, and is a VQEG standard [10]. While VQA research has been going on for decades, the video coding industry, often working in nearby ITU facilities in Geneva, has paid little attention. But now, these groups are starting to pay more attention to these VQA efforts, incorporating SSIM/MS-SSIM as well as PSNR into post-compression assessment calculations (though not yet in the encoder decision making process as of July, 2018). The old workhorse, PSNR, religiously used for decades, and central in encoding decisions, may be running out of steam.

“This ability to detect motion is key for animals, allowing them to detect the presence of predators,” according to Prof. Daniel Kerschensteiner, MD, a discoverer of the motion detection mechanism in the eye [2]. Our understanding of human vision begins with the 1982 book, *Vision*, by David Marr [13]. In it, he pursued edge representations, multiscale

processing (just before wavelets and multiresolution analysis [35,36]), and angle-dependent Gabor functions (modulated Gaussians) as basic functions for visual representation. These ideas remain valid today, even if the details differ. In fact, time-frequency and time-scale variations (i.e., translations, modulations, and dilations) are the only invariances of the Gaussian minimizers of Heisenberg's Uncertainty Principle, a mathematical theorem with deep implications in physics [14]. Marr also observed its import in image processing. Thus, Gabor functions (wave packets: translated, modulated Gaussians) are models of elementary particles, as well as elementary packets of signals, while time-scale translates lead to wavelets. What signal processing can learn from physics is astonishing, not least that as in physics, Lagrange calculus plays a central role, in both video compression and deep learning. But Marr's ideas also play an important role in formulating video quality assessment (VQA). For down-to-earth video quality applications in the field of video compression, we have a reference video, consisting of uncompressed video and considered pristine, and various compressed versions for use in applications such as streaming or broadcast, which are distorted. Every one of the millions of streams of video made available by the likes of Netflix, Amazon, YouTube, Hulu, Baidu, and many others around the world is judged on its quality prior to serving/casting. Yet for this task, the industry has mainly used image quality metrics, in fact, for the most part just mean-squared error (MSE), and its relative PSNR, ignoring motion. In large part this is due to complexity considerations. In fact, a usable VQA must meet two stringent requirements: (a) provide high predictive power for subjective visual quality, and (b) be computationally efficient. While both are challenging, it is in this last part that many VQA metrics developed over the years really fall short. Somehow, we need to improve metrics yet make them more computable.

If successful, we envision at least three separate, increasingly larger but more demanding applications of VQA. First, VQA can be used in stream selection (send the best quality video), which an elementary, often offline application. Second, VQA can be used in video restoration at the receiver (restore for best visual quality). This could be combined with deep learning to train blocks of video frames on the original video, which can provide effective restoration in compressed and other distorted videos [18-20]. This is a powerful application, especially offline. Finally, it could be used at the encoder to decide how best to encode a video with a given codec (code for true visual quality). While stream selection (at server) and restoration (at receiver) *can* require real-time performance, and thus pose complexity constraints, it is the encoding application that is by far the most challenging; we will focus on this application. The issue is that all modern encoders rely on using rate-distortion optimization (RDO) [9,14] to make decisions, based on an interplay between distortion D, and the rate R, to optimize the Lagrangian (where λ is a constant called a Lagrange multiplier):

(1) $L = D + \lambda R = \sum_i D_i + \lambda R_i\,; \quad \delta L = 0 \Rightarrow \delta L_i = 0 \Rightarrow \lambda = -\frac{D_i}{R_i}, a\ constant.$

Thus, given any number of independent parameters to optimize (e.g., various pixel quantizers), these are jointly optimized when the slopes of negative distortion over rate are all equal [14]. Now, in coding a 4K video, a modern encoder such as H.265 must make millions of RDO decisions per second, on everything from mode selection and motion estimation, to quantization and filtering. Since many video applications require real-time encoding (e.g., live events), usually in hardware, this puts severe constraints on how RDO is actually computed. The rate R is straightforward: how many bits it takes to encode the data (though even this is estimated to save cycles, not computed). But what to use for the distortion D, comparing a coded MxN block B to the reference version, is more open. Typically, the simple mean squared error (MSE) or L2-norm is used to represent the block-based **spatial error** $E_{k,spat}$. In fact, this is further simplified to just the Sum of Absolute Differences (SAD, or L1-norm), mainly to avoid squaring!

