APSIPA Transactions on Signal and Information Processing, 2023, 12, e36

Overview Paper

Overview of Intelligent Signal Processing Systems

Kun-Chih (Jimmy) Chen[1], Wen-Hsiao Peng[1] and Chris Gwo Giun Lee[2*]

[1]*National Yang Ming Chiao Tung University, Taiwan*
[2]*National Cheng Kung University, Taiwan*

ABSTRACT

Niklaus Emil Wirth introduced the innovative concept of Programming = Algorithm + Data Structure [109]. Inspired by this, we advance the concept to the next level by stating that Design = Algorithm + Architecture. With a concurrent exploration of algorithm and architecture called algorithm/architecture co-exploration, this paper provides an overview of the leading paradigm shift in advanced visual and signal processing system design from embedded systems to the cloud and edge. As algorithms with high accuracy become exceedingly more complex and edge or Internet-of-Things generated data become increasingly larger, flexible parallel and reconfigurable processing are crucial in the design of lightweight systems with low complexity and low power. Therefore, the intelligent designs crossing levels of algorithm, system architecture, and microarchitecture, based on algorithmic-intrinsic complexity assessments, including efficient computation, data storage, data transfer, and potentials for parallelism are crucial and are surveyed. In particular, at the algorithmic level, this paper surveys state-of-the-art learned image and video codecs and their low-complexity implementations. The analytics architecture is also overviewed to explore the joint algorithmic and architecture co-design space. Furthermore, we survey intelligent technologies to control the system temperature and power consumption under a safe

*Corresponding author: Chris Gwo Giun Lee; email: clee@mail.ncku.edu.tw.

Received 01 August 2022; Revised 30 June 2023
ISSN 2048-7703; DOI 10.1561/116.00000053

computing environment for low-power design at the microarchitecture level.

Keywords: Intelligent, low complexity, low power, learning-based codec, analytics architecture, thermal-aware control.

1 Introduction

Visual signal processing system[1] (VSPS) based on artificial intelligence (AI) and beyond 5G communication will constitute futuristic technologies to enable next-generation SMARTECH. These include new emerging applications, such as biotech, autonomous cars, etc. Algorithms with higher accuracies have become exceedingly complex in the cloud. Recently, due to applications requiring faster responses, decentralized cloud, having low communication latency and low-complexity have gained popularity. Anticipating carbon footprint reduction, low-power edge systems, are also in high demand. To satisfy these design goals, lightweight intelligent systems have been introduced.

The developmental trend of VSPSs, with design of algorithms on architectural platforms, is depicted in Figure 1. Traditional software design on general-purpose central processing units (CPUs), although flexible for implementing algorithmic changes, are inefficient and require high power consumption. In contrast, in the early 1980s, application-specific integrated circuit (ASIC) hardware designs have been introduced. Although having high performance and low power, they required redesigning upon changes in the algorithm; thus, they lacked flexibility. Hence, in the 1990s, the field-programmable gate array (FPGA) was introduced with high-speed performance, medium flexibility owing to reconfigurability, and medium power requirements. The programmable instruction set digital signal processor (DSP) and application-specific instruction set processor, characterized by high flexibility, medium performance, and semi-high-power consumption, were introduced in the 2000s. Embedded multicore processors, graphics processing units (GPUs), and cloud computing platforms characterizing highly flexible algorithm programmability, but requiring high power, have also been introduced recently. Edge and neuromorphic computing from the other end, characterized by high performance and low power, has been introduced by trading off precision and low latency.

As AI algorithms requiring high accuracy become exceedingly more complex and Edge/IoT-generated data become increasingly larger, cross-level-of-abstraction processing is crucial in the design of efficient intelligent VSPSs requiring low complexity and low power. Therefore, this requires design information from both algorithmic behavior and architectural aspects, which

[1]All acronyms and corresponding full names are tabulated in the Appendix.

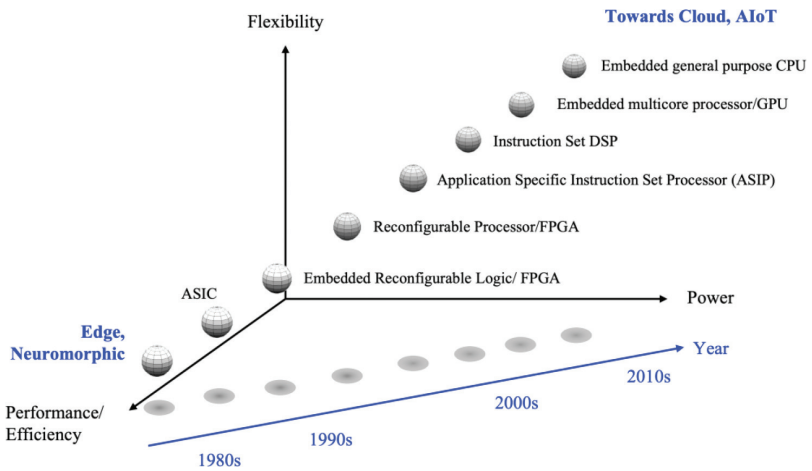
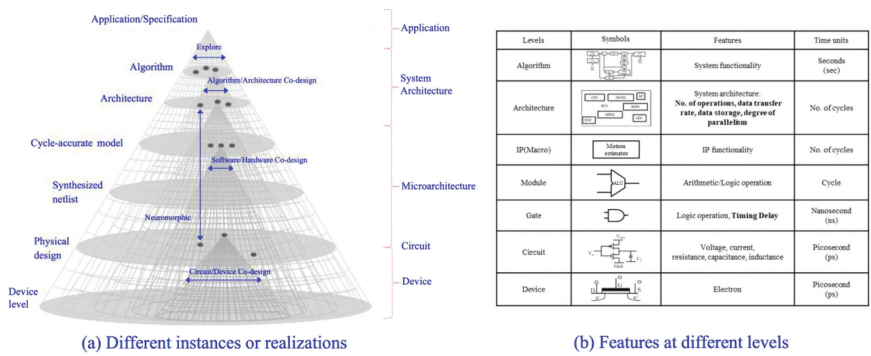


Figure 1: Trend for visual and signal processing systems.

include both software and hardware in cross-level design space exploration (DSE), as illustrated in Figure 2.

DSE is a top-down design methodology originally introduced by Kienhuis [59] to address the design challenges of large systems. Figure 2(a) depicts the design space spanned by the design parameters or features, tabulated in Figure 2(b), at different corresponding levels of abstraction, from application to algorithms, system architecture, microarchitecture, circuits, and devices. The abstraction of physical characteristics to higher levels within the design space enables the design to be focused on pertinent parameters. Therefore, this facilitates higher design efficiency of larger systems. The key essence of top-down methodology is to gain insight into lower-level information, such



Levels	Symbols	Features	Time units
Algorithm		System functionality	Seconds (sec)
Architecture		System architecture: **No. of operations, data transfer rate, data storage, degree of parallelism**	No. of cycles
IP(Macro)		IP functionality	No. of cycles
Module		Arithmetic/Logic operation	Cycle
Gate		Logic operation, **Timing Delay**	Nanosecond (ns)
Circuit		Voltage, current, resistance, capacitance, inductance	Picosecond (ps)
Device		Electron	Picosecond (ps)

Figure 2: Trend for visual and signal processing systems.

as software and hardware design parameters for the system, to make the right decision during the early design phase from the top level. According to the specification defined based on the corresponding application, a proper design solution must be selected at each abstraction level. During exploration, beginning from the algorithmic space, the selection of the right solution, represented by the apex in Figure 2(a), should span the corresponding design space or cone that encompasses all right solutions and design spaces at lower abstraction levels. Making proper decisions as early as the algorithmic level is crucial to avoiding design changes, which are more difficult during later design stages at lower levels.

Joint exploration of co-design spaces, as depicted in Figure 2(a), facilitates a systematic mechanism by which design spaces are crossed or traversed from top to bottom levels. Algorithm/architecture co-exploration (AAC) explores the joint algorithmic and architecture co-design space, where algorithmic functionality or behavior is mapped onto the system architecture design parameters, as described in Figure 2(b), through a transaction-level model (TLM) or dataflow model (DFM). Software/hardware co-design (SHC) maps the system architecture design parameters onto the microarchitecture or very large scale integration (VLSI) level. Recently, the joint exploration of circuit and device design spaces in mapping physical characteristics to electronic levels, or circuit/device co-design (CDC), has also gained popularity.

The abstraction of the design space, with corresponding design features, at different levels facilitates more efficient and especially feasible designs of larger systems. As depicted in Figure 2(b), physical features such as resistance, capacitance, and inductance at the circuit level are abstracted into the "timing delay" feature at the microarchitecture level, which characterizes the time required to achieve a sinusoidal steady state. For very large systems, design parameters are further abstracted into algorithmic-intrinsic complexity metrics, including the number of operations, data storage, data transfer rate, and the degree of parallelism, which models the transaction or flow of data at the system architecture level. This TLM or DFM can be mapped onto any of the platforms outlined in Figure 1 and is therefore platform-independent. Similar to the abstraction of physical characteristics from the circuit level (Figure 2(b)) to the microarchitecture level in VLSI system designs via SHC [103], AAC was introduced by Lee *et al.* [69] to provide an analytics architecture for abstraction at the system level. AAC provides information on both the software and hardware requirements of VSPS platforms during the early design phase, beginning from algorithms at the top level, which mitigates the potential pitfalls of system integration failure.

AAC addresses the challenges for system design for various scenarios. Traditional VSPS designs were based on either algorithmic or architectural works that have been explored or developed independently. A frequently encountered scenario involves mapping highly complex algorithms under exploration onto

existing architectural platforms as outlined (Figure 1), which may either be challenging or unfeasible. As depicted in Figure 2(a), this corresponds to the choice of either the left or middle dot representing the solution at the algorithmic level, which is outside the design space or scope framed by the cone, with the apex dot representing the appropriate solution. A second scenario may require the mapping of a fully specified algorithm onto a non-existing platform with design parameters to be explored accordingly. As shown in Figure 2(a), having fully specified the right solution at the algorithmic level, architectural solutions within the framed cone should be selected to ensure or efficiently design the computational platform selected from Figure 1. In the third scenario, having the highest degree of freedom, both algorithms and architectures can be developed freely. In all three design scenarios, proper mapping of algorithms onto architectural platforms could be assured, by joint AAC space exploration, to avoid either over- or under-designing at both algorithmic and architectural levels.

Section 2 surveys the analytics architecture based on AAC, with Sections 3 and 4 serving as case studies, at the algorithmic and architectural levels respectively. Section 3 provides an overview of learning-based image and video codecs to address recent advances in end-to-end learned image and video compression based on the four complexity metrics. Section 4 investigates the technologies used to control system temperature and power consumption in a safe computing environment.

2 Algorithm/Architecture Co-Design

In AI, analytics algorithms are typically used to analyze speech, image, video data, etc. In AAC, different algorithmic realizations are analyzed in the form of DFM to further increase efficiency and flexibility in constituting "analytics architecture". Former Architecture-C model and TLM, which consist of both algorithmic behavior and architectural implementation information, have been developed into DFM that is widely used to model algorithms and architecture concurrently [3, 10, 41, 42, 61]. Each dataflow contains a different processing scheduling or order at a specific data granularity and can be considered a different architecture instantiation or realization for the same algorithm. AAC explores the joint algorithm and architecture co-design space. It provides guidance in exploring various DFMs of an algorithm. With quantitative analysis based on complexity metrics, the most suitable DFM is chosen to be mapped onto an optimal architectural design. As such, the DFM serves as a bridge in mapping algorithms onto architectural platforms.

A graph in conjunction with dataflow, or dataflow graph (DFG) provides an efficient and systematic formalism for modeling algorithmic realizations together with architectural information on computation [68, 70]. In addition, having information on both algorithmic behavior and architectural informa-

tion including software and hardware, the DFG provides a mathematical representation which better models the underlying computational platform for systematic analysis, and thus providing flexible and efficient management of the computing resources. Traditional linear difference equation, however, describes only algorithmic behavior.

Section 2.1 provides an overview on algorithmic-intrinsic complexity metrics. Intelligent signal processing for parallel and reconfigurable computing systems are discussed in Section Section 2.2. Section 2.2 further provides overview on emerging cross-level system.

