# BaSIC: BayesNet Structure Learning for Computational Scalable Neural Image Compression

Yufeng Zhang[1], Hang Yu[2], Shizhan Liu[1], Wenrui Dai[1], and Weiyao Lin*[1]

[1] Shanghai Jiao Tong University, China .
{worldlife,shanluzuode,daiwenrui,wylin}@sjtu.edu.cn
[2] Ant Group, China. hyu.hugo@antgroup.com

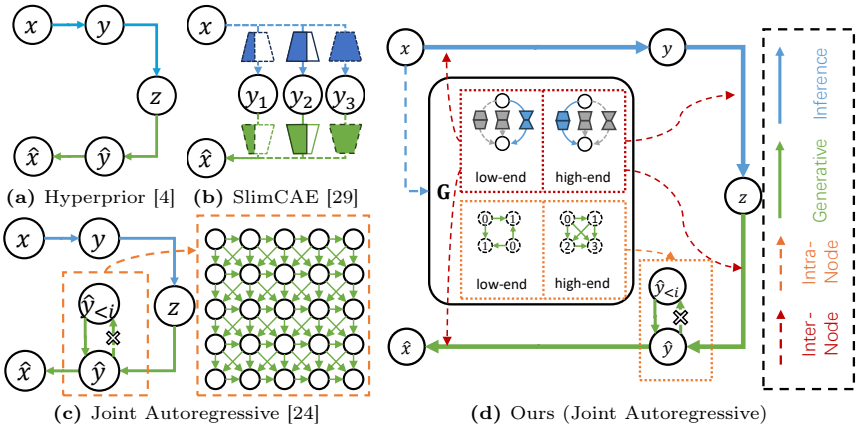
**Abstract.** Despite superior rate-distortion performance over traditional codecs, Neural Image Compression (NIC) is limited by its computational scalability in practical deployment. Prevailing research focuses on accelerating specific NIC modules but is restricted in controlling overall computational complexity. To this end, this work introduces BaSIC (BayesNet structure learning for computational Scalable neural Image Compression), a comprehensive, computationally scalable framework that affords full control over NIC processes. We learn the Bayesian network (BayesNet) structure of NIC for controlling both neural network backbones and autoregressive units. The learning of BayesNet is achieved by solving two sub-problems, i.e., learning a heterogeneous bipartite BayesNet for the inter-node structure to regulate backbone complexity, and a multipartite BayesNet for the intra-node structure to optimize parallel computation in autoregressive units. Experiments demonstrate that our method not only facilitates full computational scalability with more accurate complexity control but also maintains competitive compression performance compared to other computation scalable frameworks under equivalent computational constraints. Code will be available in https://github.com/worldlife123/cbench_BaSIC.

**Keywords:** Image Compression · Computational Scalable Compression · BayesNet Structure Learning · Autoregressive Model

## 1 Introduction

Neural Image Compression (NIC) technologies [7, 11] have significantly outstripped the performance of conventional image codecs, ushering in an era of intelligent image compression. These advanced codecs are intended for deployment across a spectrum of devices, from low-end devices like mobile phones to high-end devices like GPU computation clusters. Each type of device necessitates that codecs tailor their performance to specific computational or memory

* Corresponding Author.




**(a)** Hyperprior [4] **(b)** SlimCAE [29]

**(c)** Joint Autoregressive [24] **(d)** Ours (Joint Autoregressive)

**Fig. 1:** Bayesian network diagrams for various NIC frameworks. Our proposed framework BaSIC (d) generalizes the concepts in (a), (b), and (c).

constraints, a requirement known as **Computational Scalability**. High-end devices can endure more intricate computations for superior encoding quality, whereas low-end devices may prioritize reduced computational load over encoding efficiency to conserve energy.

While conventional image codecs typically accommodate this functionality (e.g. WebP [28] offers seven levels of computational complexity to regulate lossy image compression), incorporating this attribute into NIC remains a significant hurdle for its broader adoption. Recent studies have focused predominantly on accelerating NIC computations rather than addressing computational scalability. Works such as [11, 12, 25] target the acceleration of autoregressive modules on GPUs, which are frequently used in NIC. Others, including [15, 31, 38], explore neural network acceleration techniques—given that NIC fundamentally relies on neural network architectures. On the other hand, some recent efforts have attempted to integrate dynamic neural networks like slimmable networks [32] within NIC frameworks to adjust the computational complexity [27,29,36]. However, these approaches have only partially addressed computational scalability and often neglect crucial components in NIC, such as autoregressive modules, which are essential for compression performance.

In this work, we endeavor to create a NIC framework with enhanced computational scalability, incorporating computational control across all NIC components. We call this framework BaSIC (BayesNet structure learning for computational Scalable neural Image Compression). To this end, we re-examine NIC through the broader lens of the Bayesian network (BayesNet) paradigm. Figure 1 illustrates the BayesNet depiction of several examplary NIC architectures. Specifically, our approach is rooted in the hyperprior BayesNet framework (Figure 1a), which postulates that the generative model can be factorized as $p(\mathbf{x}|\mathbf{y})p(\mathbf{y}|\mathbf{z})p(\mathbf{z})$, where $\mathbf{x}$ represents observed data, $\mathbf{y}$ the prior, and $\mathbf{z}$ the hyperprior. In this context, SlimCAE (Figure 1b) can be interpreted as allowing

selective edge types between $\mathbf{y}$ and $\mathbf{x}$, and between $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$, with each type manifesting different conditional dependencies, modeled via slimmable neural networks with varied channel widths. On the other hand, the joint autoregressive framework (Figure 1c) introduces autoregressive intra-node dependencies within $\hat{y}$ to enhance compression, at the expense of increased computational demands due to the requisite sequential processing [12]. These examples clearly demonstrate the intimate connection between the structure of the BayesNet and the complexity inherent in NIC frameworks.

Our advancement is a NIC framework that exploits end-to-end BayesNet structure learning [23, 33] to meticulously optimize computational scalability. This innovative framework is not bound to specific neural network backbones, making it versatile enough to enhance computational scalability in most existing NIC frameworks with a hyperprior BayesNet representation. Our proposed framework is visualized in Figure 1d. In particular, we represent the BayesNet structure as $\mathbf{G}$ and impose a prior $p(\mathbf{G})$ on it. Given that the structure we seek to learn must conform to a directed acyclic graph (DAG), we tailor the prior $p(\mathbf{G})$ for compression frameworks within this DAG constraint, considering two distinct instances: the **inter-node BayesNet**, representing conditional dependencies between nodes, and the **intra-node BayesNet**, which pertains to dependencies within nodes. We depict the former as a heterogeneous bipartite graph and design a structure learning method to seamlessly incorporate dynamic neural networks like slimmable networks [32]. For the latter, we design a network architecture parameterized by multipartite or $k$-partite graphs to enable a parallel autoregressive process. By manipulating the partite number $k$, we can adjust the temporal demands of the autoregressive process, thereby directly impacting decompression speed.