(2) $E_{k,spat} = SAD = \sum_{i,j=1}^{M,N} |B_{ref,i,j} - B_{coded,i,j}| = \left|\left|F_{ref} - F_{coded}\right|\right|, the\ L1\ norm.$

(3) $E_{k,spat} = MSE = 1/MN \sum_{i,j=1}^{M,N} |B_{ref,i,j} - B_{coded,i,j}|\text{^}2; \qquad PSNR = 10 * \log(255\text{^}2/MSE)).$

For decades, coded videos have been graded by PSNR values to the point of intimate familiarity (e.g., a 38+dB PSNR for 8-bit video is entertainment quality), while the internals of the video codecs actually use SAD to optimize its decisions. Given these computational bottlenecks, one can understand the reluctance to incorporate more sophisticated yet untrusted VQA metrics that have been proposed for years. Enter Netflix, one of the world's largest video encoding companies, accounting for 36% of all Internet traffic in the US in 2015 [8] and growing, and Amazon Web Services (AWS), with a million+ server-farm, aided by Graphical Processing Units (GPUs) and Field Programmable Gate Arrays

(FPGAs). Netflix does all its video crunching offline on AWS, and has sizable headroom for more computations. Suddenly, VQA is mainstream. One observation we make is that in applying a VQA at an encoder, while restrictive in complexity, does have the benefit of having at hand the actual motion estimation used in the encoder; more later (see Detailed Innovations). As mentioned, up till now all VQA metrics in wide use were really Image QAs (IQAs). A recent metric proposed in 2016 by Netflix finally aims to remedy this, introducing a video multi-algorithm fusion (VMAF) metric [4,5]; it fuses two image quality metrics, and a feature called "motion," into a metric using a support vector machine (SVM) regressor. The two image quality metrics are the well-known Visual Information Fidelity (VIF) [6], and the Detail Loss Metric (DLM) [7]. Both VIF and DLM are based on extracting features from images in the wavelet transform domain for multiscale processing (**Marr-inspired!**). In fact, both VIF and DLM have features at several different scales, but in the end **4 VIF scaled features**, and **1 of DLM**, are used in the VMAF0.6.1 model. As these work well, we keep them. For now, we focus on the "motion" feature used in VMAF, which is not a metric at all, nor even compares a distorted video with a reference (like the other two true metrics), but is simply the sum of absolute difference (SAD, or L1-norm) of the frame differences in the reference video. Surprisingly, even this allows them to claim the state of the art [4], in terms of power to predict subjective quality by human observers. But realizing this mission is incomplete, Netflix has generously released its software and dataset to encourage more work [5]. We will employ this software throughout (and cite Netflix results).

We observe more carefully that the "motion" feature used in VMAF [5] makes no use of the distorted video at all. Instead, it uses the following. If an original (uncompressed) video sequence is a set of frames $\{F_k\}$, k=0,…,K, VMAF uses the Sum of Absolute Frame Difference (SAFD) as a motion feature (Netflix calls this Mean of Co-located Pixel Difference), where ||*|| is the L1-norm. We will simplify refer to this Netflix feature as "**M**", for motion.

(4) $SAFD = \sum_{k=1}^{K} ||F_k - F_{k-1}||$. (Actually, it uses $\sum_{k=1}^{K-1} \min\{||F_k - F_{k-1}||, ||F_{k+1} - F_k||\}$.)

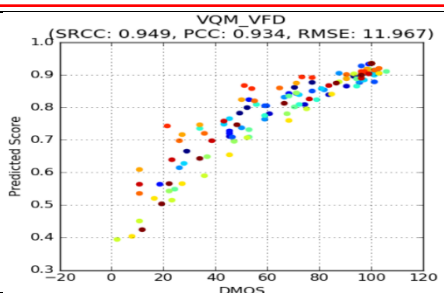
While this is informative about how much motion is in the video (and thus in part how difficult it may be to compress), it sheds no light on the quality of the motion in a compressed reconstructed (distorted) stream, which is the point of the VQA. Nevertheless, as it does carry motion information, it manages to achieve a slight gain over other competitive systems in predicting visual quality especially when combined with another leading approach: VQM-VFD [10,11], on a Netflix dataset [4], see figure 5. Here SRCC and PCC are the Spearman and Pearson Correlation Coefficients, and RMSE is Root MSE. The VQM-VFD itself, designed in 2011, is a complicated metric, using 8 hand-tuned spatio-temporal features derived from blocks of video frames, over a short period of time, in this instance, 0.2s. It is a classic study on how things were done prior to the deep learning revolution in 2012 [12], although oddly, it also uses a simple 2-layer neural network to provide its mapping. **We aim to improve on the quality of motion representation in VMAF**. Specifically, for original video frames $\{F_k\}$, k=0,…,K, and distorted video frames $\{G_k\}$, k=0,…,K, since the frame difference precisely corresponds to motion (all changes to pixels), we develop temporal motion based metrics using the difference of frame differences (key innovation/simplification). Let's call this FastVDO feature "**DM**" for differential motion.

(5) $E_{k,temp} = ||(F_k - F_{k-1}) - (G_k - G_{k-1})||$, *with the L1-norm (but can be L2,or Lp, etc).*

This is zero precisely when the motion information matches between the original and distorted videos. In combination with purely image-based measures, this can lead to a mathematical metric. We also develop several variants of this concept.