2.1 Metrics and Assessment of Algorithmic Intrinsic Complexity

Complexity assessment of signal processing algorithms, in accommodating versatile scenarios and platforms, poses a major challenge in system design. Conventional assessment of the computer algorithm's complexity was performed via software execution times based on an ideal single-thread machine with infinite memory and bandwidth. However, advanced VSPS designs require highly precise complexity measures, that are platform independent, and could be used for both software and/or hardware. As such, Lee *et al.* [69] introduced four metrics to measure the algorithmic intrinsic complexity of different algorithmic realizations or DFGs: (I) number of operations, (II) data storage (III) data transfer rate, and (IV) degree of parallelism.

The number of operations characterizes the number of arithmetic operations in a DFG. This is a factor that was traditionally assessed as instruction execution time, without distinguishing different types of arithmetic operations, which are, in reality, characterized by very different complexities. A multiplication, for example, should be more complex than an addition because the underlying cost of multiplication is composed of multiple additions. Subtraction may be assessed as almost the same complexity with addition, as it could be implemented as addition of 2's complement. Division is the most complex arithmetic operation and hence requires the highest power consumption. Complexity for constant or variable arithmetic operations should also be differentiated. For instance, constant addition is less complex than variable addition because it might be optimized in advance. Furthermore, precision of the arithmetic operations should be considered for low power designs, even at the algorithmic level, as this affects the potential wiring layout for the datapath. Therefore, based on the assessment in [69], the number of operations is more transparent in actual scenarios as opposed to timing complexity.

Data storage, as estimated from the DFG, is the buffer size required owing to data lifetime. This is affected by data causality, whereby some necessary data must be maintained for a certain period for subsequent processing. In digital systems, data lifetime is a general method of estimating the minimum storage requirement [91]. Nonetheless, this metric is highly dependent on the data

granularity and processing order in different DFGs. By exploring different data granularities and processing orders of DFGs [5], the system might achieve lower requirements for data storage. This data storage metric constitutes the second highest power consumption factor among the four complexity metrics.

The data transfer rate is the amount of data communicated between modules and when estimated from DFG, characterizes average bandwidth which is algorithmic-intrinsic. The peak bandwidth is measured when the targeted platform is specified. For different storage hierarchies, the transfer speed and cost differ significantly according to the data access pattern, e.g., burst access is more complex than discrete access. Generally, the average data transfer rate is measured as a baseline, followed by refinements using the precise characteristics of the targeted platform [69]. The data transfer metric constitutes the highest power consumption factor in complexity analysis.

The degree of parallelism, at the data level, is the potential of a DFG to be parallelized to increase the throughput. This feature enables a reduction in power consumption by lowering the working frequency or shortening the working speed while providing the same throughput. If an algorithm requiring high computations can be parallelized easily, its complexity should not be considered high because many platforms, such as SIMD, MIMD, VLIW, and SIMT, provide multiple approaches to parallelize the tasks. Reconfigurability may also be used to extract commonalities in maximizing software and/or hardware utilization, and which similar to parallelism, should also reduce design complexity and lower power consumption.

It is noteworthy that these four metrics must be considered wholistically for precise algorithmic complexity assessment. Previous algorithms, for example, have been designed to reduce the number of operations which are, however, under the cost of higher data storage and hence requires higher power consumptions. Multiple pass algorithms would also increase data transfer rate and hence complexity with correspondingly higher power. These algorithm intrinsic metrics, although providing quantitative measurements in a proportionate sense, should render sufficient insight into comparative complexity and power analysis at the algorithmic level, as will be surveyed in the case studies in Section 3.

In DSE as illustrated in Figure 2(a), these four algorithmic intrinsic metrics, depicted in Figure 2(b), facilitate complexity assessment of algorithms which are independent of, and could be mapped onto, all platforms as outlined in Figure 1. Upon progression to later system level design stages, parameters corresponding to selected platform for targeted applications, would be incorporated. As such, the system is further optimized via SHC towards embedded software and/or VLSI hardware design at the microarchitecture level.

2.2 Intelligent and Emerging Cross-Level Systems

Intelligent signal processing has gained high popularity in the last decade. In the recent introduction of analytics architecture, parallel and reconfigurable computing has been formulated via DFG which are analogous to the analysis and synthesis equations of the well-known Fourier transform pair [67]. In parallel computing, a connected component is eigen-decomposed to unconnected components for concurrent processing. For computation resource saving, commonalities in DFGs are analyzed for reuse when synthesizing or reconfiguring the VSPS platform.

Based on spectral graph theory, Lee *et al.* [70] eigen-decomposed connected DFGs to spectrum of eigenvalues. The linearly independent eigenvectors corresponding to zero eigenvalues are unconnected graph components which could then be processed in parallel. This analytic spectrum of unconnected components, as in machine learning, serves as information or features extracted from the original connected DFG signal. In addition to quantification of parallelization via eigenvalues, the eigenvectors should further facilitate instruction set architecture design of the underlying computational platform. Decision making is performed via the bi-partite or k-partitioning based on the principle axis theorem optimized for data independency. In reconfigurable computing, commonalities are analyzed on DFGs for reuse. Chen *et al.* [23] introduce an efficient and flexible architecture with algorithmic convolution for CNN which are eigen-transformed to matrix operations with higher symmetry which facilitates fewer operations, lower data transfer rate and storage anticipating lower power when synthesizing or reconfiguring the eigenvectors for the computational platform. As such, recent advancement in analytics architecture has enabled "AI in designing AI".

The deep learning algorithm can be highly parallelized because of the characteristics of tensor operations; however, for different tensor configurations, reconfiguring processing elements is required to maximize throughput, i.e., increasing utilization [99]. More studies have been dedicated to designing Edge AI chips to increase the degree of parallelism and reconfigurability to enhance hardware utilization [26, 45, 90]. NVIDIA provides the NVIDIA Deep Learning Accelerator (NVDLA) with efforts to standardize the architecture for an inference engine for deep neural networks [117]. More case studies on intelligent multicore systems are provided in Section 4.

Cross-level embedded systems designed based on AAC constitutes another significant trend in intelligent lightweight signal processing. Cassidy and Andreou [11] introduced Amdahl's law-based optimization function for crossing parallel computing at the algorithmic and system levels to the microarchitecture level, targeting low power and low delay. Kung *et al.* [64] demonstrated the feasibility of lowering the power consumption during the acceleration of convolutional neural network (CNN) inferencing. This was facilitated by

the systolic array and CNN algorithm to reduce the wire length, which was also optimized at the microarchitecture and physical layers during layout and upon mapping the design onto a 3D integrated circuit. Chen *et al.* [27] introduced Eyeriss, a cross-level algorithm and microarchitecture co-design-based accelerator for deep CNNs via optimal row stationary dataflow, to minimize the data transfer rate and hence achieve high power efficiency. Yu *et al.* [115] introduced a neuromorphic visual system to cross the system, circuits, and device levels to achieve low power consumption. A computing-in-memory edge AI system at 65 nm was provided by Chen *et al.* [25] to reduce data transfer and data storage requirements with convolution operations performed on analog circuitry fabrics and achieve low power consumption. Intelligent thermal systems case studies are further provided in Section 4.

3 Learned Image and Video Compression Systems: Design and Implementation

In this section, we review recent advances in end-to-end learned image and video compression at application and algorithmic levels. The fact that image and video compression is the underpinning technology of many ICT applications and has proven commercial value motivated our study. This AI-enabled technology has significant potential in many emerging applications, such as perceptual compression for realism, extreme low-rate compression, application-specific image/video compression, and compression for hybrid human and machine vision. Its high flexibility to different objectives distinguishes it from traditional codecs, attracting considerable interest from both academia and industry. In particular, we provide insight into its algorithm-intrinsic complexity aspects by surveying and comparing state-of-the-art learned video codecs. We also provide pointers for some recent low-complexity implementations.

Discrete cosine transform (DCT)-based image and video coding techniques have been adopted by international standards, such as the Joint Photographic Experts Group (JPEG), International Telecommunication Union (ITU) H.261/264/265/266, and Moving Picture Experts Group (MPEG)-2/4/H, for nearly 30 years. Although researchers are still attempting to improve its efficiency by fine-tuning its components and parameters, its basic structure has not changed in the past two decades. The arrival of deep learning has recently spurred a new wave of development in end-to-end learned image and video compression. This fast-growing research area has attracted more than 100+ publications in the literature, with state-of-the-art end-to-end learned image compression exhibiting comparable compression performance to H.266/versatile video coding (VVC) intra-coding in terms of peak-signal-to-noise-ratio-RGB (PSNR-RGB) and much better multi-scale structural similarity (MS-SSIM) results. End-to-end learned video coding is also catching up

quickly. Some preliminary studies have reported comparable PSNR-RGB results to H.265/high-efficiency video coding (HEVC) or even H.266/VVC under a low-delay setting. These interesting results have resulted in intense activity in international standards organizations, such as JPEG AI [58], and various challenges, such as the Challenge on Learned Image Compression (CLIC) [31]at the IEEE/CVF Computer Vision and Pattern Recognition Conference (CVPR) and Grand Challenge on Neural Network-based Video Coding at the IEEE International Symposium on Circuits and Systems (ISCAS) [43].

3.1 Advances in Learned Image and Video Compression Systems

This section presents an overview of the recent advances in end-to-end learned image and video compression systems.

3.1.1 End-to-End Learned Image Compression Systems

The variational autoencoder (VAE)-based compression framework with the hyperprior [7] has become the de facto standard for end-to-end learned image compression. As shown in Figure 3, such an image compression framework features three major components: the encoder transform, decoder transform, and hyperprior autoencoder. The encoder transform converts a three-component RGB raw image $x \in R^{3\times W\times H}$ of size $W \times H$ through a convolutional neural network into a compact set of feature maps $y \in R^{320\times \frac{W}{16}\times \frac{H}{16}}$. The components in y are uniformly quantized as $\hat{y}$ before entropy coding. In particular, every component in y is often modeled as a distinct Gaussian distribution, with its

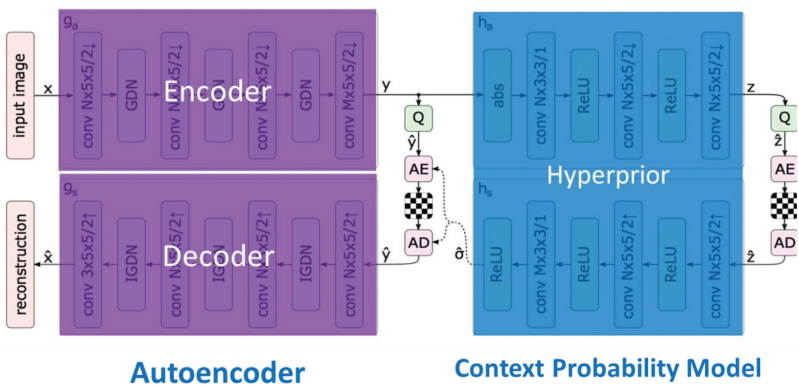


Figure 3: Illustration of the variational autoencoder (VAE)-based image compression framework with hyperprior [7]. Q, AE, AD refer to quantizer, arithmetic encoding, and arithmetic decoding, respectively. M, N are the channel numbers of the convolutional layers.

own mean and variance determined by the hyperprior autoencoder. Collectively, these distributions are exploited to evaluate the coding probabilities for $\hat{y}$ to entropy-encode $\hat{y}$ into a bitstream. The hyperprior autoencoder takes y as the input and produces another layer of latents $z \in R^{192\times\frac{W}{64}\times\frac{H}{64}}$, known as the *hyperprior*, which is the side information for deriving the coding probabilities of $\hat{y}$. The hyperprior z itself must be quantized and packed into the bitstream, often representing a small portion of the entire bitstream. The decoding process begins by decoding the quantized hyperprior $\hat{z}$, followed by decoding the main latents $\hat{y}$. The training of the three components is performed end-to-end with an objective that aims to balance the distortion and rate.

Since the advent of the VAE-based coding framework with the hyperprior [7], extensive research effort has focused on (I) enhancing the expressiveness of the main encoder and decoder and (II) improving the entropy coding of y to demonstrate the full potential of end-to-end learned codecs. Examples in the first category include the introduction of importance maps [102], non-local attention modules [24], Swin transformer [121], block-based autoencoders [110], and recurrent neural networks [73]. The second category includes a variety of hyperprior variants, such as combining the hyperprior with the auto-regressive context model [84], the checkerboard context model [46], the channel-wise hyperprior [85], the coarse-to-fine hyperprior [49], and the use of Gaussian mixture models [28].