When deploying the codec, the configurations of both the inter-node and intra-node BayesNets can be tailored to match the computational budget of target devices. For instance, on less capable devices, we might employ simpler neural network edges within the inter-node BayesNet and a smaller partite number within the intra-node BayesNet, whereas more capable devices can leverage a more complex network and a larger partite number, allowing for scalable computational resource allocation based on the specifics of the device at hand.

To summarize, our contributions are mainly three-fold:

- We propose an end-to-end computationally scalable NIC framework founded on BayesNet structure learning, facilitating fine-tuned adjustments to computational complexity across all facets of the NIC process.
- We devise a generative model based on a heterogeneous bipartite structure for the inter-node BayesNet, harnessing the capabilities of existing dynamic neural networks to realize computational scalability effectively.
- We design a generative model structured as a multipartite graph for the intra-node BayesNet. This model is tailored to accommodate and streamline parallel computation in practical applications.

## 2 Related Work

In this section, we briefly review recent works regarding the computational complexity of NIC as well as BayesNet structure learning.

### 2.1 End-to-end Neural Image Compression

The hyperprior framework [4] serves as a foundation for most end-to-end NIC models, favored for its effective compression performance and straightforward implementation. Subsequent studies [7, 19, 24] have enhanced the hyperprior by integrating a masked convolution-based autoregressive model in the latent space to reduce bit-rate. While these improvements offer better compression, they often overlook the increased computational demands. To apply NIC in practice, some recent works are now addressing the need to expedite these models.

**Autoregressive Model Acceleration** To circumvent the slow serial iteration steps of masked convolutions on GPUs, newer studies introduce parallelized autoregressive models for GPU acceleration. For instance, [25] presents a channel-wise parallel autoregressive model, while [12] proposes a spatially parallel model using a checkerboard mask, requiring just two parallel stages. This concept is extended to a multistage model by [21]. Further, [11] amalgamates channel-wise and spatially parallel autoregressive models. Additional acceleration methods include content-adaptive adjustments to parallel autoregressive models [35] and designing simplified autoregressive models [16]. Yet, such models are typically hand-crafted and may not represent the most efficient trade-off between compression complexity and performance.

**Neural Network Acceleration** Considering that neural networks (i.e., edges in the BayesNets) represent a significant portion of NIC's computational complexity, some works focus on accelerating these directly. For example, there are designs for more efficient neural network operators like a simplified GDN [15] and CPU-friendly down/up-samplers [38]. Others apply general neural network acceleration techniques within compression models, such as MorphNet [15] and learnable channel pruning [30, 31].

**Computation Scalable Framework** A handful of recent works have also noticed the importance of computational scalability for NIC. SlimCAE [29] employs slimmable networks [32] as the framework's backbone, enabling dynamic adjustments of bit-rate and computation through channel width modulation. Further developments include universal slimmable networks for more precise scalability [36] and the integration of input-adaptive complexity allocation methods [27,36]. Nonetheless, these approaches focus solely on the scalability of neural networks, largely disregarding the autoregressive models' scalability.

### 2.2 BayesNet Structure Learning

BayesNet structure learning aims to discern the conditional relationships among a set of variables with a graph representation, which ideally should be a directed acyclic graph (DAG). Traditional approaches often involve score-based [8, 34]

and constraint-based methods [6, 18]. However, they typically operate within a constrained search space, which can lead to less-than-optimal graphs. More recent initiatives introduce neural network techniques for more robust graph generation within the realm of BayesNet structure learning. Some align this learning task with variational inference to facilitate graph structure development within an end-to-end framework [23, 33], utilizing graph acyclicity regularizers [37] that may not guarantee acyclicity. Alternatively, other research employs recurrent-style graph generators that assure acyclicity through iterations [9], but these can become inefficient with an increase in the number of nodes. In our work, we diverge from these established methods due to their inefficiency when dealing with the dense pixelation of images. Instead, we analyze the BayesNet representations inherent in existing NIC frameworks and directly parameterize the graph structure as DAGs, ensuring both acyclicity and efficiency.

## 3 Overview of BaSIC

Assume we have a set of observations $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$[3], the objective of BayesNet structure learning is to infer a distribution over the structure of the BayesNet $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \boldsymbol{\theta})$ that best describes the observations. According to the Bayesian formalism, such optimization problem could be solved by minimizing expectation over $p(\mathbf{G}|\mathbf{X})$ of any function $f$ related to $\mathbf{G}$, under the constraint that $\mathbf{G}$ is acyclic:

$$\min_{\mathbf{G}} \mathbb{E}_{p(\mathbf{G}|\mathbf{X})}[f(\mathbf{G})], \quad \text{s.t. } \mathbf{G} \text{ is acyclic.} \tag{1}$$

For a computationally scalable compression framework, we tackle BayesNet structure learning, linking BayesNet configurations to computational complexity and shaping the function $f(\mathbf{G})$ accordingly.

Apart from modeling of $\mathbf{X}$, the compression framework also involves latent variables $\mathbf{L} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N, \mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N, \ldots\}$ to generate $\mathbf{X}$. The resulting BayesNet should capture conditional dependencies within and between $\mathbf{X}$ and $\mathbf{L}$. This generative relationship is described by:

$$p(\mathbf{G}, \mathbf{L}, \mathbf{X}) = p(\mathbf{G})p(\mathbf{L}|\mathbf{G})p(\mathbf{X}|\mathbf{G}, \mathbf{L}). \tag{2}$$

The expectation from Equation 1 is then:

$$\mathbb{E}_{p(\mathbf{G},\mathbf{L}|\mathbf{X})}[f(\mathbf{G}, \mathbf{L})] = \mathbb{E}_{p(\mathbf{G}|\mathbf{X})p(\mathbf{L}|\mathbf{X},\mathbf{G})}[f(\mathbf{G}, \mathbf{L})]. \tag{3}$$

Existing compression frameworks without $\mathbf{G}$ approximate $p(\mathbf{L}|\mathbf{X})$ with an inference network $q(\mathbf{L}|\mathbf{X})$, as in VAEs [17]. Incorporating $\mathbf{G}$ into our design, we refine our inference network to $q(\mathbf{G}, \mathbf{L}|\mathbf{X})$ for enhanced approximation:

$$\mathbb{E}_{p(\mathbf{G},\mathbf{L}|\mathbf{X})}[f(\mathbf{G}, \mathbf{L})] \approx \mathbb{E}_{q(\mathbf{G}|\mathbf{X})q(\mathbf{L}|\mathbf{X},\mathbf{G})}[f(\mathbf{G}, \mathbf{L})]. \tag{4}$$

[3] All symbols used in this paper are listed in Appendix.