VQM-VFD, 2011. Figure 6, from the VMAF blog show that VQM-VFD [10,11], developed by the National Telecom. Information Admin. of US Dept. of Commerce [23], is one of the most effective measures of video quality available, a feat unlikely without incorporating temporal information. Indeed, it uses **8 hand-tuned features**, covering both spatial and temporal information (which it calls SI and TI, respectively), and was designed specifically to catch channel errors such as frame delay (VFD=variable frame delay). Now in the age of deep learning, hand-tuned features are quickly being replaced by machined learned features, which are not explicitly extracted, but learned and stored in network weights and biases. If needed, we can pursue improving even this metric using deep learning. For now, we elaborate on the one significant temporal feature used in it: TI_Gain. It computes functions using spatio-temporal (ST) blocks.

Metric	SRCC	PCC	RMSE
PSNR	0.746	0.725	24.577
SSIM	0.603	0.417	40.686
FastSSIM	0.685	0.605	31.233
PSNR-HVS	0.845	0.839	18.537
VQM-VFD	0.949	0.934	11.967
VMAF 0.3.1	0.953	0.963	9.277



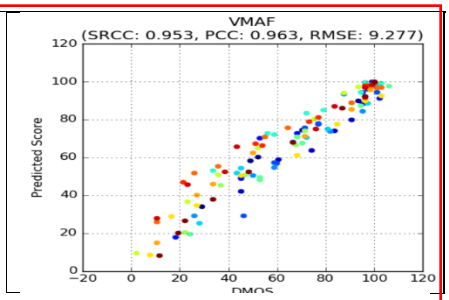


Figure 6. Netflix performance comparison of VMAF, VQM-VFD, and other metrics, vs Differential Mean Opinion Score (subjective testing), from online at [5], in (a) table, (b) graphs (color dots are per seq. True and Predicted Scores). SRCC and PCC are the Spearman and Pearson Corr. Coefficients, respectively, while RMSE is root mean square error. While (a) and (b) indicate that VQM-VFD performs very well, [5] indicates VMAF is both better and faster. Note that Netflix streams videos by http over TCP, so packet loss is not an issue. VMAF was a breakthrough, in both performance and speed. (b) is from cited Netflix blog. Compare fig. 10.

(6) $fTI = rms\big(Y(i,j,t) - Y(i,j,t-1)\big)$; $pTIG = \max\left[log10\left(\frac{fTIorig}{fTIproc}\right), 0\right]$; functions of ST blocks.

$$TI_{Gain} = STabove95\%tail(pTIG).$$

Here, fTI is the RMS of the frame difference at time t (motion) of the luminance component Y of a spatio-temporal (ST) block of video, where the spatial extent is set at 0.4 degrees in viewing angle, and the temporal extent is 0.2s of video (at 30fps, this is 6 frames). pTIG compares the motion energy in the original and processed (distorted) videos, in a manner analogous to PSNR, but clipped at 0. Finally, T_Gain is computed as the difference between (a) the average of the 95th to 100th percentile values, and (b) the 95th percentile value, of pTIG. Without analyzing this further, we note immediately that (a) it performs fairly well (better on jittery data); (b) it is complicated. Our mission to find something both more effective and more computable. As mentioned, we can employ deep learning on the whole problem, and then extract a functional model. But we return now to VMAF, and a recent update in April, 2018 from the Univ. of Texas called ST-VMAF [16].

ST-VMAF, April, 2018. ST-VMAF [16] builds on the VMAF metric of Netflix, but recognizing its weakness in representing motion, specifically adds spatio-temporal features. "VMAF … does not fully exploit temporal video quality measurements which are relevant to temporal video distortions." [16, Abstract]. **We fully agree**, and zero-in on a key temporal feature used in it. First, ST-VMAF is tested on a variety of datasets, and gives notable gains. While strong performance in terms of high correlation coefficients to subjective scores is always desirable, we have two qualifiers: (1) Most of the datasets have network-induced errors in the distorted videos (e.g., timing error, TE), so that this is addressing problems not met by HTTP/TCP streamers like Netflix (except to mobile devices). (2) It is also computationally complex, and difficult to implement into an encoder for actual decision processing. Compare this and equation (6) to our equation (5), which is computationally simpler and captures true motion error. Our results on the NFLX dataset are in fig. 10.