In addition to the VAE-based coding framework, flow-based models exist [47, 82]. One problem faced by the VAE-based framework is that the main encoder and decoder are generally lossy. Even without quantizing the main latents y, the perfect reconstruction of the input image x on the decoder side is not feasible. However, flow-based models, have the desirable property that the mapping between input x and its latent representation y is bijective and reversible, a property that is also shared by DCT or discrete wavelet transform in traditional codecs. Among the flow-based models, the one adopting augmented normalizing flows (ANF) [47] has the additional benefit of being able to incorporate any VAE-based method to improve its model expressiveness.

Figure 4 shows the rate-distortion comparison between the state-of-the-art learning-based image coding methods and the conventional coding methods, e.g., VVC [9], BPG [8], in terms of PSNR-RGB, and MS-SSIM. The best end-to-end learned codecs achieve similar PSNR-RGB results to VVC intra-coding while exhibiting much better MS-SSIM results.

On the industrial side, ISO/IEC and ITU recently launched a new standardization project, known as JPEG AI [58], with the aim of standardizing by 2024 a learning-based coding technology that can efficiently compress images and demonstrate effective performance for compressed-domain image processing and computer vision tasks. Their call for proposals [40] issued in early 2022 attracted 10 responses from academia and industry. These responses were

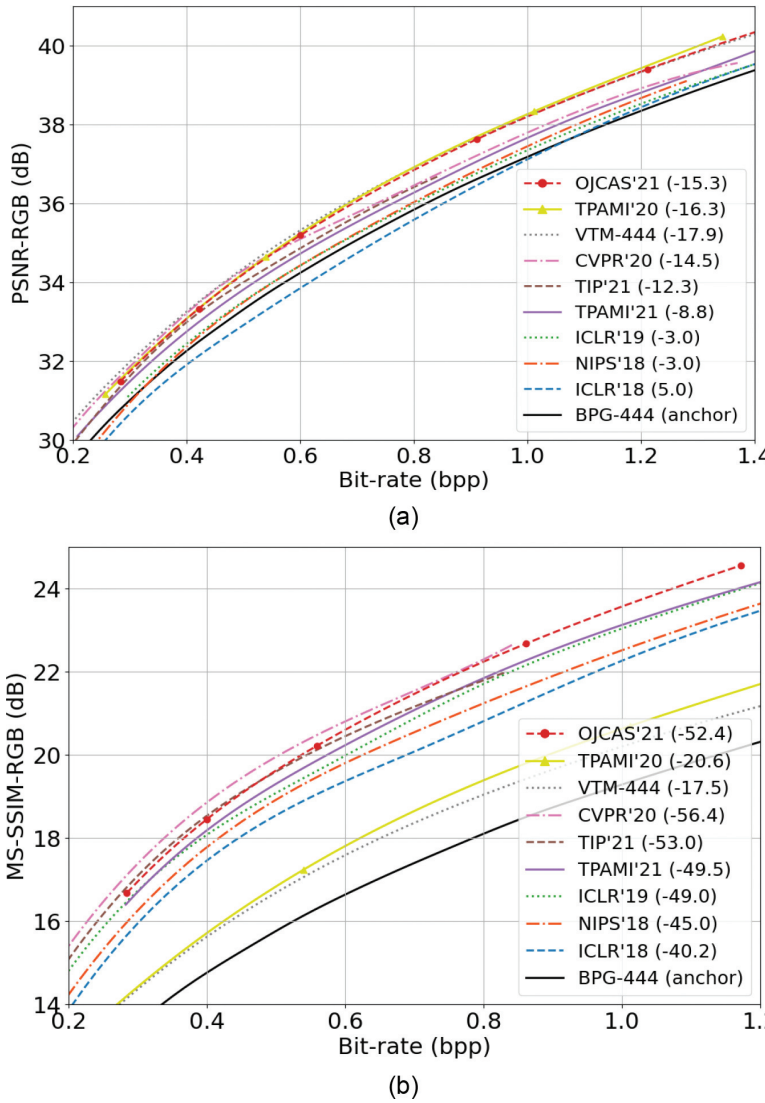


Figure 4: Rate-distortion performance evaluation [47] on Kodak image dataset [62], including OJCAS'21 [47], TPAMI'20 [82], CVPR'20 [28], TIP'21 [24], TPAMI'21 [50], ICLR'19 [50], NIPS'18 [84], ICLR'18 [7], VTM [9], BPG [8]. Except for TPAMI'21 [50], which adopts the same model for both quality metrics, the other learned image compression methods train separate models for different quality metrics. Moreover, all the learned methods achieve variable-rate compression with distinct models, i.e., one model for each rate point.

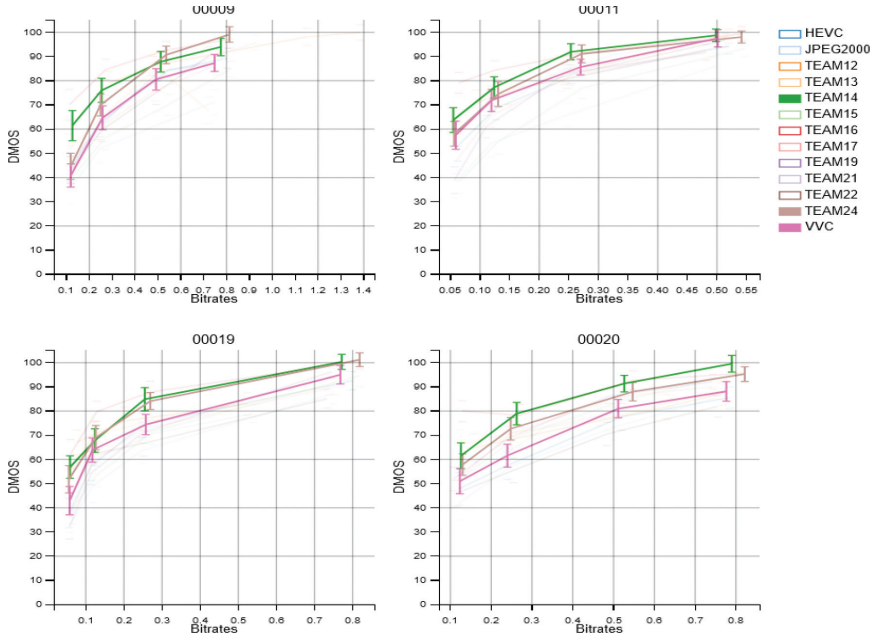


Figure 5: Subjective quality evaluation of JPEG AI Call-for-Proposals responses, where the performance of team 14 [4] and team 24 [37], and VVC [9] are highlighted.

evaluated both objectively and subjectively against several anchors, including VVC, HEVC, JPEG 2000, and JPEG XL. In terms of technologies, most proposals adopted the VAE-based scheme (Figure 3), whereas one used the ANF-based coding structure [47]. Interestingly, the top performers (team 14 [4] and team 24 [37]) exhibited better subjective quality in terms of the differential mean opinion score (DMOS) than VVC in several test cases (Figure 5). At the time of writing, JPEG AI was creating the first working draft of the standard.

3.1.2 End-to-End Learned Video Compression Systems

The success of end-to-end learned image compression has motivated a new wave of development in end-to-end learned video compression. Most learned video compression systems follow the traditional, hybrid-based coding architecture, namely, temporal prediction followed by transform-based residual coding. Figure 6 shows the differences between traditional and learned video compression frameworks. Several key components in the traditional codec have been replaced with end-to-end learnable neural networks. For example, block-based motion estimation and compensation modules can be replaced with an optical flow estimation network and a motion compensation network,

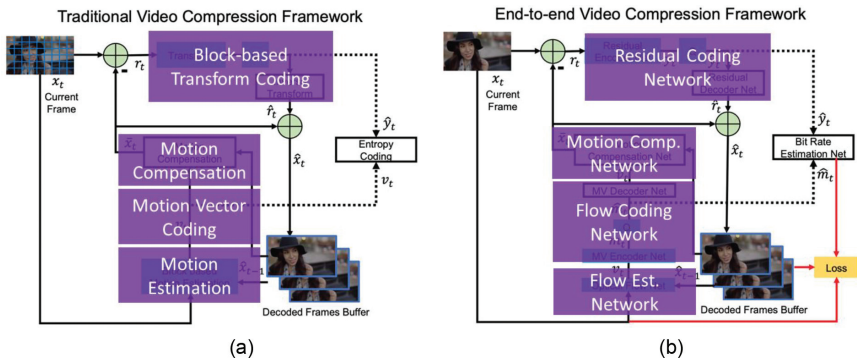


Figure 6: Comparison between (a) traditional and (b) learned hybrid-based video coding frameworks [80].

respectively. Both networks typically operate at the frame level rather than at the block level. Moreover, the learned flow and residual coding networks, which share a similar architecture to VAE-based image compression, are used instead of block-based motion vector and block-based transform coding.

Figure 7 further presents a taxonomy of end-to-end learned video compression. In terms of timeline, DVC [80] was the first work to incorporate temporal predictive coding in an end-to-end learning framework. It follows the traditional concept of estimating motion between consecutive video frames, followed by residual coding. Along this line of research, termed inter-frame residual coding, several follow-up studies have been conducted [48, 53, 74] to improve and/or extend DVC in several significant aspects. For example, to account for motion uncertainty, the scale-space flow (SSF) [2] creates a frame predictor by signaling a spatially varying Gaussian kernel for blurring the reference frame. HLVC [112] forms a multi-hypothesis prediction by using neural networks to fuse information from multiple reference frames. To reduce motion overhead, RaFC [51] borrows the classical concept of variable block-size motion compensation to adapt the resolution of the flow map features both locally and globally. MLVC [74] adopts predictive motion coding by extrapolating a flow map predictor from the decoded flow maps. Similarly, ELF-VC [97] signals an incremental flow map between the target frame and a motion-compensated frame derived from the extrapolated flow map. To better extrapolate flow maps from multiple reference frames, VLVC [39] models the pixel-based motion trajectory using a polynomial function. Taking a different approach, RLVC [113] propagates causal temporal information using a recurrent neural network to formulate a temporal prior for entropy coding and adapt the autoencoder to every residual frame. While some studies [2, 39, 48, 51, 74, 97, 112, 113] conducted motion compensation in the pixel domain, others, such as NVC [77] and FVC [53], indicated that feature-domain motion compensation can result in better prediction efficiency.

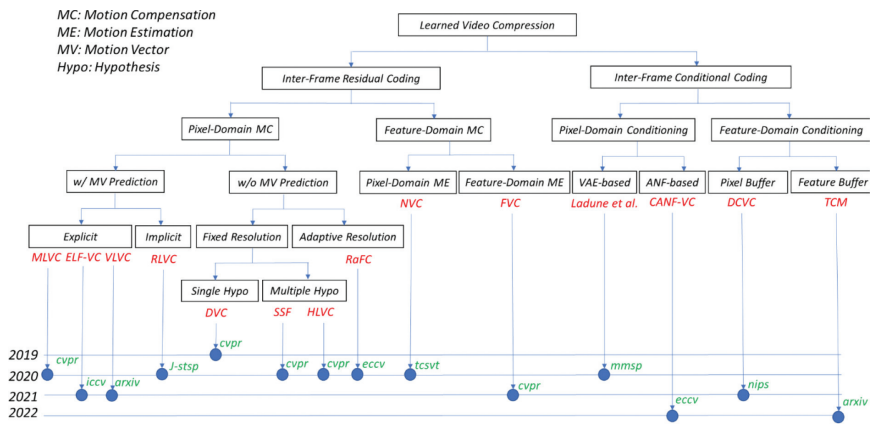


Figure 7: Taxonomy of end-to-end learned video coding methods, including MLVC [74], ELF-VC [97], VLVC [39], RLVC [113], DVC [80], SSF [2], HLVC [112], RaFC [51], NVC [77], FVC [53], Ladune *et al.* [66], DCVC [56], TCM [100], and CANF-VC [48].