This approach allows $\mathbf{G}$ to direct the computational complexity of both the inference and the generative processes. Furthermore, we adopt a differentiable approach to learning $\mathbf{G}$, in line with recent advances [23, 33], and incorporate this aspect into our model specification.

**Definition of $f(\mathbf{G}, \mathbf{L})$** In line with lossy image compression standards [15, 31], our optimization combines bit-rate $\mathcal{L}_R$, distortion $\mathcal{L}_D$, and complexity $\mathcal{L}_C$:

$$\mathbb{E}_{q(\mathbf{G}|\mathbf{X})q(\mathbf{L}|\mathbf{X},\mathbf{G})}[f(\mathbf{G},\mathbf{L})] = \mathcal{L}_R(\mathbf{G},\mathbf{L}) + \lambda_D\mathcal{L}_D(\mathbf{G},\mathbf{L}) + \lambda_C\mathcal{L}_C(\mathbf{G},\mathbf{L}). \quad (5)$$

Concretely, $\mathcal{L}_R$ is computed by enacting the BayesNet's generative process with latent node samples $\hat{\mathbf{L}}$:

$$\mathcal{L}_R(\mathbf{G},\mathbf{L}) = \log_2 p_{\mathbf{L}|\mathbf{G}}(\hat{\mathbf{L}}|\mathbf{G}), \quad \hat{\mathbf{L}} \sim q(\mathbf{L}|\mathbf{X},\mathbf{G}) \quad (6)$$

Moreover, $\mathcal{L}_D$ and $\mathcal{L}_C$ can be tailored to specific optimization objectives. $\mathcal{L}_D$ may encompass a range of complexity measures or their amalgamations, such as conventional metrics like Mean-Squared Error (MSE) or more advanced machine learning metrics like Frechet Inception Distance (FID) [13]. Conversely, $\mathcal{L}_C$ can include various complexity metrics or their combinations, such as the number of Floating-point Operations (FLOPs), network parameters, time, or energy consumption. $\lambda_D$ and $\lambda_C$ are the corresponding weights. By setting $f(\mathbf{G}, \mathbf{L})$ in line with the established rate-distortion (R-D) trade-off, Equation 4 falls in step with loss functions prevalent in standard compression frameworks.

**Definition of $q(\mathbf{G}|\mathbf{X})$** As a determiner of the BayesNet structure, $\mathbf{G}$ affects computational complexity. In practical compression scenarios, including $\mathbf{G}$ in the bitstream could impact the bit-rate. Typically, it may be more advantageous to treat $\mathbf{G}$ as independent of $\mathbf{X}$, suggesting $q(\mathbf{G}|\mathbf{X}) = p(\mathbf{G})$. However, in scenarios where complexity is tailored to input data, the model $q(\mathbf{G}|\mathbf{X})$ can be employed.

**DAG Constraint of $p(\mathbf{G})$** As previously noted in Equation 1, $p(\mathbf{G})$ should reflect a distribution over DAG structures. For edges between different node groups in compression models, such as $p(\mathbf{x}|\mathbf{y})$ or $p(\mathbf{y}|\mathbf{z})$, ensuring a DAG structure is straightforward since bipartite graphs are inherently acyclic; we call this component the **Inter-node BayesNet** ($\mathbf{G}_{inter}$). For conditional dependencies within node groups, like $p(\mathrm{y}_i|\mathrm{y}_{i-1})$ which also require DAG adherence, we refer to it as the **Intra-node BayesNet** ($\mathbf{G}_{intra}$). The forthcoming sections will delve into the methodology for learning both $p(\mathbf{G}_{inter})$ and $p(\mathbf{G}_{intra})$ as DAGs.

## 4 Structure Learning for Inter and Intra-Node BayesNets

### 4.1 Heterogeneous Bipartite Graphs for Inter-node BayesNet

In this section, we specify $p(\mathbf{G}_{inter})$. Note that $\mathbf{G}_{inter}$ is a bipartite graph, and so it inherently satisfies the acyclic constraint.

**Naive Bipartite Parameterization of $p(\mathbf{G}_{inter})$** Consider the task of learning a bipartite graph for $p(\mathbf{y}|\mathbf{z})$. We split the nodes into $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_I$ and $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_J$. Bipartite graphs can be represented by an $I \times J$ adjacency matrix

with each element indicating an edge between $\boldsymbol{y}$ and $\boldsymbol{z}$. Consequently, $\mathbf{G}_{inter}$ can be modeled as $I \times J$ Bernoulli distributions:

$$p(\mathbf{G}_{inter}^{\mathbf{y},\mathbf{z}}) = \text{Bernoulli}(\pi_{\mathbf{y},\mathbf{z}}), \quad \pi_{\mathbf{y},\mathbf{z}} \in \mathbb{R}^{I \times J}. \tag{7}$$

The edge $p(\mathbf{y}|\mathbf{z}, \mathbf{G}_{inter}^{\mathbf{y},\mathbf{z}})$ is then defined as:

$$p(\mathbf{y}|\mathbf{z}, \mathbf{G}_{inter}^{\mathbf{y},\mathbf{z}}) = \prod_{i,j}^{I,J} p(\mathbf{y}_j|\mathbf{z}_i)\hat{\boldsymbol{G}}_{inter}^{\mathbf{y},\mathbf{z}}[i,j], \tag{8}$$

where $\hat{\boldsymbol{G}}_{inter}^{\mathbf{y},\mathbf{z}}$ are samples from $p(\mathbf{G}_{inter}^{\mathbf{y},\mathbf{z}})$. Nonetheless, implementing such a BayesNet can present challenges. For instance, normalization layers such as GDN [3], often integral to compression models, are difficult to integrate because the nodes are presumed to be mutually independent.