Database	# Seqs	Resolution	Distortion
Live VQA	150	768x432	Codecs, TE
Live Mobile	160	720p	Codecs, TE, RA
CSIQ-VQA	216	832-480	Codecs, TE, AWN
VMAF+	290	1080p	Codecs, Scaling
NFLX	300	1080p	Codecs, Scaling
SHVC	64	1080p	Codecs
VQEG HD3	135	1080p	Codecs, TE
EPFL	144	704x576, 1/4	Codecs, TE
USC-JND	3520	VGA-1080p	Codecs
Live-NFLX	112	1080p	QoE
Live-HTTP	15	720p	QoE

Metric	NFLX data
PSNR	0.705
PSNR-hvs	0.819
SSIM	0.788
MS-SSIM	0.741
ST-RRED	0.764
SpEED-QA	0.781
ST-MAD	0.768
VQM-VFD	0.931
VMAF 0.6.1	0.928
ST-VMAF	0.927
E-VMAF	0.93

Figure 7. Some informative tables from [16]. (a) various available databases for work in VQA, with citations in [16], importantly highlighting the type of distortion imposed. [16] develops some variants of VMAF (called ST-VMAF, E-VMAF), which show strong results on the various databases, which have transmission errors (TE), additive white noise (AWN), rate adaptation (RA), and RA and/or rebuffering (QoE). The NFLX database is one of the few with basically only coding artifacts. (b) shows the SRCC between predicted and subjectively rated scores. We note that on NFLX, VQM-VFD does slightly better than ST-, E-VMAF. Note updated VMAF version (0.6.1) is used.

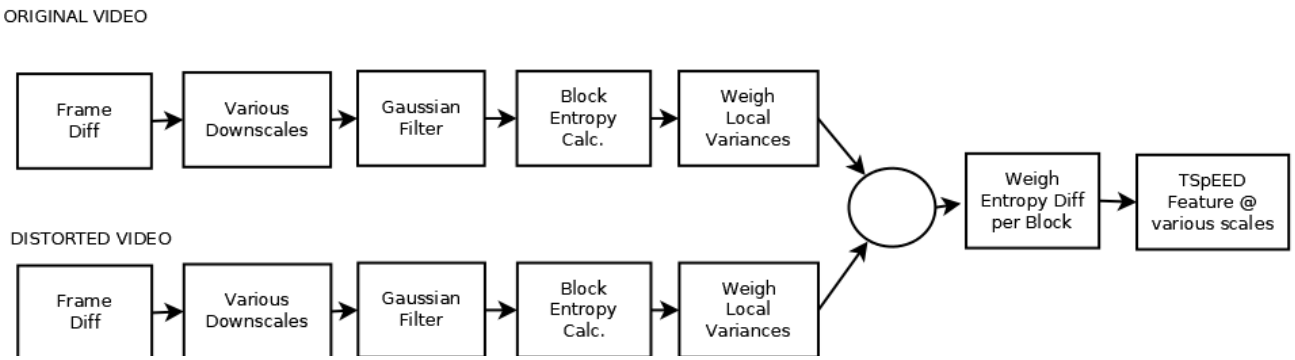
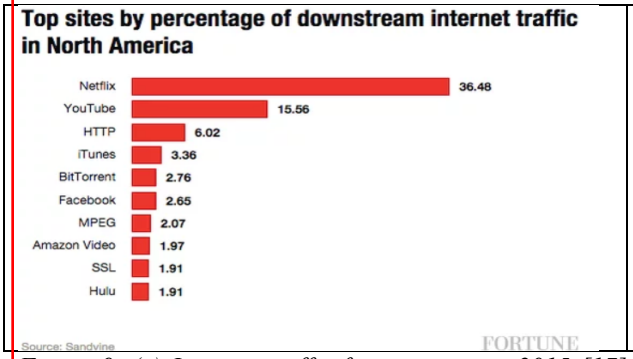


*Figure 8. A temporal feature used in ST-VMAF [16], called T-SpEED. It works at various scales (**Marr-inspired**) requiring downsampling, does Gaussian filtering, subtracts local means, calculates local entropies, modified by local weights, then finally compares reference to distorted videos, and then does averaging over blocks. So it seems computationally complex. See fig. 9, where it appears to be ~5X more complex than VMAF. The variant E-VMAF is similarly complex.*



Metric	Sec/Frame
VMAF	0.3
E-VMAF	1
ST-VMAF	1.13
ST-RRED	1.14
VQM-VFD	2.5
ST-MAD	10

Figure 9. (a) Internet traffic fractions, circa 2015 [17]. Since then, Netflix has grown several-fold. As of May, 2018, Netflix has surpassed Disney as the most valuable media company, and is watched more than cable or broadcast TV. (b) Relative complexity of leading VQA, as measured on a modern CPU, estimated from [16], for computing on 1080p data.

VMAF is the most computable, while ST-VMAF and its variant E-VMAF take more a second or more per frame on 1080p, let alone 4K/8K. That's a valid consideration, especially for encoder optimization applications.