More recently, a new school of thought [66], known as (inter-frame) conditional coding, has emerged (Figure 8). The concept of conditional coding involves encoding a target frame x_t conditionally based on the motion-compensated frame x_c without explicitly evaluating the prediction residual $x_t - x_c$. [66] demonstrated that the entropy rate $H(x_t - x_c)$ of the prediction residual is greater than or equal to the conditional entropy $H(x_t|x_c)$. This refutes the traditional inter-frame residual coding from being optimal in the information-theoretic sense. Some early attempts [56, 66, 100] addressed this problem by learning a conditional variational autoencoder (CVAE) to approach conditional entropy (Figure 8). In [66], this was achieved by concatenating x_c and x_t for encoding and their latent representations for decoding. DCVC [56] further uses the latent representation extracted from x_c as the conditioning variable. For its improved version (TCM) [100], the authors advocated that the conditioning variable should result from the decoded features from which the RGB frame is reconstructed, rather than the reconstructed RGB frame itself, arguing that the decoded features frequently contain more information than the decoded RGB frame. However, this comes at the cost of a significantly larger decoded picture buffer size. CANF-VC [48] presents a purely conditional coding framework using conditional ANF for both interframe and motion coding. Nevertheless, conditional coding presents promising compression results, taking end-to-end learned video codecs to a new level of compression performance. This also leaves sufficient room for further investigation.

Figure 9 presents the rate-distortion comparison among the state-of-the-art learning-based video codecs, the test model of HEVC (HM) [1], and $\times$265 (in very slow mode) [111] in terms of PSNR-RGB and MS-SSIM. The best

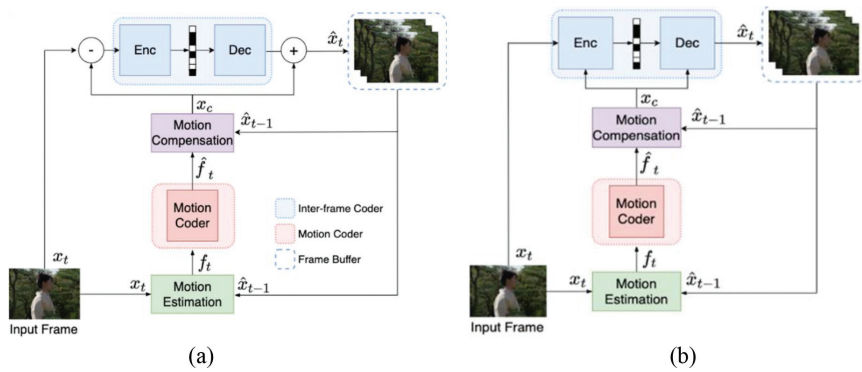


Figure 8: Illustration of (a) inter-frame residual coding and (b) inter-frame conditional coding based on conditional VAE, where x_t denotes an input frame, $\hat{x}_t$ is its reconstructed version, x_c is the motion-compensated frame, f_t is the optical flow map between x_t and $\hat{x}_{t-1}$, and $\hat{f}_t$ is the reconstructed optical flow map.

end-to-end learned codecs achieve better PSNR-RGB results than HM while exhibiting much better MS-SSIM results.

3.2 Complexity Characterization for Learned Video Compression Systems

This section presents the complexity characterization of learned video compression systems in terms of intrinsic algorithmic complexity metrics. As discussed in Section 2.1, these metrics are (I) kilo-multiply-accumulate-operations per pixel (KMAC/pixel), (II) the number of network parameters (i.e., the model size in millions of parameters), (III) the decoded picture buffer size (measured in the equivalent number of full-resolution reference frames), and (IV) the peak memory requirement (measured in the equivalent number of full-resolution feature maps). KMAC/pixel is an indication of the required number of algorithmic operations per pixel, whereas the other metrics reflect the storage requirements. In terms of the decoded picture buffer size (III), some methods require storing only the decoded video frames as reference frames, whereas others may additionally require storing feature maps, optical flow maps, etc. In comparison, the peak memory requirement (IV) is closely related to the neural network design. Some architectures may generate more feature maps to be stored than others do. Notably, both metrics (III) and (IV) have implications for memory bandwidth requirements when the reference frames and/or feature maps must be stored in the external memory. The numbers shown in Figure 10 correspond to the complexity characteristics of P-frame encoding, which also include the operations required for P-frame decoding owing to their closed-loop coding architecture. We plot Bjøntegaard-delta (BD) rate savings

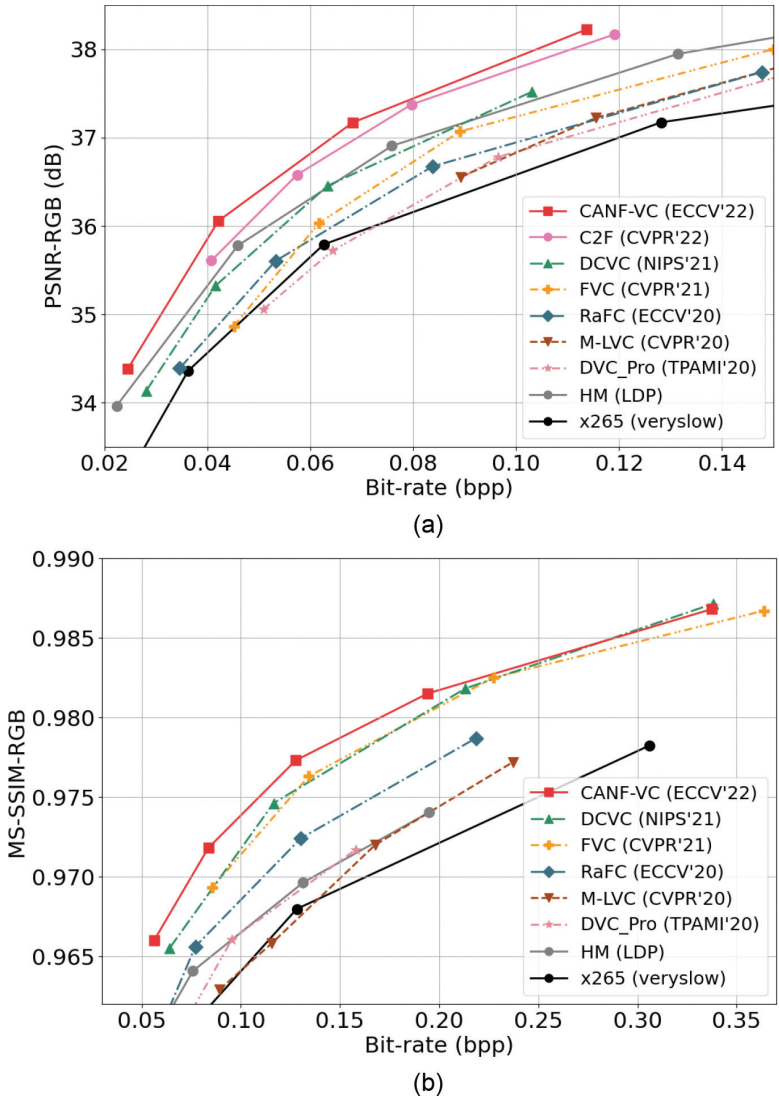


Figure 9: Rate-distortion performance evaluation [48] on UVG dataset [83], including CANF-VC [48], C2F [52], DCVC [56], FVC [53], RaFC [51], M-LVC [74], DVC_Pro [81], HM [1], and ×265 [111]. All the competing methods are evaluated with full-sequence encoding and a group-of-picture (GOP) size of 12. Note that the intra-frame codec may vary by method. Following common practice, all the learned methods achieve variable-rate compression with distinct models, i.e., one model for each rate point. Moreover, separate models are trained for different quality metrics.

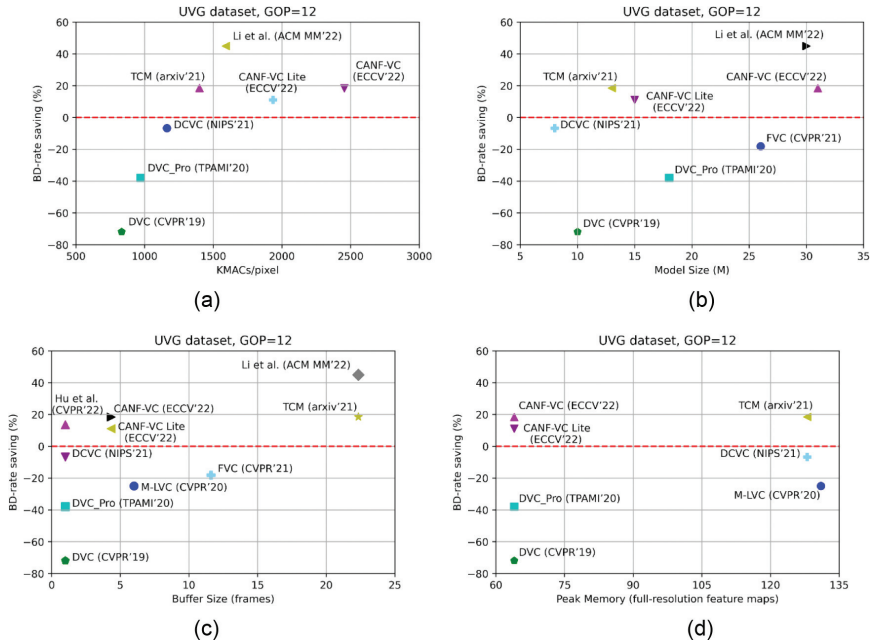


Figure 10: Complexity characterization of several learned video compression systems, including DVC [80], DVC_Pro [81], DCVC [56], TCM [100], CANF-VC Lite [48], CANF-VC [48], Li *et al.* [72], FVC [53], Hu *et al.* [52], M-LVC [74], and FVC [53]. The BD-rate savings of these methods are visualized as functions of (a) KMAC/pixel, (b) model size, (c) decoded picture buffer size, and (d) peak memory requirement. The data are provided for P-frame coding only.

relative to HM Test Model Version 16.20 [1] as functions of these complexity measurements to visualize the complexity-performance trade-offs of different competing methods.

Figure 10(a) shows the trend that the BD-rate saving increases as KMAC/pixel increases, suggesting that better compression can typically be achieved at the expense of more arithmetic operations. Notably, we can roughly deduce the KMAC/pixel of the decoder by dividing the numbers shown in the figure by a factor of 1.5. This scaling factor is derived from analyzing DVC [80] and is meant for a rough estimation. Consequently, the KMAC/pixel of the learned video decoder is approximately two to three orders of magnitude higher than the typical single-digit KMAC/pixel of the traditional decoder.

As shown in Figure 10(b), the model size does not share the same trend as that of KMAC/pixel. Although larger-sized models frequently yield better compression performance than smaller ones, in some cases, reasonably good compression performance is achieved with relatively small models, e.g., CANF-VC Lite [100] and TCM [100]. This result is attributed to the effort placed into optimizing the model size. Note that while the model size can

be a good indicator of model capacity, it may not accurately reflect model complexity.

Figure 10(c) and (d) show that, in terms of the buffer size and peak memory requirement, these competing methods present a completely different landscape. In particular, both schemes in [100] and [72], although benefiting from using high-resolution feature maps for more efficient conditional coding, demand large buffers to store these feature maps. Moreover, their peak memory requirements are much higher than those of the other works. These observations have implications for their practicality and the type of computational platform that can be utilized for their implementation.

In summary, we note that these complexity aspects have not yet gained sufficient attention. Most studies in this area are still attempting to demonstrate the full potential of learned video compression. We stress that understanding these aspects of algorithmic intrinsic complexity is the key to ensuring the practicality and efficiency of the algorithm, particularly when selecting a specific computation platform to implement the algorithm.

Please refer to [55] for the complexity characterization of learned image compression systems and [55, 75] for the runtime analyses of the learned image and video compression. Caution must be exercised when interpreting encoding and decoding runtimes as they are highly platform-dependent.

3.3 Some Notable Low-Complexity Implementations

Several paths of research have been dedicated to the low-complexity implementation of learning-based image and video compression. This section overviews some recent publications in which more related works can be found.

Among the problems of low-complexity learned image compression, the spatial auto-regressive context model has been discussed the most. Its highly sequential operation owing to the sample-based query of the autoregressive model makes it easily the computational bottleneck of the decoder. He *et al.* [46] introduced a checkerboard context model to break the strong dependencies between spatially adjacent image latents. Similarly, Minnen and Singh [85] proposed a channel-wise autoregressive-entropy model. Generally, these methods sacrifice compression performance for lower complexity. In contrast, Esenlik *et al.* [37] and Alshina *et al.* [4] leveraged data parallelism to implement an autoregressive context model based on wavefront parallel processing, which exhibited only a slight impact on compression performance.

Following the same concept of wavefront parallel processing, Wu *et al.* [110] presented a block-based learned image compression system that divides an image into rows of non-overlapping blocks and processes the blocks in different rows simultaneously. For better compression, they introduced inter-block prediction from the neighboring blocks, which are to the left of and above the current coding block. Another block-based system was described in [73].