**Heterogeneous Parameterization of** $p(\mathbf{G}_{inter})$ We offer an innovative parameterization for the bipartite graph, tailored for heterogeneous DAG structure learning within image compression frameworks. Rather than dividing nodes, we maintain individual nodes in adjacent layers and diversify only the connecting edges. Each node pair is linked through multiple neural network edges of varying sizes and complexities, from which only one is selected. In this context, $\mathbf{G}_{inter}$ is characterized by $N$ categorical distributions:

$$p(\mathbf{G}_{inter}^{\mathbf{y},\mathbf{z}}) = \text{Categorical}(\pi_{\mathbf{y},\mathbf{z}}), \quad \pi_{\mathbf{y},\mathbf{z}} \in \mathbb{R}^{N}. \tag{9}$$

Thus, the connection $p(\mathbf{y}|\mathbf{z}, \mathbf{G}_{inter}^{\mathbf{y},\mathbf{z}})$ can be expressed as:

$$p(\mathbf{y}|\mathbf{z}, \mathbf{G}_{inter}^{\mathbf{y},\mathbf{z}}) = \sum_{n}^{N} \hat{\boldsymbol{G}}_{inter}^{\mathbf{y},\mathbf{z}}[n] p_n(\mathbf{y}|\mathbf{z}). \tag{10}$$

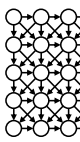
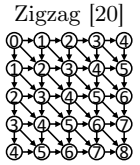
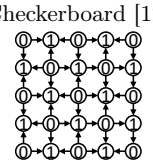
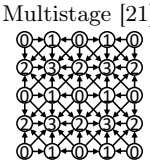
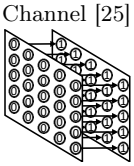
This heterogeneous DAG learning approach does not merely determine **whether** nodes are interdependent; it specifies **how** they are interconnected. Given the inherent dependencies among latent nodes in compression frameworks, this nuanced parameterization is particularly apt for representing inter-node BayesNets.

**Implementation of the edges** Once $\mathbf{G}_{inter}$ is established, we proceed to construct the heterogeneous edges, represented as $p_n(\mathbf{y}|\mathbf{z})$, utilizing neural networks. Following [29], we employ slimmable networks and select the $n$-th channel width configuration for each $p_n(\mathbf{y}|\mathbf{z})$. This approach is memory-efficient as it does not necessitate additional memory overhead compared to a singular network implementation for $p(\mathbf{y}|\mathbf{z})$. A comprehensive example detailing this implementation is provided in Section 5 for further reference.

**Optimization of** $p(\mathbf{G}_{inter})$ We can leverage samples from $p(\mathbf{G}_{inter})$ as weights to blend the various edges, enabling end-to-end optimization via the Gumbel-softmax reparameterization trick [14], eliminating the need for an additional loss function. The computational complexity associated with this optimization can be quantified using the same mixing weights:

$$\mathcal{L}_C(\mathbf{G}_{inter}^{\mathbf{y},\mathbf{z}}) = \sum_{n}^{N} \hat{\boldsymbol{G}}_{inter}^{\mathbf{y},\mathbf{z}}[n] C(p_n(\mathbf{y}|\mathbf{z})), \tag{11}$$

**Table 1:** Statistics of different intra-node BayesNet configurations for a $3 \times 32 \times 32$ tensor. Numbers inside the nodes represent the topological indices of the partites. For Maskconv each partite only include one node, so we omit the topological index.

| | MaskConv [24] | Zigzag [20] | Checkerboard [12] | Multistage [21] | Channel [25] |
|---|---|---|---|---|---|
| |  |  |  |  |  |
| Stages | 1024 | 63 | 2 | 4 | 3 |
| FLOPs | 11718 | 8835 | 2976 | 11718 | 2048 |
| Time (ms) | 11783.0 | 727.3 | 46.5 | 82.6 | 71.2 |

where $C(p_m(\mathbf{y}|\mathbf{z}))$ represents the complexity measure for a single edge such as FLOPs.

**Relation to Neural Architecture Search (NAS)** Our approach to parameterizing $p(\mathbf{G}_{inter})$ bears resemblance to Differentiable Architecture Search [22] in both concept and optimization techniques. Both methods utilize continuous relaxation to combine outputs from diverse neural network modules. However, the underlying objectives differ. NAS focuses on developing neural networks with enhanced performance by refining the computational graph. In contrast, our objective is to construct optimized probabilistic graphs tailored for compression, with neural networks serving as approximators of the conditional probabilities.

### 4.2 Multipartite Graph for Intra-node BayesNet

As noted in Section 2.2, traditional BayesNet structure learning has been constrained by the need to maintain a DAG, which, when applied to image data with its vast number of pixels, can greatly reduce efficiency. Table 1 illuminates an alternative approach that can satisfy both acyclic and parallel constraints effectively. It reveals that the number of parallel stages corresponds to the multipartite quantity within each BayesNet. Since edges can only direct from topologically earlier nodes to later ones, nodes within the same partite can be processed simultaneously. Bayesian inference can thus be executed by sequentially processing each partite from the first to the last. This process not only maintains acyclicity but also confines the number of parallel stages to the number of partites. Hence, we propose to parameterize $p(\mathbf{G}_{intra})$ with categorical distributions at each node, each category corresponding to the node's topological index within the partites. Consequently, $p(\mathbf{G}_{intra})$ takes the following form:

$$p(\mathbf{G}_{intra}) = \prod_{c,h,w}^{C,H,W} p(\mathrm{T}_{c,h,w}), \quad \mathrm{T}_{c,h,w} = \mathrm{Categorical}(\pi_{c,h,w}), \tag{12}$$

where $\mathrm{T}_{c,h,w}$ denotes the topological indices of partites per categorical distribution, with $C, H, W$ being the node count along the channel, height, and width

dimensions, respectively. Adjusting the number of partites (or parallel stages) $\dim \pi_{c,h,w} = S_{intra}$ can modulate the computation time.

**Optimization of $p(\mathbf{G}_{intra})$** Having constrained the number of partites to manage computational time, we now focus on learning the intra-node BayesNet structure to minimize bit-rate. The optimization challenge is formulated as:

$$\min_{p(\mathbf{G}_{intra})} \mathbb{E}_{p(\mathbf{G}_{intra})} p(\mathbf{X}, \mathbf{L}|\mathbf{G}_{intra}), \quad \text{s.t. } \hat{\mathbf{G}}_{intra} \text{ is acyclic}, \tag{13}$$

where $\hat{\mathbf{G}}_{intra}$ is sampled from $p(\mathbf{G}_{intra})$. However, unlike the optimization for $\mathbf{G}_{inter}$, partite indices are ordinal and don't lend themselves to being used as mixing weights, making the Gumbel-softmax reparameterization trick inapplicable. We therefore approach $\mathbf{G}_{intra}$ as discrete variables and employ the Monte-Carlo method with the VIMCO objective [26] for optimization:

$$\mathcal{L}_{\mathbf{G}_{intra}} = \mathbb{E}_{q(\mathbf{G}^{1:M}_{intra}|\mathbf{X})} \log \frac{1}{M} \sum_{i=1}^{M} \log p(\mathbf{X}, \mathbf{L}|\mathbf{G}^{i}_{intra}), \tag{14}$$

where $\mathbf{G}^{i}_{intra}$ is the i-th Monte-Carlo sample and $M$ denotes number of samples.