Detailed Innovations and Research Vision

Innovation 1. Since the temporal frame difference represents motion, additionally consider the difference of frame differences between the reference (original) and distorted videos, $\{F_k\}$, $\{G_k\}$, k=0,…,K, respectively:

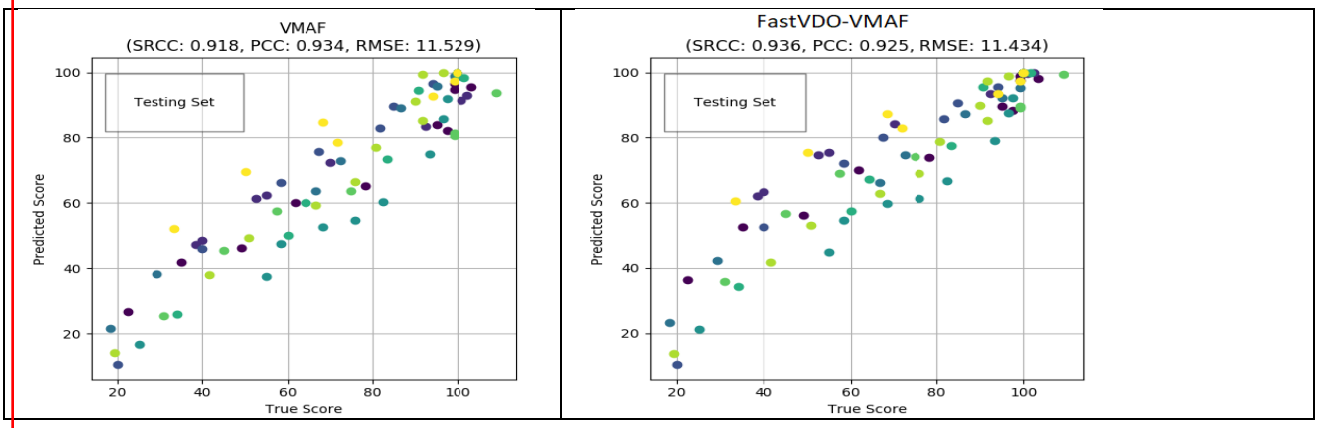
(5) $E_{k,temp} = \|(F_k - F_{k-1}) - (G_k - G_{k-1})\|$, the L1-norm (but can also be L2, Lp, Entropy, etc).

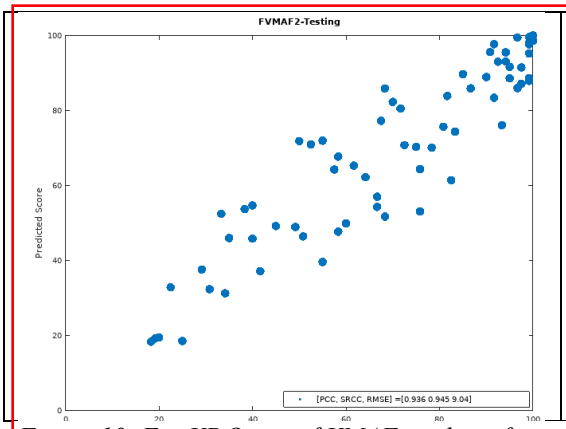
This represents the mismatch between the true and distorted motion, and thus is the **temporal error Differential Motion**. Metrics can be developed by taking any of L1, L2, or Entropy. This is to be combined with spatial error, linearly at first.

Innovation 2. In a video encoder, consider this difference now at the block level, again taking L1, L2, etc. For the distortion measure D in doing RDO (equation 1), set as below. With L2, this is a metric. Set the distortion as a linear combination:

(7) $D = aE_{k,spat} + bE_{k,temp}$, a + b =1, a, b > 0.

Innovation 3. In incorporating these models into VMAF, initially keeping VIF and DLM for spatial error, and using linear, then explicitly functional, and finally to DL models to optimally (nonlinearly) combine these measures into a powerful VQA. Moreover, **use complex VQAs judiciously**, i.e. in using RDO to do motion estimation, first reduce the motion search to the top few candidates with standard VQAs, and then select the best candidate with a more advanced DL-fused spatio-temporal error. Several variants of such approaches are also possible, and will be reported on elsewhere. But fig. 10 does show promising results for applying a very simple functional model in the training phase, computable as a look-up table, in the VMAF construction. This very elementary model, which remains highly computable (we will clock speeds elsewhere) already achieves good results, while a using a more powerful DL-model requires more data for training. These results are of course preliminary, and further testing with a larger database of training/testing is now warranted. We intend to test pure coding artifacts, and artifacts from channel errors, separately. HTTP/TCP streamers (who also use buffering) only suffer coding artifacts, while streaming to mobile devices over mobile (not Wi-Fi) networks may suffer further channel-induced errors. Both are important problems, but we intend to first focus on the HTTP streaming problem.