It divides an image into macroblocks, each of which contains several sub-blocks of pixels coded using a spatial recurrent neural network.

Recently, techniques have been developed to reduce the complexity of analysis and synthesis transforms. Zhu *et al.* [121] used a swin transformer [78] to replace the convolutional layers, achieving not only improved compression performance but also lower complexity. Kim *et al.* [60] introduced an asymmetric autoencoder that adopts a larger network for the analysis transform and a smaller network for the synthesis transform. They argued that the analysis transform must satisfy both the rate and distortion requirements, thereby justifying the use of a more capable network.

Another path of research addresses the precision problems of the autoencoder and hyperprior. Ballé *et al.* [6] showed that the use of floating-point arithmetic, particularly in the hyperprior autoencoder, can cause decoding to fail catastrophically when the encoding and decoding are performed on platforms that process floating-point arithmetic differently. Thus, they proposed the use of integer networks. Sun *et al.* [104] presented an effective method for quantifying network weights for fixed-point arithmetic. Jia *et al.* [57] showcased an FPGA implementation of a learned video-coding system that adopts integer networks as the backbone.

Achieving variable-rate compression using a single model is another focal point of research. Separate autoencoders are generally trained to optimize the compression performance at different rate points. This approach is prohibitively expensive and impractical for real-world applications. Jia *et al.* [57] addressed this problem by introducing affine feature transformation layers to adapt feature maps to multiple rate points, along with adaptive quantization. The concept was extended to make feature transformation spatially adaptive [29].

Finally, the reader is referred to [57, 119], and [105] for real-time implementations of learned image and video compression. We note that Sun *et al.* [105] proposed a fine-grained processing pipeline that performs row-wise convolution for different CNN layers in a fully synchronized manner. The synchronization is achieved via a ping-pong row buffer to transfer data from one CNN layer to the next. Consequently, the size of the ping-pong buffer critically determines the granularity at which the CNN layers are processed simultaneously. While their work is more of the implementation of an existing learned image codec, it touches upon several design considerations that can potentially be further optimized from the perspective of algorithm and architecture co-design.

4 Thermal-aware Low-complexity and Low-power Manycore System Designs

Regarding system architecture design, running the involved parallel computing algorithm on a manycore system is a popular method of accelerating signal

processing. However, owing to the high demands of complex applications in manycore systems, a large workload increases the operating temperature of the system and results in significant thermal and power problems. In this section, we investigate the technologies used to control system temperature and power consumption in a safe computing environment. We first survey some low-computing-cost methods using thermal sensors to monitor the transient temperature. Thus, we can reduce the thermal impact during full-system temperature estimation. Because some overheated computing units may be throttled, we further introduce data delivery methods using game theory or AI algorithms. Finally, we review the design methodology to save computing energy from architectural design perspectives, which aims to reduce the on-chip memory (e.g., buffer) and off-chip memory (e.g., DRAM) size.

As mentioned previously, the thermal problem becomes more severe owing to the high power density in the manycore system. To mitigate the thermal problem, we investigate the relationship between the phenomena of thermal and power on a manycore system in Section 4.1. Subsequently, we introduce techniques for managing the power and temperature of the system in Section 4.2. In Section 4.3, routing methods are introduced to migrate the power and temperature distribution on the manycore system to reduce the negative impact caused by thermal problems. Finally, we investigate the techniques from the hardware architecture perspective to mitigate the thermal problems in Section 4.4.

4.1 Introduction of Thermal and Power Problems in Manycore Systems

In physics, heat is defined as the transfer of thermal energy across a well-defined boundary around a thermodynamic system. The fundamental methods of heat transfer in engineering include conduction, convection, and radiation. Physical laws describe the behavior and characteristics of each of these methods. Real systems often exhibit complex combinations of variables. A thermodynamic system is defined as a portion of the mass of the universe selected for thermodynamic analysis. The system is separated from its surroundings by a boundary that encloses all the characteristics of this system. Therefore, the thermodynamic system has a fixed mass, and all matter is either in the system or in the surroundings. Exchanges in the work or heat between the system and surroundings occur across this boundary. In thermodynamics, the objective is to determine the energy (or heat) transfer and the change in the state of the system. The boundary of the manycore system is defined as a package. Moreover, no mass transfer occurs between the system and the surroundings when crossing the boundary of the manycore system (i.e., the package). Hence, a manycore system is primarily solid and fits the definition of a thermodynamic system.

The analysis of the correlation between the thermal and traffic in a manycore system can be performed macroscopically with a constant-volume constant-pressure case of the work process. According to the first law of thermodynamics, thermal energy is a portion of the internal energy, which is known as the system temperature. The sources of thermal energy are frequently data processing in each processing element (PE) or data transmission between each PE. Among the various manycore interconnections, network-on-chip (NoC) is an emerging on-chip interconnection for efficient data transmission. Therefore, the packet switching at each router of the NoC and the packet delivery between each router also contribute thermal energy to the manycore system. Therefore, the traffic behaviors in the NoC-based manycore system directly affect its thermal behaviors.

4.2 Thermal and Power Management

4.2.1 Techniques of Temperature Monitoring

Owing to the high demand for complex applications in manycore systems, a large workload increases the operating temperature of a system and results in significant thermal problems. Many dynamic thermal management (DTM) methods have been developed in recent decades to regulate the system temperature at a safe value [14, 18, 19, 32, 94, 98, 114]. DTM methods are used to control the temperature based on the sensing temperature of on-chip thermal sensors. Owing to the manufacturing cost, the number of on-chip thermal sensors is frequently restricted. For example, Intel's Sky Lake 64-core processor embeds seven thermal sensors on the chip [30]. Consequently, it is important to determine the appropriate locations for placing number-limited thermal sensors. However, the problem of thermal sensor placement has been proven to be NP-hard [96]. The search for a good thermal sensor distribution has become a design challenge in manycore system designs and has received much interest in recent years.

To determine the proper locations for thermal sensors under low searching complexity, many researchers have proposed placing them based on information about temperature behaviors [79, 120] or power dissipation [63, 116]. Zhu *et al.* [121] employed the entropy theory to evaluate the uncertainty of the on-chip temperature variation, which was used to estimate the placement of thermal sensors. Wu *et al.* [110] evaluated the distribution of thermal energy on a chip to determine the optimal location of thermal sensors. Prior to positioning the involved thermal sensors, Ranieri *et al.* [95] utilized principal component analysis and the correlation of the reference thermal maps. Mukherjee *et al.* [88] divided the platform and selected the sensor placement method depending on the sensing coverage of the included thermal sensors. Long *et al.* [79] used a grid-based approach for the placement of thermal sensors and utilized the

obtained sensing information to predict the locations of hotspots on the chip using the interpolation method. In contrast, temperature information can be considered long-term information regarding power accumulation (i.e., thermal energy) [12]. Hence, Zhang *et al.* utilized the correlation of the power density of each location to estimate the temperature distribution on the chip, which was used to locate the placements of the thermal sensors [116]. Küflüoglu *et al.* [63] defined certain power vectors to represent the various functional groups on a chip and identified hot cells, which are the optimal locations for thermal sensors. Although the aforementioned methods may minimize the search complexity required to identify areas for thermal sensor placement, the installation of thermal sensors is time-consuming because of the necessity to gather information based on previous knowledge. In addition, conventional methods have limited applications. This is because the current approaches can determine the proper locations for thermal sensor placement when the on-chip workload is fixed. However, the temperature-changing behavior of on-chip devices is often dynamic and unexpected [101]. Therefore, it is still difficult to determine appropriate locations for thermal sensor placement when the on-chip workload is unpredictable (i.e., when the application on the chip is not fixed or the on-chip temperature behavior is time-varying).

As mentioned previously, determining the optimal thermal sensor placement method to obtain precise estimates of the full-chip temperature distribution without prior information is difficult. Therefore, some studies have aimed to determine the proper locations for thermal sensor placement without any prior knowledge and to achieve accurate estimation results of the full-chip temperature distribution during runtime. To achieve this, Chen *et al.* used the compressive sensing (CS) theory to place thermal sensors and dynamically estimate the full-chip temperature distribution [15]. CS theory has been demonstrated to be an effective technique for recovering the original signals from a less-information-packed signal without any offline processing [33]. However, owing to the high computational complexity of the adopted reconstruction approach, it is unsuitable for full-chip temperature monitoring in real-time multicore systems. Additionally, this technique emphasizes the floorplan of the NoC system, which does not consider the placement of thermal sensors in the current popular multicore system platform. To leverage thermal sensor placement on a non-NoC manycore system, Chen *et al.* suggested a grid-based thermal sensor placement approach, which enables the installation of thermal sensors on abstract multicore system platforms. Moreover, the authors further applied the matrix inversion bypass (MIB) property to reduce the computational complexity of the full-chip temperature distribution estimation at runtime. Because the MIB has been demonstrated to be an effective method to implement hardware [54], the MIB-based full-chip temperature reconstruction method can aid in estimating the full-chip temperature distribution in real time.

4.2.2 Techniques of Temperature Control

To make NoC-based manycore systems thermally safe, DTM has become increasingly popular, and many different types of DTM have been proposed in the past few years. Generally, DTM can be divided into two types: reactive DTM (RDTM) and proactive DTM (PDTM) [19]. When the operating temperature of the NoC system reaches the warning level, RDTM is activated to manage the system temperature, as defined by Chao *et al.* [14]. In contrast, PDTM controls the temperature of potential thermally emergent computing units in advance based on temperature prediction [18].

Temperature control methodologies employing DTM can be subdivided into temporal and spatial DTM [18, 114]. Temporal DTM approaches attempt to regulate the processing speed of thermally emerging NoC nodes [14, 98]. The benefit of temporal DTM is that temperature can be regulated in a short period. As a result of the reduced processing speed of thermal-emergent computing units, an NoC system with a temporal DTM mechanism often experiences a significant performance reduction. In contrast, spatial DTM techniques utilize power migration to manage the system temperature of a 3D NoC system [32, 94, 114]. Spatial DTM redirects packets away from thermally emergent NoC nodes using specific routing algorithms. Thus, it is not essential to reduce the processing speed of thermally emergent NoC nodes, which mitigates the performance effect during the temperature management phase. Because the interlayers and intralayers of NoC systems do not have heterogeneous thermal conductivities, the thermal conduction of each NoC layer is different. This means that cooling takes longer than with temporal DTM approaches.

4.2.3 Techniques of Power and Thermal Management

In an NoC-based manycore system, power and temperature can be managed in several ways: 1) configuring distinct power modes for each router to control the network's total power usage; clock-gating modes can be used to deactivate idle ports; 2) regulating voltage-frequency islands to enable power to be distributed throughout the network without exceeding the power budget; 3) adopting congestion-aware routing to prevent the creation of hotspots in high-traffic areas of networks.

Power and temperature management in NoCs frequently emphasizes congestion-aware routing algorithms to distribute the traffic load among the platform's available resources. Several papers on 2D NoCs have been presented [5, 36]. However, attempts are significantly more restricted in 3D NoCs owing to the difficulty of developing deadlock-free routing algorithms in wormhole switching networks. [108] focused on power optimization for 3D regular NoCs, whereas [13] optimized performance with temperature considerations. In this study, to optimize performance, the routing algorithm propagated power to the

bottom layer and adjusted traffic levels to prevent packet congestion. However, the proposed approach is static and has a limited capacity for traffic load balancing. Although power consumption was not explicitly addressed in [13], we may deduce that lower traffic congestion and improved thermal distribution result in reduced power consumption. However, the majority of previous research focused on addressing latency, power, and temperature challenges in homogeneous networks.

Several learning-based algorithms and methods have been proposed for interconnection networks [36, 93, 118]. A reinforcement learning (RL) technique was used in [118] to increase the energy efficiency of NoCs by automatically learning an optimal control policy. In Bi-LCQ [38], the Q-routing method is applied to a cluster-based NoC. A different attempt from routing was presented in [93], called SVR-NoC. Using support vector regression, this strategy attempts to forecast the traffic flow delay and average channel waiting time. In [108], a Q-leaning-based technique was used to regulate power in irregular NoCs via deflection routing. HARAQ [36] blends Q-routing with non-minimal routing to provide a high-performance wormhole switching network. Generally, by using optimal or near-optimal routes between each pair of source and destination nodes, the latency, power, or temperature can be automatically modified using a flexible routing scheme, such as that suggested by EbDa [34, 35], and by utilizing a Q-learning mechanism similar to HARAQ [36, 108]. This comprises both minimal and non-minimal pathways, which are necessary for balancing the network traffic. To implement the learning approach in NoC, we must describe the cost functions that may target latency, power consumption, or temperature. This approach can be extended to regular and irregular NoCs.