Moreover, note that Equation 12 presumes all partites are independent—an unlikely scenario as proximal nodes often exhibit stronger correlations. In practice, we can approximate $p(\mathbf{G}_{intra})$ with an unconditional generative model to consider such correlations and optimize this proxy instead, described as:

$$p(\mathbf{G}_{intra}) \propto g(\mathbf{N}), \quad \mathbf{N} \sim \mathcal{N}(0, 1), \tag{15}$$

with $g(\mathbf{N})$ symbolizing a generative network derived from standard normal noise.

**Implementation of the Intra-node BayesNet** With the model $p(\mathbf{G}_{intra})$ established, we can draw samples of topological indices for partites, denoted $\hat{T}_{c,h,w}$, from each respective $\mathrm{T}_{c,h,w}$. The subsequent step involves establishing conditional dependencies among these groups. Since dependencies are restricted to nodes from different partites, a dynamic masked convolution operation is utilized. This operation ensures each node's dependencies are limited to preceding nodes within the convolutional kernel's range. For instance, the intra-node BayesNet for $\mathbf{y}$ can be characterized by:

$$\prod_{c,h,w}^{C,H,W} p(\mathrm{y}_{c,h,w}|\mathbf{G}_{intra}) \propto \mathrm{dynconv}(\hat{\mathrm{y}}_{c,h,w}, \mathit{W}_{c,h,w}(\hat{T}_{c,h,w})), \tag{16}$$

where $\hat{\mathrm{y}}_{c,h,w}$ are samples from $\mathrm{y}_{c,h,w}$, and $\mathit{W}_{c,h,w}$ represents a dynamically adjusted convolutional kernel influenced by adjacent nodes' topological indices. As an illustration, consider a 2D $5 \times 5$ convolution: the dynamic kernel $\mathit{W}_{c,h,w}$ used for node $\mathrm{y}_{c,h,w}$ is defined by:

$$\mathit{W}_{c,h,w}[i, j+2, k+2] = \begin{cases} \mathit{W}_{i,j+2,k+2} & , \hat{T}_{c,h,w} < \hat{T}_{i,h+j,w+k} \\ 0 & , \hat{T}_{c,h,w} \geq \hat{T}_{i,h+j,w+k} \end{cases}, \begin{cases} i & \in \{0, 1, \ldots, C-1\} \\ j, k & \in \{-2, -1, 0, 1, 2\} \end{cases}. \tag{17}$$

For in-depth guidance of its implemention, please refer to Appendix.

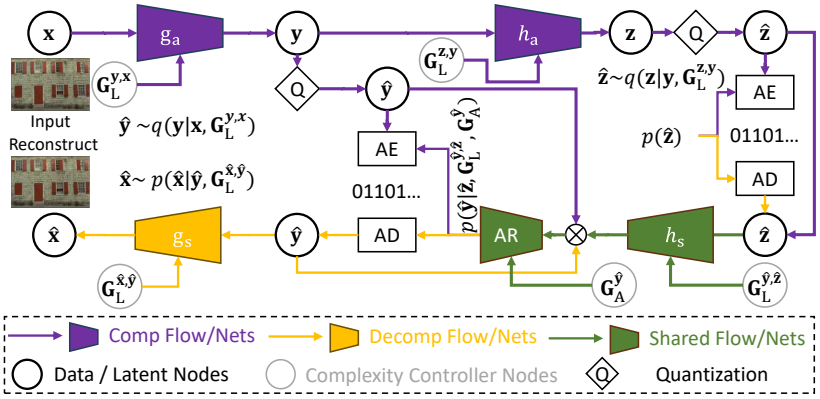



**Fig. 2:** The architecture of the implemented example based on the joint autoregressive framework [24]. Comp/Decomp means Compression / Decompression. AE/AD symbolize the arithmetic encoder and decoder, respectively. AR denotes the autoregressive model that executes the learned intra-node BayesNet. Q stands for quantization. $\boldsymbol{x}$ correspond to input image and $\hat{\boldsymbol{x}}$ the reconstructed image.

## 5 From BayesNet to BaSIC

Upon determining the BayesNet structure, we proceed to specify the nodes and edges in this BayesNet, crafting a novel NIC framework, BaSIC. This section details an implementation instance of the envisioned computationally scalable NIC, leveraging the prevalent joint autoregressive and hyperprior compression model [24]. It should be noted that this represents just one of the viable strategies for implementing BaSIC, with the potential for alternatives to offer improved results. The architecture of our example is depicted in Figure 2, and we will explore how it seamlessly integrates with the aforementioned BayesNets.[4]

**Inter-node BayesNet** The inter-node BayesNet is constructed over each edge between nodes within the original BayesNet, specifically:

$$\begin{aligned} &q(\mathbf{y}|\mathbf{x}, \mathbf{G}_{inter}^{\mathbf{y},\mathbf{x}}) \propto g_a(\mathbf{x}, \mathbf{G}_{inter}^{\mathbf{y},\mathbf{x}}), \quad q(\mathbf{z}|\mathbf{y}, \mathbf{G}_{inter}^{\mathbf{z},\mathbf{y}}) \propto h_a(\mathbf{y}, \mathbf{G}_{inter}^{\mathbf{z},\mathbf{y}}), \\ &p(\hat{\mathbf{y}}|\hat{\mathbf{z}}, \mathbf{G}_{inter}^{\hat{\mathbf{y}},\hat{\mathbf{z}}}) \propto h_s(\hat{\mathbf{z}}, \mathbf{G}_{inter}^{\hat{\mathbf{y}},\hat{\mathbf{z}}}), \quad p(\hat{\mathbf{x}}|\hat{\mathbf{y}}, \mathbf{G}_{inter}^{\hat{\mathbf{x}},\hat{\mathbf{y}}}) \propto g_s(\hat{\mathbf{y}}, \mathbf{G}_{inter}^{\hat{\mathbf{x}},\hat{\mathbf{y}}}) \end{aligned} \tag{18}$$

To achieve computational scalability, heterogeneous bipartite graphs are employed as outlined in Equation 10 for $\mathbf{G}_{inter}^{\mathbf{y},\mathbf{x}}$, $\mathbf{G}_{inter}^{\mathbf{z},\mathbf{y}}$, $\mathbf{G}_{inter}^{\hat{\mathbf{y}},\hat{\mathbf{z}}}$, and $\mathbf{G}_{inter}^{\hat{\mathbf{x}},\hat{\mathbf{y}}}$. The implementation of $g_a, g_s, h_a$, and $h_s$ follow [29]'s slimmable networks, with dynamic channel widths including 48, 72, 96, 144, and 192 for computational scalability. Note that we use a fixed 192 channels for all latent variables $\mathbf{y}, \mathbf{z}$, and only channels in the middle layers of each slimmable network are variable. This modification decouples bitrate scalability from computational scalability, enhancing the convenience in real-world applications.