Code	ALG	SRCC	PCC	RMSE
Python	VMAF0.6.1	0.918	0.934	11.529
Python	FVMAF1	0.936	0.925	11.434
Octave	FVMAF1	0.943	0.93	9.872
Octave	FVMAF2	0.945	0.936	9.04

Figure 10. FastVDO test of VMAF, with performance on a Netflix database, where half the data is set as training, half testing, and utilizing the Netflix VMAF software [5]. Our results differ slightly from Netflix because we retrain VMAF (and our metrics) on a portion of the NFLX data, and test on the remaining. Raster from upper left: (a) original VMAF 0.6.1; (b) a FastVDO-variant FVMAF1, a linear model with enhanced motion measures (eqn. 5); (c), FVMAF2, with a nonlinear functional model; (d) tabulation of correlation results. Note that the True Score has spurious points above 100, whereas the Predicted Score is restricted to 100; so in Octave, we clipped all scores to 100 (this improves FVMAF1 also). The encouraging performance of FVMAF2 indicates some value in using nonlinear fusion of basic measures. Recall there are actually 7 input parameters (4 for VIF, DLM, M, DM); for simplicity, we initially use a factorized functional model.

Application to HDR Video Quality Analysis

Now that we have covered how to develop novel, and powerful, fused VQA measures, we can consider specializations to applications such as HDR and 360 videos. Here we only indicate candidate measures to add in the mix to construct a fused metric; new performance results will be reported elsewhere. HDR video is video characterized by both high dynamic range (typically 1K cd/m^2 or higher), significantly beyond the standard dynamic range (SDR) video of typically 100 cd/m^2, and wide color gamut (typically BT.2020), with a color volume significantly beyond the standard BT.709. This type of video provides noticeable value in high end consumer applications such as home theater, but also presents unique challenges in video compression. [37] catalogs a long list of proposed HDR IQAs. Here we can choose to include the wPSNR (or wMSE) metric among the measures to fuse, which is currently used in HDR codec video evaluation, as well the encoder optimization work, in the VVC codec development effort. This metric weighs errors in samples according to brightness, where brighter samples are weighed more (more noticeable). **wPSNR** is computed as follows:

(8) $$wPSNR = 10 * log \frac{X^2}{wMSE}, \quad wMSE = \sum_{all\ pixels\ i\ in\ block} w_i \left(luma\left(x_{orig,i}\right) \right) * \left(x_{orig,i} - x_{dec,i} \right)^2,$$

where X is the maximum pixel value for the specific bit depth. The weight, based on luma, is computed as:

(9) $y_i = 0.015 * luma(x_{orig,i}) - 1.5 - 6; \quad y_i = y_i < -3 \ ? -3 : (y_i > 6 \ ? 6 : y_i); \quad w_i(luma(x_{orig,i})) = pow(2.0, y_i \div 3.0).$

But we also have a more powerful candidate, based on our own development work in HDR video coding. We already have workable measures for SDR videos; moreover, we have a framework in which HDR is coded by first converting HDR to SDR by a scaling function (a function of spatial location and time). Rather than develop the mechanism here, we cite [15, 34] and our own SPIE paper here, "HDR Compression in the JVET Codec," San Diego, August, 2018. In essence, we develop a monochrome scaling function $\lambda = \lambda(x,y,t)$ – visualizable as a grayscale video – that captures the surplus information in an HDR video over SDR. Let's call this function the Dynamic Surplus (**DS**). Now, in comparing an original and a distorted (compressed video), we can take standard Lp measures of the difference in these functions, what we can call the differential dynamic surplus (**DDS**), as the error E_HDR. For p=2, this is again MSE, but of this DS function.

(10) RGB_HDR = λ(x,y,t)*RGB_SDR. DS = || λ(x,y,t)||, where ||*|| is an Lp measure, p>=1, i.e., p=1, or 2.

(11) E_HDR = DDS = || λ_orig – λ_dis ||, where ||*|| is an Lp measure, p>=1, i.e., p=1, 2.

This would be in addition to previous SDR-based measures, including spatial and temporal measures. We note that among other work, [37] compiled the performance of a long list of potential HDR Image Quality Assessment metrics (HDR IQAs). In their study, they found the top two performers were HDR-VDP2, and HDR-VQM, both achieving above 0.95 in both PCC and SPCC. HDR-VQM is a variant of the VQM we have discussed, while HDR-VDP2 is a well-studied HDR IQA. While such high scores for HDR image analysis are encouraging, we have not found strong supporting evidence that these measures are as effective in video quality analysis -- to date, no objective metrics for HDR video have been widely accepted as being highly predictive of human visual scores. However, the use of such measures in the mix, along with our differential motion DM, and differential dynamic surplus (DDS), in a potentially deep-learning fused metric, may prove powerful in capturing the quality of HDR video. Advances in this area will be reported elsewhere.