4.3 Thermal- and Power-aware Routing Method

4.3.1 Design Challenges of Traffic- and Thermal-aware Routing

Based on the aforementioned analysis in Section B.2, both temporal and spatial DTM approaches have a negative impact on system performance owing to the scenario of heterogeneous temperature and traffic behavior in an NoC-based manycore system. Consequently, to integrate the benefits of temporal and spatial DTM techniques, information about traffic and temperature behavior must be examined simultaneously during NoC operation. To achieve these objectives, several researchers have proposed thermal-and traffic-aware packet routing algorithms in recent years [16, 65]. Generally, these routing approaches deliver packets away from thermally controlled NoC nodes depending on certain real-time information on the NoC system. Among them, [16] and [17] transported packets by referencing traffic information. Although the traffic distribution becomes balanced, the temperature distribution may become imbalanced. In contrast, some approaches [65, 76] consider only the temperature information while

Table 1: Full case of the relationship between temperature and traffic load information.

		Temperature Information	
		High	**Low**
Traffic Load	**Heavy**	Case 1: Consecutively heavy traffic load	**Case 2: Traffic block**
	Light	**Case 3: Highly historical traffic load**	Case 4: Small traffic load

delivering the packets, but they may result in heavy traffic congestion problems. This is because the rates of change for temperature and traffic are different (i.e., the rate of change in temperature is much slower than that of traffic).

Clearly, managing the traffic load and temperature distribution solely based on traffic or temperature information is inefficient. The scenarios resulting from various combinations of the temperature and traffic information are listed in the Appendix. Traditional traffic and thermal-aware adaptive routing methods assume that a heavy traffic load implies a high-temperature scenario and a light traffic load causes a low-temperature scenario. Nonetheless, temperature is a phenomenon of a protracted traffic scenario. In other words, the rates of change in temperature and traffic load are significantly different (i.e., the rate of change in temperature is low, and that of traffic is high). Therefore, the traditional approaches do not consider Cases 2 and 3 in Table 1, which are the design problems of the current traffic and temperature-aware routing approaches. Furthermore, the objective of this field's design is to synchronize the temperature and traffic information throughout the thermal control period, which aids in solving Cases 2 and 3 in Table 1.

4.3.2 Techniques of Thermal-aware Routing

Because of the thermal management involved, the topology of the NoC-based manycore system varies over time [19]. The time-varying topology change is defined as a nonstationary irregular mesh (NSI-mesh). The NSI-mesh topology may render the conventional routing method inefficient. This is because the packets may be blocked in the throttled router and cause significant traffic congestion in the NoC.

To deliver data successfully in the NSI-mesh, Chao *et al.* proposed the transport layer-assisted routing (TLAR) scheme [14]. Because the transport layer information includes the topology information from the source node to the destination node, TLAR is used to deliver packets based on the topology

information in the transport layer of NoC systems. Before injecting a packet into the network, the selected routing mechanism is determined based on the topology information of the route from the source node to the destination node. Thus, the packets are detoured from the inactive router in advance, which prevents the problem of traffic congestion surrounding the inactive routers. By extending the TLAR scheme, Chen *et al.* proposed a topology-aware adaptive routing (TAAR) method [21] to consider a cascaded routing mechanism, which increases routing flexibility. To increase the routing path diversity, Chen *et al.* further considered the non-minimal routing path and proposed traffic- and thermal-aware adaptive beltway routing (TTABR) to partially detour packets through non-minimal routing paths [17].

TLAR-based routing methods are used to deliver packets based on information regarding instant topology and traffic load on paths. In contrast, some routing methods have been proposed to consider the temperature information during packet delivery. Liu *et al.* [76] proposed a dynamic thermal-balancing routing algorithm and aimed to adaptively select the routing path to deliver packets based on the thermal conditions of neighboring nodes. Thus, the temperature distribution in the NoC-based many-core system can be balanced. In contrast, Kuo *et al.* proposed the use of information on the mean time to throttle (MTTT) of each NoC node to determine the routing paths and proposed a proactive thermal-budget-based beltway routing (PTB^3R) method [65].

Although the routing methods introduced above can mitigate the performance impact during the temperature control period, they assume that the information on temperature change and traffic load change is almost the same (i.e., they only consider Cases 1 and 4 in Table 1). However, as mentioned earlier, the rate of change in the temperature behavior is much slower than that of the traffic behavior. Therefore, these methods cannot process the scenarios of Cases 2 and 3 in Table 1 and still suffer from heavy traffic congestion. To solve this problem, Chen *et al.* applied game theory to propose a game-based thermal-delay-aware adaptive routing (GTDAR) mechanism. GTDAR employs a voting game model to perform dynamic buffer length adjustment to those based on current and historical temperature-traffic information. Thus, the long-term temperature information can be transferred to short-term traffic information, which is beneficial for routing path selection. To further balance the traffic load distribution, the authors further applied the Nash equilibrium to distribute the packet delivery. Thus, the GTDAR can consider the traffic and temperature information simultaneously to determine the proper routing direction and achieve a balanced traffic and temperature distribution.

4.4 Architectural-level Design to Mitigate Thermal and Power Problems

Because of its highly flexible interconnection and topology, NoC-based on-chip interconnection has become an emerging method for constructing manycore

systems. Conventional NoC systems are used to deliver packets via packet switching with routers. Regarding router design, designers frequently use several input buffers to leverage high-performance packet switching. Nevertheless, these buffers typically consume significant amounts of power and circuit areas [44]. As mentioned earlier, the power problem may eventually become a thermal one, which counteracts the benefits of flexible NoC interconnections.

Bufferless NoC designs have become a promising development trend for mitigating thermal and power problems. The design concept of bufferless NoCs is to leave buffers out of the traditional router architecture design. Because there is no buffer to store packets, we must guarantee that packet routing and flow control are both contention-free. Otherwise, a large number of packets may be dropped, resulting in dramatic computing accuracy loss. To solve this problem, many studies have utilized time division multiplexing (TDM) information to schedule packet routing to prevent packet contention. Mirza *et al.* [87] first analyzed the packet flow information between each task of a specific application. Subsequently, they scheduled packet flows to match the bandwidth requirements of the target application. However, this approach frequently requires a slot table to record the exact real-time bandwidth requirement, which results in a large-area overhead. In contrast, Picornell *et al.* considered the worst-case route to schedule packet routing, which aids in simplifying the complexity of TDM scheduling. However, the performance may be degraded owing to worst-case considerations [92].

Because the TDM-based bufferless NoC design suffers from large-area overhead and performance degradation, Venkataramani *et al.* proposed a bufferless software-defined NoC called *ASCENT* [107]. *ASCENT* adopts a synchronous dataflow (SDF) model of computation to represent the input streaming applications. Through an offline analysis, information on the task execution time and inter-task communication time is represented precisely in the SDFs. With precise timing information regarding task execution and communication, a deterministic schedule of the tasks and packets can be generated for runtime execution. At runtime, a finite-state machine (FSM) can be used to reconfigure the router based on schedule information. Because the *ASCENT* NoC uses a software scheduler to perform packet routing, the router hardware does not require specific logic circuits to compute routing and flow control, which increases the control flexibility and reduces the hardware overhead. Compared with the conventional TDM-based bufferless NoCs, the *ASCENT* NoC can improve the throughput with better hardware efficiency.

As mentioned earlier, because of the dramatically large off-chip memory size for the input image data, the system throughput and energy consumption are both dominated by off-chip memory access. Regarding contemporary machine-learning accelerator designs, minimizing the off-chip memory reduction is a design challenge, which would not only improve the system throughput but also reduce the system energy consumption [89]. By reusing the data through

the memory hierarchy, the characteristics of data locality are involved and data movement is reduced, thereby significantly reducing energy consumption [106]. In contrast, NoC interconnection has become a potential method to construct DNN accelerators and has gained much interest [20]. Through multicasting routing technology, designers can leverage input-reuse or weight-reuse methods on the NoC platform and significantly reduce the off-chip memory. Mirmahaleh *et al.* [86] proposed a method for distributing data on the NoC platform and considered the memory access mechanism in some modern DNN models, such as AlexNet, VGG, and GoogleNet. Chen *et al.* [22] proposed a weight-wise convolution processing mechanism and introduced a hybrid data-reuse method to support data-reuse and weight-reuse methods simultaneously. Moreover, the authors used a multicast routing method to significantly reduce off-chip memory access. Thus, the energy consumption of the data movement between the off-chip memory and DNN accelerator can be reduced.

5 Conclusion

Cross-level abstraction designs have become the leading paradigm for complexity-aware and power-aware VSPS designs. In addition to the traditional software/hardware co-design, algorithm/architecture co-design has become a mainstream system design methodology. Circuit/device co-exploration has recently gained popularity. As we move from circuits to systems, linear differential and difference equations are used to model lumped and logic circuits, respectively. They are now escalated to stochastic dataflow graphs in modeling cross-level systems, e.g., system-on-chip, edge, cloud, IoT, neuromorphic, and even quantum computing platforms.

We provide an overview of recent advances in learned image and video compression systems. Learned image compression is becoming more mature, with technologies converging to the VAE-based coding framework with hyperpriors. In contrast, learned video compression remains an active area of research. Conditional coding, which originates from traditional codecs and is widely used in many learned video compression systems, is emerging as an attractive solution to residual-based predictive coding. In particular, we profile the complexity of several learned video compression systems based on four intrinsic complexity metrics. The results reveal a growing trend in which more computations and/or higher buffering requirements are traded for higher compression performance. However, the caveat is that most learned video compression systems encounter practical problems. To determine an efficient microarchitecture design, we explore a design methodology to mitigate the performance, energy, and thermal impact of memory access and on-chip data delivery. This paper aims to open up research opportunities for algorithm/architecture co-design to enable low-complexity implementations.

Financial Support

This work was supported by the National Science and Technology Council (grant numbers 110-2221-E-110-026-MY3 and 109-2221-E-006-191-MY2).

Appendix

ACRONYMS	Full name
VSPS	visual signal processing system
AI	artificial intelligence
CPU	central processing unit
ASIC	application-specific integrated circuit
FPGA	field-programmable gate array
DSP	digital signal processor
GPU	graphics processing unit
AAC	algorithm/architecture co-exploration
TLM	transaction-level model
DFM	dataflow model
SHC	software/hardware co-design
VLSI	very large scale integration
CDC	circuit/device co-design
DFG	dataflow graph
CNN	convolutional neural network
DCT	discrete cosine transform
JPEG	Joint Photographic Experts Group
ITU	International Telecommunication Union
MPEG	Moving Picture Experts Group
VVC	versatile video coding
PSNR-RGB	peak-signal-to-noise-ratio-RGB
MS-SSIM	multi-scale structural similarity
VAE	variational autoencoder
HEVC	high-efficiency video coding
CLIC	Challenge on Learned Image Compression
CVPR	IEEE/CVF Computer Vision and Pattern Recognition Conference
ISCAS	IEEE International Symposium on Circuits and Systems
ANF	augmented normalizing flows
SSF	scale-space flow

ACRONYMS	Full name
DMOS	differential mean opinion score
CVAE	conditional variational autoencoder
BD	Bjøntegaard-delta
NoC	network-on-chip
DTM	dynamic thermal management
RDTM	reactive DTM
PDTM	proactive DTM
NSI-mesh	nonstationary irregular mesh
TLAR	transport layer-assisted routing
TAAR	topology-aware adaptive routing
TTABR	traffic- and thermal-aware adaptive beltway routing
MTTT	mean time to throttle
PTB^{3}R	proactive thermal-budget-based beltway routing
GTDAR	game-based thermal-delay-aware adaptive routing
TDM	time division multiplexing
SDF	synchronous dataflow
FSM	finite-state machine

References

[1] HM-16.20, https://vcgit.hhi.fraunhofer.de/jvet/HM/-/tree/HM-16.20.

[2] E. Agustsson, D. Minnen, N. Johnston, J. Ballé, S. J. Hwang, and G. Toderici, "Scale-Space Flow for End-to-End Optimized Video Compression," in *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[3] F. E. Allen, "Control flow analysis," *SIGPLAN Not.*, 5, 1970, 1–19.