[4] Further details are provided in Appendix.

**Intra-node BayesNet** In the established framework, the intra-node BayesNet employs masked convolution applied to $\hat{\mathbf{y}}$. We substitute this with our suggested multipartite graph-based intra-node BayesNet $p(\hat{\mathbf{y}}|\mathbf{G}^{\hat{\mathbf{y}}}_{intra})$, which enables parallel computing on GPUs. The intra-node BayesNet uses a $5 \times 5$ dynamic masked convolution as detailed in Equation 17, with $C$ equals to number of partites and $H, W$ equals to image height and width, respectively. Note that to facilitate the optimization of the intra-node BayesNet, the additional VIMCO loss expressed in Equation 14 should be incorporated into the overall loss function.

**Optimization** The optimization process involves a loss function that simultaneously accounts for both the BayesNet structure (Equation 14) and the neural network parameters (Equation 4):

$$\mathcal{L} = \mathcal{L}_R + \lambda_D \mathcal{L}_D + \lambda_C \mathcal{L}_C + \mathcal{L}_{\mathbf{G}_{intra}} \tag{19}$$

**Computational Scalability** The described implementation enables computational scalability across two dimensions. Firstly, by modulating all $\mathbf{G}_{inter}$, which govern the channel width in slimmable networks, we can fine-tune the computational load of the inter-node neural networks. $\mathbf{G}_{inter}$ can be modulated during training through the $\lambda_C$ parameter, or during inference by selecting from $\mathbf{G}_{inter}$ to alter channel widths. Notably, the latter can be streamlined by training with a uniform $\mathbf{G}_{inter}$ distribution and setting $\lambda_C = 0$. Secondly, by varying the number of partites in $\mathbf{G}^{\hat{\mathbf{y}}}_{intra}$, we can adjust the stages of parallel computations in autoregressive models, thereby influencing the decompression time. In practice, we could train multiple autoregressive models with varying stages and choose among them during inference to modify decompression times.

## 6 Experiments

### 6.1 Experiment Setup

**Experiment Tasks** Following previous works [29], our experiments focus on the task of lossy image compression with computational scalability. We benchmark our BaSIC framework against other computational scalable frameworks, as well as conduct ablation studies to validate the efficacy of the proposed intra-node BayesNet learning methods. Further experimental results and details can be found in Appendix , including compression performance evaluation and comparison under fixed computational constraints and more ablation studies regarding the inter-node BayesNet. For evaluation, we employ Pixel Signal-Noise Ratio (PSNR) for distortion and Bits Per Pixel (BPP) for bit-rate, unless stated otherwise.

**Datasets** We use a merged training dataset from ImageNet [10] and CLIC [2], preprocessed following the method in [11]. The Kodak [1] dataset serves for validation and testing.

**Implementation Details** For the proposed method, we use the joint autoregressive framework [24] as the basis, as described in Section 5. Unless otherwise

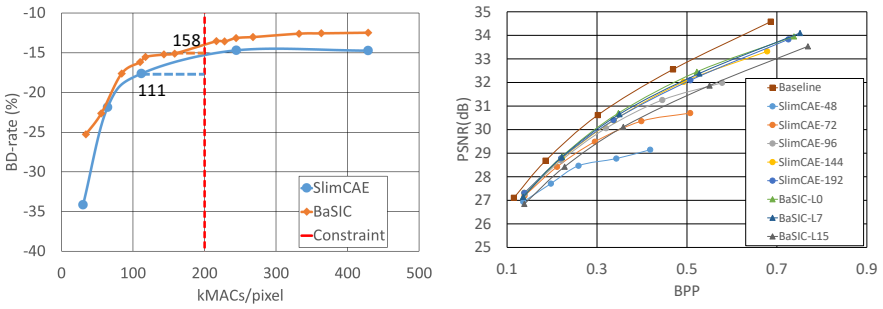



**Fig. 3:** Comparative results for different computational scalable frameworks.

specified, we follow CompressAI [5] to set the hyperparameter $\lambda_D$, while employing $\lambda_C = 0$ during training and a greedy search among 16 intra-node BayesNets with varied FLOPs during inference. Training spans approximately 500k iterations, employing the Adam optimizer with a learning rate of 0.0001. To compare running times equitably, all experiments are performed on identical hardware (an Intel i7-6800K CPU, NVIDIA Titan Xp GPU, and 32GB of memory).

### 6.2 Comparison to Other Computation Scalable Frameworks

This section assesses our computational scalable BaSIC method against other scalable frameworks. We configure our framework without autoregressive modules, similar to SlimCAE [29], to compare performance at various computational levels. We adopt Hyperprior [4] as a reference, reporting BD-rate for diverse MACs for both methods. Also, the rate-distortion plots for SlimCAE's full range of complexity and three levels (L0, L7, L15) for our method are included for reference. Note that $\lambda$-scheduling is deactivated in SlimCAE to facilitate uniform BD-rate analysis across levels. The results are depicted in Figure 3.

Our method, employing the same slimmable network backbone as SlimCAE, demonstrates a marginal performance edge at most computation levels, likely attributable to the fixed channels of latent variables that enhance higher bit-rate performance. Specifically, our method saves around 2% more bits than SlimCAE under the maximum complexity, and around 10% more bits under the minimum complexity. While SlimCAE [29] employs $\lambda$-scheduling to tackle this, it inadvertently ties bit-rate and distortion to computational complexity, which is less desirable for cases like low-powered devices needing high-quality images. In contrast, our inter-node BayesNet independently modulates the computational complexity of individual neural network modules, as evidenced by the rate-distortion plots that show minimal impact on performance with reduced computation. Moreover, the varied complexity levels indicate our method's superior control over computational complexity, allowing us to assign different inter-node BayesNets (channel widths) to distinct layers. SlimCAE, conversely, applies the same channel width across all layers, limiting flexibility. For instance,

**Table 2:** Bit-rate comparison for different autoregressive modules.

| Stages | - | | 2 | | | 4 | | 10 | |
|---|---|---|---|---|---|---|---|---|---|
| Methods | MaskConv | Baseline | CB | CW | Ours | MS | Ours | ELIC | Ours |
| BPP | 0.3150 | 0.3333 | 0.3216 | 0.3222 | **0.3207** | 0.3210 | **0.3183** | 0.3174 | **0.3156** |
| Speed (MB/s) | 0.32 | 345.55 | 191.23 | 173.46 | 172.57 | 107.93 | 74.67 | 17.85 | 17.32 |

to meet a 200kMAC/pixel threshold, our closest model uses 158kMAC/pixel, while SlimCAE's nearest option drops to 111kMAC/pixel, as shown in Figure 3.