Application to 360 Video Quality Analysis

Similarly, 360 video is video that ideally lives on a 2D-sphere. In reality, sensors and displays are rectangular, so projection formats play a central role, and a commonly used projection is the equi-rectangular projection, ERP. For 360 video, we can choose to include the **WS-PSNR** (or WMSE) for 360 video [15] among the measures to fuse, which is currently used in the assessment of 360 video in the VVC codec development effort. In brief, the WS-PSNR differs from the ordinary PSNR in that it accounts for the sampling difference between a flat (rectangular) representation and a true spherical one. Since in the ERP domain, the sampling density diverges towards the poles, it is suitably weighted by a cosine, given by:

(12)
$$WS\_PSNR = 10 \log\left(\frac{MAX_I^2}{WMSE}\right), \qquad w(i,j)_{ERP} = \cos\frac{(j+0.5-N/2)\pi}{N}$$

$$WMSE = \frac{1}{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1} w(i,j)} \sum_{i=0}^{M-1}\sum_{j=0}^{N-1} (y(i,j)-y'(i,j))^2 * w(i,j)$$

5 COsNCLUSIONS

We have reviewed some applications of deep learning in video coding development, focusing on video quality assessment (VQA). In turn, VQA can be directly used in coding decisions, provided that the computational complexity of the VQA permits its insertion into the millions of decisions an encoder must make. In this regard, we reviewed some state of the art VQAs, including the VMAF measure. We then proposed enhancements, related to motion representation, which are modest in computational burden, but add value. We also proposed a simple nonlinear model, fast computable with a lookup table, which further improves the performance. This preliminary result looks encouraging, and will be investigated with more extensive datasets in future publications. We also suggest modifications for use in measuring video quality in the context of HDR and 360 videos.

6 REFERENCES

[1] R. Sekuler et al, "Motion Perception," book chapter in Steven Yantis (ed.), *Steven's Handbook of Experimental Psychology*, John Wiley and Sons, 2002.
[2] https://www.sciencedaily.com/releases/2015/06/150616190723.htm
[3] https://www.cnbc.com/2018/01/30/digital-tv-and-video-industry-to-exceed-100-billion-study-says.html
[4] https://medium.com/netflix-techblog/toward-a-practical-perceptual-video-quality-metric-653f208b9652

[5] Netflix. https://github.com/Netflix/vmaf
[6] H. Sheikh et al, *Alan, "Image Info. and Visual Quality". IEEE Tr. on Image Proc., 15(2):430–444, 2006.*
[7] S. Li et al, "Image Quality Assessment by Separately Evaluating Detail Losses and Additive Impairments," IEEE Trans. Multimedia, 13 (5): 935-949, Oct., 2012.
[8] http://fortune.com/2015/10/08/netflix-bandwith/
[9] G. Sullivan et al, "Rate Distortion Optim. for Video Compression," IEEE Sig. Proc. Mag., Dec., 1998.
[10] VQM-VFD. https://www.its.bldrdoc.gov/resources/video-quality-research/vqm-faq.aspx
[11] M. Pinson et al, "Temporal Video Quality Model Accounting for Variable Frame Delay Distortions," IEEE Trans. Broadcasting, v. 60 (4), Dec., 2014.
[12] Y. LeCun et al, "Deep Learning," Nature, vol. 521, May, 2015.
[13] D. Marr, *Vision*, W. H. Freeman and Company, 1982 (reissued by MIT Press, 2010).
[14] P. Topiwala (ed.), *Wavelet Image and Video Compression*, Kluwer/Springer, 1998.
[15] A. Segall et al, "Joint Call for Proposals on Video Coding with Capability beyond HEVC," JVET-H1002, Macau, CN, Oct., 2017.
[16] C. Bampis et al, "SpatioTemporal Feature Integration and Model Fusion for Full Ref. VQA," https://arxiv.org/pdf/1804.04813.
[17] Fortune. http://fortune.com/2015/10/08/netflix-bandwith/
[18] L. Zhou et al, "Convolutional Neural Network Filter (CNNF) for intra frame," ITU|ISO/IEC JVET-I0022, Gwangju, KR, Jan., 2018
[19] T. Wang et al, "A Novel Deep Learning-Based Method for Improving Coding Eff. From the Decoder-end of HEVC," IEEE Data Compression Conf. (DCC), Snowbird, UT, 2017.
[20] C. Dong et al, "Compression Artifacts Reduction by a Deep Convolutional Network," IEEE Int'l Conf. Computer Vision (ICCV), 2015.
[21] VQEG. https://www.its.bldrdoc.gov/vqeg/about-vqeg.aspx
[22] NTIA. https://www.ntia.doc.gov/
[29] G. Sullivan, P. Topiwala, A. Luthra, "The H.264/AVC Advanced Video Coding Standard: Overview and Intro. to the Fid. Range Extensions," Proc. SPIE, Denver, July, 2004.
[30] ITU-T/H.265, High-efficiency video coding, 2018. https://www.itu.int/rec/T-REC-H.265-201802-I/en
[31] B. Bross, "Versatile Video Coding (Draft 1)," JVET-1001, San Diego, CA, April, 2018. http://phenix.it-sudparis.eu/jvet/
[32] K. Rao, J. Hwang, *Techniques & Standards for Image, Video and Audio Coding*, Prentice-Hall, 1996.
[34] ISO/IEC JTC 1/SC 29/WG 11 N 17050, TR 23008-15. "Information technology — High efficiency coding and media delivery in heterogeneous environments — Part 15: Signalling, backward compatibility and display adaptation for HDR/WCG video," Draft, April, 2018.
[35] I. Daubechies, "Orthonormal Bases of Compactly Supported Wavelets," Com. Pure Appl. Math., Oct., 1988.
[36] S. Mallat, "A Theory of Multiresolution Signal Decomp.:…" IEEE Trans. PAMI, v.11, no.7, July, 1989.
[37] P. Hanhart et al, "Benchmarking of objective metrics for HDR image quality assessment," EURASIP J. Im. Video Proc., 2015:39.
[38] ITU-T/H.261, "Video Codec for Audiovisual Services at px64 kbits," ITU publications, March, 1993.
[39] G. Sullivan, et al, "Overview of the High-Efficiency Video Coding (HEVC) Standard," IEEE Trans. CSVT, v. 22, no. 12, Dec., 2012.