[4] E. Alshina, A. Boev, Z. Cui, Y. Feng, G. Gaikov, T. Guo, S. Ikonin, P. Jia, A. Karabutov, A. B. Koyuncu, E. Koyuncu, M. Li, J. Mao, S. Qian, J. Sauer, Y. Shi, T. Solovyev, M. Sychev, J. Wang, D. Yu, and Y. Zha, "Presentation of the Huawei response to the JPEG AI Call for Proposals," in *ISO/IEC JTC 1/SC29/WG1 wg1m9605*, 2022.

[5] G. Ascia, V. Catania, M. Palesi, and D. Patti, "Implementation and analysis of a new selection strategy for adaptive routing in networks-on-chip," *IEEE Transactions on Computers*, 57, 2008, 809–20.

[6] J. Ballé, N. Johnston, and D. Minnen, "Integer Networks for Data Compression with Latent-variable Models," in *Proceedings of International Conference on Learning Representations*, 2019.

[7] J. Ballé, D. Minnen, S. Singh, S. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," in *Proceedings of International Conference on Learning Representations*, 2018.

[8] BPG, https://bellard.org/bpg.
[9] B. Bross, Y.-K. Wang, Y. Ye, S. Liu, J. Chen, G. J. Sullivan, and J.-R. Ohm, "Overview of the versatile video coding (VVC) standard and its applications," in *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.
[10] L. Cai and D. Gajski, "Transaction level modeling: an overview," in *The Proceedings of the 1st IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, Newport Beach, CA, USA, 2003.
[11] A. S. Cassidy and A. G. Andreou, "Beyond Amdahl's Law: An Objective Function That Links Multiprocessor Performance Gains to Delay and Energy," *IEEE Transactions on Computers*, 61(8), 2012, 1110–26.
[12] Y. Cengel and M. Boles, *Thermodynamics: An engineering approach*, 7th edition, McGraw Hill, 2011.
[13] C. Chao, K. Jheng, H. Wang, J. Wu, and A. Wu, "Traffic- and thermal-aware run-time thermal management scheme for 3D NoC systems," in *ACM/IEEE International Symposium on Networks-on-Chip (NOCS)*, 2010, 223–30.
[14] C.-H. Chao, K.-C. Chen, T.-C. Yin, S.-Y. Lin, and A.-Y. Wu, "Transport Layer Assisted Routing for Run-Time Thermal Management of 3D NoC Systems," *ACM Transactions on Embedded Computing Systems*, 13(1), 2013.
[15] K.-C. Chen *et al.*, "Thermal sensor allocation and full-system temperature characterization for thermal-aware mesh-based NoC system by using compressive sensing technique," in *International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, 1–4.
[16] K.-C. Chen *et al.*, "Topology-Aware Adaptive Routing for Non-Stationary Irregular Mesh in Throttled 3D NoC Systems," *IEEE Transactions on Parallel and Distributed Systems*, 24(10), 2013, 2109–20.
[17] K.-C. Chen *et al.*, "Traffic- and Thermal-aware Adaptive Beltway Routing for Three Dimensional Network-on-Chip Systems," in *IEEE International Symposium on Circuits and Systems (ISCAS'13)*, May 2013, 1660–3.
[18] K.-C. Chen, E.-J. Chang, H.-T. Li, and A.-Y. Wu, "RC-based Temperature Prediction Scheme for Proactive Dynamic Thermal Management on Throttle-based 3D NoCs," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 26(1), 201, 206–18.
[19] K.-C. Chen, C.-H. Chao, and A.-Y. Wu, "Thermal-aware 3D Network-on-Chip (3D NoC) Design: Routing Algorithms and Thermal Managements," *IEEE Circuits and Systems Magazine*, 15(4), 2015, 45–69.

[20] K.-C. Chen, M. Ebrahimi, T.-Y. Wang, and Y.-C. Yang, "NoC-based DNN Accelerator: A Future Design Paradigm," in *13th IEEE/ACM International Symposium on Networks-on-Chip (NOCS 2019)*, October 2019.

[21] K.-C. Chen, S.-Y. Lin, H.-S. Hung, and A.-Y. Wu, "Topology-Aware Adaptive Routing for Non-Stationary Irregular Mesh in Throttled 3D NoC Systems," *IEEE Transactions on Parallel and Distributed Systems*, 24(10), 2013, 2109–20.

[22] K.-C. Chen, Y.-C. Yang, and Y.-S. Liao, "An Arbitrary Kernel-Size Applicable NoC-Based DNN Processor Design with Hybrid Data Reuse," in *IEEE International Midwest Symposium on Circuits & Systems*, August 2021.

[23] S.-Y. Chen, G. G. C. Lee, T.-P. Wang, C.-W. Huang, J.-H. Chen, and C.-L. Tsai, "Reconfigurable Edge via Analytics Architecture," in *Proc. 2019 IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, IEEE, 2019, 117–21.

[24] T. Chen, H. Liu, Z. Ma, and Q. Shen, "End-to-End Learnt Image Compression via Non-Local Attention Optimization and Improved Context Modeling," in *IEEE Transactions on Image Processing*, 2021.

[25] W.-H. Chen, K.-X. Li, W.-Y. Lin, K.-H. Hsu, P.-Y. Li, C.-H. Yang, C.-X. Xue, E.-Y. Yang, Y.-K. Chen, Y.-S. Chang, T.-H. Hsu, Y.-C. King, C.-J. Lin, R.-S. Liu, C.-C. Hsieh, K.-T. Tang, and M.-F. Chang, "A 65 nm 1 Mb nonvolatile computing-in-memory ReRAM macro with sub-16ns multiply-and-accumulate for binary DNN AI edge processors," in *2018 IEEE International Solid - State Circuits Conference - (ISSCC)*, 2018, 494–6.

[26] Y.-H. Chen, T. Krishna, J. Emer, and V. Sze, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," in *IEEE International Solid-State Circuits Conference, ISSCC 2016, Digest of Technical Papers*, 2016, 262–3.

[27] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks," *IEEE Journal of Solid-State Circuits*, 52(1), 2017, 127–38.

[28] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, "Learned image compression with discretized gaussian mixture likelihoods and attention modules," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2020.

[29] Y. Choi, M. El-Khamy, and J. Lee, "Variable Rate Deep Image Compression With a Conditional Autoencoder," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, 3146–54.

[30] G. Chrysos, "Intel Xeon phi coprocessor-the architecture," 2014, Intel Whitepaper.

[31] CLIC at CVPR, http://compression.cc/.

[32] A. K. Coskun, T. S. Rosing, and K. C. Gross, "Proactive temperature balancing for low cost thermal management in MPSoCs," in *2008 IEEE/ACM International Conference on Computer-Aided Design*, 2008, 250–7.

[33] W. Ding *et al.*, "Compressive sensing based channel estimation for OFDM systems under long delay channels," *IEEE Trans. Broadcast.*, 60(2), 2014, 313–21.

[34] M. Ebrahimi and M. Daneshtalab, "A general methodology on designing acyclic channel dependency graphs in interconnection networks," *IEEE Micro*, 38, 2018, 79–85.

[35] M. Ebrahimi and M. Daneshtalab, "Ebda: A new theory on design and veri?cation of deadlock-free interconnection networks," in *ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA)*, 2017, 703–15.

[36] M. Ebrahimi, M. Daneshtalab, F. Farahnakian, J. Plosila, P. Liljeberg, M. Palesi, and H. Tenhunen, "Haraq: Congestion-aware learning model for highly adaptive routing algorithm in on-chip networks," in *IEEE/ACM Sixth International Symposium on Networks-on-Chip*, 2012, 19–26.

[37] S. Esenlik, Y. Li, Y. Wu, K. Zhang, L. Zhang, and Z. Zhang, "Bytedance's Response to the JPEG AI Call for Proposals," in *ISO/IEC JTC 1/SC29/WG1 wg1m9605*, 2022.

[38] F. Farahnakian, M. Ebrahimi, M. Daneshtalab, P. Liljeberg, and J. Plosila, "Bi-lcq: A low-weight clustering-based q-learning approach for NoCs," *Microprocessors and Microsystems*, 38, 2014, 64–75.

[39] R. Feng, Z. Guo, Z. Zhang, and Z. Chen, "Versatile Learned Video Compression," 2021, arXiv preprint arXiv:2111.03386.

[40] Final Call for Proposals for JPEG AI, https://ds.jpeg.org/documents/jpegai/wg1n100095-094-REQ-Final_Call_for_Proposals_for_JPEG_AI.pdf.

[41] A. H. Ghamarian, M. C. W. Geilen, *et al.*, "Throughput Analysis of Synchronous Data Flow Graphs," in *Application of Concurrency to System Design, 2006. ACSD 2006. Sixth International Conference*, 2006, 25–36.

[42] K. Gilles, "The semantics of a simple language for parallel programming," in *Information Processing'74: Proceedings of the IFIP Congress*, 1974, 471–5.

[43] Grand Challenge on Neural Network-based Video Coding at ISCAS, https://www.iscas2022.org/grand-challenge.

[44] P. Gratz, C. Kim, R. Mcdonald, S. W. Keckler, and D. Burger, "Implementation and evaluation of on-chip network architectures," in *2006 International Conference on Computer Design*, 2006.

[45] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, "EIE: Efficient Inference Engine on Compressed Deep Neural Network," in *Proceedings of the 43rd International Symposium on Computer Architecture*, Piscataway, NJ, USA, 2016, 243–54.

[46] D. He, Y. Zheng, B. Sun, Y. Wang, and H. Qin, "Checkerboard Context Model for Efficient Learned Image Compression," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[47] Y.-H. Ho, C.-C. Chan, W.-H. Peng, H.-M. Hang, and M. Domanski, "ANFIC: Image Compression Using Augmented Normalizing Flows," in *IEEE Open Journal of Circuits and Systems*, December 2021.

[48] Y.-H. Ho, C.-P. Chang, P.-Y. Chen, A. Gnutti, and W.-H. Peng, "CANF-VC: Conditional Augmented Normalizing Flows for Video Compression," in *Proceedings of European Conference on Computer Vision (ECCV)*, October 2022.

[49] Y. Hu, W. Yang, Z. Ma, and J. Liu, "Learning End-to-End Lossy Image Compression: A Benchmark," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[50] Y. Hu, W. Yang, Z. Ma, and J. Liu, "Learning End-to-End Lossy Image Compression: A Benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 44(8), 2022, 4194–211.

[51] Z. Hu, Z. Chen, D. Xu, G. Lu, W. Ouyang, and S. Gu, "Improving Deep Video Compression by Resolution-adaptive Flow Coding," in *Proceedings of European Conference on Computer Vision (ECCV)*, 2020.

[52] Z. Hu, G. Lu, J. Guo, S. Liu, W. Jiang, and D. Xu, "Coarse-to-fine Deep Video Coding with Hyperprior-guided Mode Prediction," in *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[53] Z. Hu, G. Lu, and D. Xu, "FVC: A New Framework towards Deep Video Compression in Feature Space," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[54] G. Huang and L. Wang, "High-Speed Signal Reconstruction with Orthogonal Matching Pursuit via Matrix Inversion Bypass," in *2012 IEEE Workshop on Signal Processing Systems, Quebec City, QC*, 2012, 191–6.

[55] IEEE ICIP, 2021, https://www.2021.ieeeicip.org/www.2021.ieeeicip.org/Tutorials.html.

[56] B. L. J. Li and Y. Lu, "Deep Contextual Video Compression," 2019, arXiv:2109.1504.

[57] C. Jia, X. Hang, S. Wang, Y. Wu, S. Ma, and W. Gao, "FPX-NIC: An FPGA-Accelerated 4K Ultra-high-definition Neural Video Coding System," in *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, 2022.

[58] JPEG-AI, https://jpeg.org/jpegai/.

[59] A. C. J. Kienhuis, "Design Space Exploration of Stream-based Dataflow Architectures," *PhD thesis*, 1999.

[60] J.-H. Kim, J.-H. Choi, J.-S. Lee, and J. Chang, "Efficient deep learning-based lossy image compression via asymmetric autoencoder and pruning," in *Proceedings of IEEE International Conference on Acoustics, Speech, & Signal Processing*, 2020.

[61] E. A. d. Kock, W. J. M. Smits, *et al.*, "YAPI: Application modeling for signal processing systems," in *presented at the Proceedings of the 37th Annual Design Automation Conference*, Los Angeles, California, USA, 2000.

[62] E. Kodak, "Kodak Lossless True Color Image Suite (Photocd PCD0992)," http://r0k.us/graphics/kodak/.