### 6.3 Comparison to Other Autoregressive Models for Intra-Node BayesNet

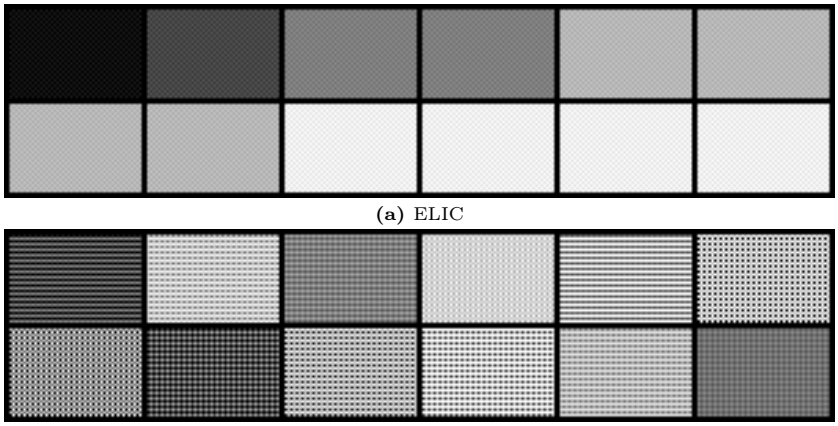
In this section, we evaluate the effectiveness of the proposed intra-node BayesNet structure learning method, by comparing it to other hand-crafted autoregressive models. For a simple and fair comparison, we first establish a baseline using a joint autoregressive framework [24] with a null context model, trained for about 500k iterations. Subsequently, we retain all other components while training various autoregressive models to replace the baseline context model for an additional 500k iterations. This approach ensures the reconstructed image remains unchanged, allowing us to isolate and evaluate the bit-rate to reflect the effectiveness of different autoregressive models. Moreover, the decompression speed is also reported to evaluate the efficiency of It's important to note that the neural network architecture for the context model is kept constant across all methods, with the sole variability stemming from the structure of the intra-node BayesNet.

The benchmark autoregressive models include Channel-wise (CW, 2-stage) [25], Checkerboard [12] (CB, 2-stage), Multi-stage spatial 2x2 [21] (MS, 4-stage) and the combined unevenly grouped channel-wise and checkerboard [11] (ELIC, 10-stage in total). We also train the MaskConv context model from [24] as a baseline. Our model is trained under constraints of 2, 4, and 10 parallel stages and evaluated against counterparts with equivalent stage configurations. The findings are presented in Table 2.

Overall, our learning-based BayesNet structures consistently demonstrate slightly better performance compared to the manually-crafted models under the same parallel stages, with similar decompression speed. Interestingly, in each group our learned method reaches close BPP to the hand-crafted ones in the next level of parallel stage, and in the 10-stage case, our method even approaches the BPP of non-parallel MaskConv model. This proves that our proposed method excels in learning effective intra-node BayesNets for autoregressive models.

**Qualitative Results** We also illustrate the learned intra-node BayesNet by our method as well as the manually-designed one of ELIC in Figure 4 to analyze their difference with hand-crafted models. [5] We could see that the BayesNet of ELIC uses a checkerboard context model over the spatial dimension, including only two

[5] More visualization results are shown in Appendix.



**(a)** ELIC

**(b)** Ours-10stage

**Fig. 4:** Partite topological indices visualization for intra-node BayesNet in the 10-stage case. Each image patch indicates topological indices of nodes in a group of channels. Blacker pixels indicate lower topological indices and whiter indicate higher topological indices. E.g., in the 10-stage case, pure-black indicates index 0 which is first iterated, and pure-white indicates group 9 which is last iterated. Better viewed with zoom-in.

partites within each channel. In comparison, our learned BayesNet incorporates more different partites (mostly three or four) in the spatial dimension, which captures more context information than the checkerboard case. This is reasonable as images are usually heavily correlated in spatial dimensions. Moreover, the learned topological indices also give some insight for designing parallel autoregressive models, that channel and spatial context models could be combined more compactly to improve their performance.

## 7 Conclusions and Discussions

This paper presents a novel approach, BaSIC, to addressing computational scalability in Neural Image Compression (NIC) by focusing on BayesNet structure learning. We decompose the challenge into two subproblems: heterogeneous bipartite graph learning for inter-node BayesNet, and multipartite graph learning for intra-node BayesNet, proposing solutions for each. Experimental results, using a joint autoregressive framework as a test case, show that our work offers improved control over computational demands relative to other scalable computation models.

## Acknowledgement

The paper is supported in part by the National Natural Science Foundation of China (No. 62325109, U21B2013).

## References

1. Kodak lossless true color image suite. https://r0k.us/graphics/kodak/ (2023), accessed: 2023-11-12
2. Workshop and challenge on learned image compression. https://clic.compression.cc/ (2023), accessed: 2023-01-09
3. Ballé, J., Laparra, V., Simoncelli, E.P.: Density modeling of images using a generalized normalization transformation. CoRR **abs/1511.06281** (2015), https://api.semanticscholar.org/CorpusID:2684987
4. Ballé, J., Minnen, D.C., Singh, S., Hwang, S.J., Johnston, N.: Variational image compression with a scale hyperprior. ArXiv **abs/1802.01436** (2018)
5. Bégaint, J., Racapé, F., Feltman, S., Pushparaja, A.: Compressai: a pytorch library and evaluation platform for end-to-end compression research. arXiv preprint arXiv:2011.03029 (2020)
6. Cheng, J., Greiner, R., Kelly, J., Bell, D.A., Liu, W.: Learning bayesian networks from data: An information-theory based approach. Artif. Intell. **137**, 43–90 (2002), https://api.semanticscholar.org/CorpusID:7821347
7. Cheng, Z., Sun, H., Takeuchi, M., Katto, J.: Learned image compression with discretized gaussian mixture likelihoods and attention modules. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 7936–7945 (2020)
8. Chickering, D.M.: Optimal structure identification with greedy search. J. Mach. Learn. Res. **3**, 507–554 (2003), https://api.semanticscholar.org/CorpusID:1191614
9. Deleu, T., G'ois, A., Emezue, C.C., Rankawat, M., Lacoste-Julien, S., Bauer, S., Bengio, Y.: Bayesian structure learning with generative flow networks. ArXiv **abs/2202.13903** (2022), https://api.semanticscholar.org/CorpusID:247158659
10. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. 2009 IEEE Conference on Computer Vision and Pattern Recognition pp. 248–255 (2009)
11. He, D., Yang, Z., Peng, W., Ma, R., Qin, H., Wang, Y.: Elic: Efficient learned image compression with unevenly grouped space-channel contextual adaptive coding. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 5708–5717 (2022), https://api.semanticscholar.org/CorpusID:247594672
12. He, D., Zheng, Y., Sun, B., Wang, Y., Qin, H.: Checkerboard context model for efficient learned image compression. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 14766–14775 (2021), https://api.semanticscholar.org/CorpusID:232404386
13. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: Neural Information Processing Systems (2017), https://api.semanticscholar.org/CorpusID:326772