AI-related References/Bibliography

[aml] https://ai.googleblog.com/2018/03/using-evolutionary-automl-to-discover.html
[aml2] https://www.sciencealert.com/google-s-machine-learning-software-has-learned-to-replicate-itself
[aml3] https://ai.googleblog.com/2017/11/automl-for-large-scale-image.html
[deepmindRL] Google website, https://deepmind.com/blog/deep-reinforcement-learning/
[DLbook] I. Goodfellow et al, *Deep Learning*, MIT Press, 2016.
[Incv4] C. Szegedy et al, "Inception-v4, Inception-ResNet…" https://arxiv.org/abs/1602.07261, 2016.
[LeCunHinton] Y. LeCun et al, "Deep Learning," Nature, vol. 521, May, 2015.
[nasnetG] B. Zoph et al, "Learn. Transf. Arch. …," www.arxiv.org/1707.07012
[neuArchSearchG] B. Zoph et al, "Neural Arch. Search…," www.arxiv.org/1611.01578
[arcnn] C. Dong et al, "Compression Artifacts Reduction by a Deep Conv. Network," Int. Conf. Comp. Vision, 2015.
[vrRnn] G. Toderici et al, "Variable Rate Image Compression with RNNs", https://arxiv.org/pdf/1511.06085.pdf.

[vrcnn] Y. Dai et al, "A Conv. NN Approach to Post-Proc. in HEVC Intra Coding," https://arxiv.org/abs/1608.06690.
[dcad] T. Wang et al, "A novel deep learning based method of improving coding efficiency from decoder-end of HEVC," Proc. Data Comp. Conf. (DCC), 2017.
[mfqe] R. Yang et al, "Multi-Frame Qual. Enh. For Compressed Video," https://arxiv.org/abs/1803.04680.
[nndct] Q. Wu et al, "Spiking Neural Network Performs DCT for Visual Images," ICIC 2009.
[cnnf] L. Zhou et al, "Conv. NN Filter (CNNF) for intra frame," JCTVC-I0022, Gwanju, Korea, 20-26 Jan., 2018.
[nnme] D. Teney et al, "Learning to Extract Motion from Videos in Conv. NNs." https://arxiv.org/abs/1601.07532.
[nnvq] A. Ahalt et al, "Vector Quantization with Artificial Neural Networks," Proc. Int'l Workshop on Adaptive Methods and Emergent Techniques for Sig. Proc. and Comm., Bayona, Spain, pp. 42-61, June, 1993.
[resDeb] S. Kosslyn, *Image and Brain:…*, MIT Press, 1994.
[ResNet] K. He et al, "Deep Residual Learning for Im. Recog.," https://arxiv.org/abs/1512.03385, 2015.
[rossInstSeg] R. Girshick, "Deep Learn. Inst.-lev…,", CVPR Tutorial, 2017.
[siftsurfhog] S. Raj et al, "Comp. Study of Algs….," Int. J. Comp. Sc. and IT, v. 8, 163-166, 2017.
[vid2natLang] K. Saenko, "Attentive Captioning…," Lecture Notes, UC Berkeley, 2017.
[VSze] V. Sze, et al, "Efficient Processing of Deep Neural Networks: A Tutorial and Survey," Proc. of IEEE, Dec., 2017.
[wikMLdata] https://en.wikipedia.org/wiki/List_of_datasets_for_machine_learning_research
[YBbook] Y. Bengio, *Learning Deep Architectures for AI*, https://www.iro.umontreal.ca/~lisa/pointeurs/TR1312.pdf.