[63] H. Küflüoglu *et al.*, "Thermally-aware sensor allocation for real-time monitoring and mitigation of FEOL aging in System-on-Chip (SoC) applications," in *IEEE International Reliability Physics Symposium (IRPS)*, 2017, 4C-6.1–4C-6.5.

[64] H. T. Kung, B. McDanel, and S. Q. Zhang, "Mapping Systolic Arrays onto 3D Circuit Structures: Accelerating Convolutional Neural Network Inference," in *2018 IEEE International Workshop on Signal Processing Systems (SiPS)*, 2018, 330–6.

[65] C.-C. Kuo *et al.*, "Proactive Thermal-Budget-Based Beltway Routing Algorithm for Thermal-Aware 3D NoC Systems," in *Int'l Symp. System-on-Chip*, October 2013, 1–4.

[66] T. Ladune, P. Philippe, W. Hamidouche, L. Zhang, and O. Déforges, "Optical Flow and Mode Selecting for Learning-based Video Coding," in *Proceedings of IEEE International Workshop on Multimedia Signal Processing (MMSP)*, 2020.

[67] G. G. Lee, "Algorithm/Architecture Co-design for Smart Signals and Systems in Cognitive Cloud/Edge," *IEEE CASS Distinguished Lecture, in IEEE Circuitd and Systems Magazine*, 20(1), 2020.

[68] G. G. Lee, C.-F. Chen, *et al.*, "Algorithmic complexity analysis on data transfer rate and data storage for multidimensional signal processing," in *Signal Processing Systems (SiPS), 2013 IEEE Workshop, 2013*, 2013, 171–6.

[69] G. G. Lee, Y.-K. Chen, M. Mattavelli, and E. S. Jang, "Algorithm/Architecture Co-Exploration of Visual Computing on Emergent Platforms: Overview and Future Prospects," *IEEE Transactions on Circuits and Systems for Video Technology*, 19(11), 2009, 1576–87.

[70] G. G. Lee, H.-Y. Lin, *et al.*, "Quantifying Intrinsic Parallelism Using Linear Algebra for Algorithm/Architecture Coexploration," *Parallel and Distributed Systems, IEEE Transactions*, 23, 2012, 944–57.

[71] J. Lee, S. Cho, and S. Beack, "Context-adaptive entropy model for end-to-end optimized image compression," in *Proceedings of International Conference on Learning Representations (ICLR)*, 2019.

[72] J. Li, B. Li, and Y. Lu, "Hybrid Spatial-Temporal Entropy Modelling for Neural Video Compression," in *Proceedings of ACM International Conference on Multimedia*, 2022.

[73] C. Lin, J. Yao, F. Chen, and L. Wang, "A spatial RNN codec for end-to-end image compression," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2020.

[74] J. Lin, D. Liu, H. Li, and F. Wu, "M-LVC: Multiple Frames Prediction for Learned Video Compression," in *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, 3546–54.

[75] N. Ling, C.-C. Jay Kuo, G. J. Sullivan, D. Xu, S. Liu, H.-M. Hang, W.-H. Peng, and J. Liu, "The Future of Video Coding," in *APSIPA Transactions on Signal and Information Processing*, 2022.

[76] F. Liu, H. Gu, and Y. Yang, "DTBR: A dynamic thermal balancing routing algorithm for Network-on-Chip," *Journal of Computers and Electrical Engineering*, 38, 2012, 270–81.

[77] H. Liu, M. Lu, Z. Ma, F. Wang, Z. Xie, Y. Cao, and Y. Wang, "Neural Video Coding using Multiscale Motion Compensation and Spatiotemporal Context Model," in *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, 2020.

[78] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows," in *Proceedings of International Conference on Computer Vision*, 2021.

[79] J. Long *et al.*, "Thermal monitoring mechanisms for chip multiprocessors," *ACM Trans. Archit. Code Optim.*, 5(2), 2008, 1–33.

[80] 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 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, 11006–15.

[81] G. Lu, X. Zhang, W. Ouyang, L. Chen, Z. Gao, and D. Xu, "An end-to-end learning framework for video compression," in *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020.

[82] H. Ma, D. Liu, N. Yan, H. Li, and F. Wu, "End-to-End Optimized Versatile Image Compression With Wavelet-Like Transform," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 44(3), 2020, 1247–63.

[83] A. Mercat, M. Viitanen, and J. Vanne, "Uvg dataset: 50/120fps 4k sequences for video codec analysis and development," in *ACM Multimedia Systems Conference*, 2020.

[84] D. Minnen, J. Ballé, and G. Toderici, "Joint Autoregressive and Hierarchical Priors for Learned Image Compression," in *Proceedings of Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

[85] D. Minnen and S. Singh, "Channel-Wise Autoregressive Entropy Models for Learned Image Compression," in *Proceedings of IEEE International Conference on Image Processing*, 2020.

[86] S. Y. H. Mirmahaleh, M. Reshadi, H. Shabani, X. Guo, and N. Bagherzadeh, "Flow mapping and data distribution on mesh-based deep learning accelerator," in *Proc. 13th IEEE/ACM Int. Symp. Netw.-Chip*, October 2019, 1–8.

[87] U. M. Mirza, F. Gruian, and K. Kuchcinski, "Mapping streaming applications on multiprocessors with time-division-multiplexed network-on-chip," in *Computers and Electrical Engineering*, 2014.

[88] R. Mukherjee *et al.*, "Thermal Sensor Allocation and Placement for Reconfigurable Systems," in *IEEE Int. Conference on Computer-Aided Design*, 2006, 437–42.

[89] S. M. Nabavinejad, M. Baharloo, K.-C. Chen, M. Palesi, T. Kogel, and M. Ebrahimi, "An Overview of Efficient Interconnection Networks for Deep Neural Network Accelerators," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems (JETCAS)*, 10(3), 2020, 268–82.

[90] A. Parashar, M. Rhu, A. Mukkara, A. Puglielli, R. Venkatesan, B. Khailany, J. Emer, S. W. Keckler, and W. J. Dally, "SCNN: An Accelerator for Compressed-sparse Convolutional Neural Networks," in *Proceedings of the 44th Annual International Symposium on Computer Architecture*, New York, NY, USA, 2017, 27–40.

[91] K. K. Parhi, *VLSI digital signal processing systems: Design and implementation*, John Wiley & Sons, 2007.

[92] T. Picornell, J. Flich, C. Hernãndez, and J. Duato, "DCFNoC: A delayed conflict-free time division multiplexing network on chip," in *Proceedings Design Automation Conference*, 2019.

[93] Z. Qian, D. Juan, P. Bogdan, C. Tsui, D. Marculescu, and R. Marculescu, "SVR-NoC: A erformance analysis tool for network-on-chips using learning-based support vector regression model," in *Design, Automation Test in Europe Conference Exhibition (DATE)*, 2013, 354–7.

[94] A.-M. Rahmani, K. R. Vaddina, K. Latif, P. Liljeberg, J. Plosila, and H. Tenhunen, "Design and management of high-performance, reliable and thermal-aware 3D networks-on-chip," *IET Circuits, Devices & Systems*, 6(5), 2012, 308–21.

[95] J. Ranieri *et al.*, "Near-Optimal Thermal Monitoring Framework for Many-Core Systems-on-Chip," *IEEE Transactions on Computers*, 64(11), 2015, 3197–209.

[96] S. Reda *et al.*, "Improved Thermal Tracking for Processors Using Hard and Soft Sensor Allocation Techniques," *IEEE Transactions on Computers*, 60(6), 2011, 841–51.

[97] O. Rippel, A. G. Anderson, K. Tatwawadi, S. Nair, C. Lytle, and L. Bourdev, "ELF-VC: Efficient Learned Flexible-rate Video Coding," in *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, October 2021, 14479–88.

[98] L. Shang, L. Peh, A. Kumar, and N. K. Jha, "Thermal Modeling, Characterization and Management of On-Chip Networks," in *37th International Symposium on Microarchitecture (MICRO-37'04)*, 2004, 67–78.

[99] Y. Shen, M. Ferdman, and P. Milder, "Maximizing CNN Accelerator Efficiency Through Resource Partitioning," in *presented at the Proceedings of the 44th Annual International Symposium on Computer Architecture*, New York, NY, USA, 2017, 535–47.

[100] X. Sheng, J. Li, B. Li, L. Li, D. Liu, and Y. Lu, "Temporal Context Mining for Learned Video Compression," 2021, arXiv:2111.13850.

[101] J. Y. Shin *et al.*, "Thermal Sensor Allocation for SoCs Based on Temperature Gradients," in *Int. Symp. Quality Electronic Design (ISQED)*, March 2015, 29–34.

[102] M. Song, J. Choi, and B. Han, "Variable-rate deep image compression through spatially-adaptive feature transform," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, 2380–9.

[103] J. Staunstrup and W. Wolf, *Hardware/Software Co-Design: Principles and Practice*, Springer-Science+Business Media, 2002.

[104] H. Sun, Z. Cheng, M. Takeuchi, and J. Katto, "End-To-End Learned Image Compression With Fixed Point Weight Quantization," in *Proceedings of IEEE International Conference on Image Processing*, October 2020.

[105] H. Sun, Q. Yi, F. Lin, L. Yu, J. Katto, and M. Fujita, "F-LIC: FPGA-based Learned Image Compression with a Fine-grained Pipeline," in *Proceedings of IEEE Asian Solid-State Circuits Conference (A-SSCC)*, 2022.

[106] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proc. IEEE*, 105(12), 2017, 2295–329.

[107] V. Venkataramani, B. Bodin, A. K. Mohite, T. Mitra, and L.-S. Peh, "ASCENT: Communication Scheduling for SDF on Bufferless Software-defined NoC, early access," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, November 2021.

[108] A. Y. WeldeZion, M. Ebrahimi, M. Daneshtalab, and H. Tenhunen, "Automated power and latency management in heterogeneous 3D NoCs," in *8th International Workshop on Network on Chip Architectures (NoCArc)*, 2015, 33–8.

[109] N. Wirth, *Algorithms + Data Structures = Programs*, Prentice-Hall, 1976.

[110] Y. Wu, X. Li, Z. Zhang, X. Jin, and Z. Chen, "Learned Block-based Hybrid Image Compression," in *IEEE Transactions on Circuits and Systems for Video Technology*, 2022.

[111] x265, https://www.videolan.org/developers/x265.html.

[112] R. Yang, F. Mentzer, L. Van Gool, and R. Timofte, "Learning for Video Compression with Hierarchical Quality and Recurrent Enhancement," in *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[113] R. Yang, F. Mentzer, L. Van Gool, and R. Timofte, "Learning for Video Compression with Recurrent Auto-Encoder and Recurrent Probability Model," in *IEEE Journal of Selected Topics in Signal Processing (J-STSP)*, 2021.

[114] I. Yeo, C. C. Liu, and E. J. Kim, "Predictive dynamic thermal management for multicore systems," in *2008 45th ACM/IEEE Design Automation Conference*, 2008, 734–9.

[115] S. Yu, B. Gao, Z. Fang, H. Yu, J. Kang, and H. S. P. Wong, "A low energy oxide-based electronic synaptic device for neuromorphic visual systems with tolerance to device variation," *Advanced Materials*, 25(12), 2013, 1774–9.

[116] Y. Zhang *et al.*, "Chip level thermal profile estimation using on-chip temperature sensors," in *IEEE International Conference on Computer Design, Lake Tahoe*, 2008, 432–7.

[117] K. Zhao, J. Wang, and D. Zang, "A Convolution Neural Network Accelerator Based on NVDLA," in *Proceedings of the 5th International Conference on Algorithms, Computing and Systems*, September 2021, 43–7.

[118] H. Zheng and A. Louri, "An Energy-Efficient Network-on-Chip Design using Reinforcement Learning," in *2019 56th ACM/IEEE Design Automation Conference (DAC)*, 2019, 1–6.

[119] Z. Zheng, X. Wang, X. Lin, and S. Lv, "Get The Best of the Three Worlds: Real-Time Neural Image Compression in a Non-GPU Environment," in *Proceedings of ACM International Conference on Multimedia*, 2021.

[120] H. Zhou *et al.*, "An information-theoretic framework for optimal temperature sensor allocation and full-chip thermal monitoring," in *Design Automation Conference (DAC)*, 2012, 642–7.

[121] Y. Zhu, Y. Yang, and T. Cohen, "Transformer-based Transform Coding," in *Proceedings of International Conference on Learning Representations*, 2022.