14. Jang, E., Gu, S.S., Poole, B.: Categorical reparameterization with gumbel-softmax. ArXiv **abs/1611.01144** (2016)
15. Johnston, N., Eban, E., Gordon, A., Ball'e, J.: Computationally efficient neural image compression. ArXiv **abs/1912.08771** (2019), https://api.semanticscholar.org/CorpusID:209404937
16. Kang, N., Qiu, S., Zhang, S., Li, Z., Xia, S.: Pilc: Practical image lossless compression with an end-to-end gpu oriented neural framework. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 3729–3738 (2022), https://api.semanticscholar.org/CorpusID:249625569
17. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. CoRR **abs/1312.6114** (2013)
18. Koivisto, M., Sood, K.: Exact bayesian structure discovery in bayesian networks. J. Mach. Learn. Res. **5**, 549–573 (2004), https://api.semanticscholar.org/CorpusID:12412451
19. Lee, J., Cho, S., Beack, S.: Context-adaptive entropy model for end-to-end optimized image compression. In: International Conference on Learning Representations (2018)
20. Li, M., Ma, K., You, J.J., Zhang, D., Zuo, W.: Efficient and effective context-based convolutional entropy modeling for image compression. IEEE Transactions on Image Processing **29**, 5900–5911 (2019), https://api.semanticscholar.org/CorpusID:195345342
21. Lin, F., Sun, H., Liu, J., Katto, J.: Multistage spatial context models for learned image compression. ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) pp. 1–5 (2023), https://api.semanticscholar.org/CorpusID:257038187
22. Liu, H., Simonyan, K., Yang, Y.: Darts: Differentiable architecture search. ArXiv **abs/1806.09055** (2018), https://api.semanticscholar.org/CorpusID:49411844
23. Lorch, L., Rothfuss, J., Scholkopf, B., Krause, A.: Dibs: Differentiable bayesian structure learning. ArXiv **abs/2105.11839** (2021), https://api.semanticscholar.org/CorpusID:235187432
24. Minnen, D.C., Ballé, J., Toderici, G.: Joint autoregressive and hierarchical priors for learned image compression. ArXiv **abs/1809.02736** (2018)
25. Minnen, D.C., Singh, S.: Channel-wise autoregressive entropy models for learned image compression. 2020 IEEE International Conference on Image Processing (ICIP) pp. 3339–3343 (2020), https://api.semanticscholar.org/CorpusID:220633468
26. Mnih, A., Rezende, D.J.: Variational inference for monte carlo objectives. In: International Conference on Machine Learning (2016), https://api.semanticscholar.org/CorpusID:5859948
27. Tao, L., Gao, W., Li, G., Zhang, C.: Adanic: Towards practical neural image compression via dynamic transform routing. 2023 IEEE/CVF International Conference on Computer Vision (ICCV) pp. 16833–16842 (2023), https://api.semanticscholar.org/CorpusID:267026318
28. webmproject: libwebp. https://github.com/webmproject/libwebp (2023), accessed: 2023-01-09
29. Yang, F., Herranz, L., Cheng, Y., Mozerov, M.G.: Slimmable compressive autoencoders for practical neural image compression. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 4996–5005 (2021), https://api.semanticscholar.org/CorpusID:232417114

30. Yin, S., Li, C., Bao, Y., Liang, Y.: Universal efficient variable-rate neural image compression. ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) pp. 2025–2029 (2021), https://api.semanticscholar.org/CorpusID:244478380
31. Yin, S., Meng, F., Tan, W.T., Li, C., Bao, Y., Liang, Y., Liu, W.: Exploring structural sparsity in neural image compression. 2022 IEEE International Conference on Image Processing (ICIP) pp. 471–475 (2022), https://api.semanticscholar.org/CorpusID:246680031
32. Yu, J., Yang, L., Xu, N., Yang, J., Huang, T.S.: Slimmable neural networks. ArXiv **abs/1812.08928** (2018), https://api.semanticscholar.org/CorpusID:56657799
33. Yu, Y., Chen, J., Gao, T., Yu, M.: Dag-gnn: Dag structure learning with graph neural networks. In: International Conference on Machine Learning (2019), https://api.semanticscholar.org/CorpusID:128358697
34. Yuan, C., Malone, B.M.: Learning optimal bayesian networks: A shortest path perspective. J. Artif. Intell. Res. **48**, 23–65 (2013), https://api.semanticscholar.org/CorpusID:5951724
35. Zhang, Y., Lu, G., Feng, D., Zhu, C., Song, L.: Content adaptive checkerboard context model for learned image compression. 2023 IEEE International Symposium on Circuits and Systems (ISCAS) pp. 1–5 (2023), https://api.semanticscholar.org/CorpusID:260002843
36. Zhang, Z., Chen, B., Lin, H., Lin, J., Wang, X., Zhao, T.: Elfic: A learning-based flexible image codec with rate-distortion-complexity optimization. Proceedings of the 31st ACM International Conference on Multimedia (2023), https://api.semanticscholar.org/CorpusID:264492165
37. Zheng, X., Aragam, B., Ravikumar, P., Xing, E.P.: Dags with no tears: Continuous optimization for structure learning. In: Neural Information Processing Systems (2018), https://api.semanticscholar.org/CorpusID:53217974
38. Zheng, Z., Wang, X., Lin, X., Lv, S.: Get the best of the three worlds: Real-time neural image compression in a non-gpu environment. Proceedings of the 29th ACM International Conference on Multimedia (2021), https://api.semanticscholar.org/CorpusID:239011955