LotteryCodec: Searching the Implicit Representation in a Random Network for Low-Complexity Image Compression

Haotian Wu [1] **Gongpu Chen** [1] **Pier Luigi Dragotti** [1] **Deniz Gündüz** [1]

Abstract

We introduce and validate the *lottery codec hypothesis*, which states that untrained subnetworks within randomly initialized networks can serve as synthesis networks for overfitted image compression, achieving rate-distortion (RD) performance comparable to trained networks. This hypothesis leads to a new paradigm for image compression by encoding image statistics into the network substructure. Building on this hypothesis, we propose LotteryCodec, which overfits a binary mask to an individual image, leveraging an over-parameterized and randomly initialized network shared by the encoder and the decoder. To address over-parameterization challenges and streamline subnetwork search, we develop a rewind modulation mechanism that improves the RD performance. LotteryCodec outperforms VTM and sets a new state-of-the-art in single-image compression. LotteryCodec also enables adaptive decoding complexity through adjustable mask ratios, offering flexible compression solutions for diverse device constraints and application requirements. Project page: https://eedavidwu.github.io/LotteryCodec/

1. Introduction

Traditional image/video compression algorithms rely on meticulously designed linear transforms. Recently, conventional methods have been increasingly challenged by emerging learning-based codecs that replace analysis and synthesis transforms in classical codecs with neural networks (Ballé et al., 2018; Cheng et al., 2020a; He et al., 2022). While achieving impressive performance gains, these autoencoder (AE)-based neural codecs often suffer from high decoding complexity and large number of network parameters, which limit their practical deployment on resource-constrained devices (Jiang et al., 2023; Wang et al., 2023). In addition, they require training on extremely large datasets to ensure robust performance across diverse image distributions — resources that are not always available. Addressing these issues to develop low-complexity and robust codecs with competitive rate-distortion (RD) performance remains a critical open challenge (Yang et al., 2023).

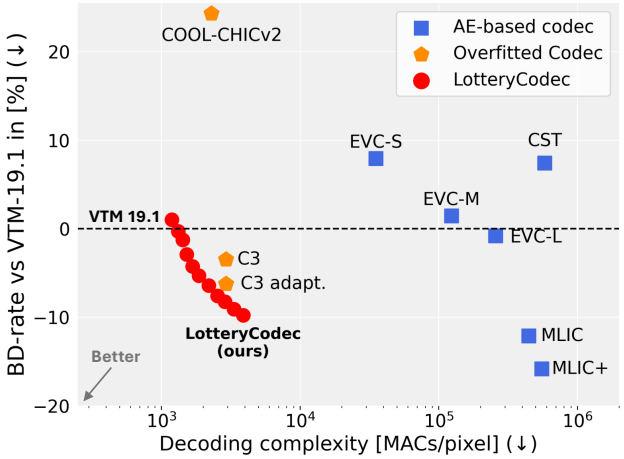


Figure 1. Rate-distortion performance (BD-rate) vs. decoding complexity on the CLIC2020 dataset. LotteryCodec achieves a superior and adaptable RD trade-off than other codecs.

Implicit neural representations (INRs) (Sitzmann et al., 2020) has emerged as a promising signal representation technique that leverages lightweight multi-layer perceptrons (MLPs) to directly parameterize continuous functions. INRs have demonstrated impressive performance in various modalities and tasks, including modality-agnostic representation (Shi et al., 2024), 3D reconstruction (Atzmon & Lipman, 2020), and view synthesis (Mildenhall et al., 2021), and have been extended to multi-instance settings via modulation mechanisms (Mehta et al., 2021). Quantizing the overfitted network yields a signal compressor. For image compression, a lightweight neural network can overfit a single image by mapping pixel coordinates to their

[1]Department of Electrical and Electronic Engineering, Imperial College London, London SW7 2AZ, U.K.. Correspondence to: Gongpu Chen <gongpu.chen@imperial.ac.uk>, Haotian Wu <haotian.wu17@imperial.ac.uk>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

corresponding intensities. Building on this idea, the first INR-based overfitted image codec, COIN, was proposed in (Dupont et al., 2021). Later, the COOL-CHIC series codecs (Ladune et al., 2023; Leguay et al., 2023) and C3 (Kim et al., 2024) further enhanced the RD performance, outperforming widely used codecs such as BPG (Bellard, 2015), HEVC (Sullivan et al., 2012), and VCC (Bross et al., 2021a), while maintaining low decoding complexity.

Despite significant advances, state-of-the-art overfitted image codecs still fall short of the RD performance of competitive classical codecs such as VTM (Bross et al., 2021b). Further improvements are challenging because achieving higher reconstruction fidelity typically necessitates larger networks, which significantly increases the compression rate. In addition, most overfitted codecs rely on fixed network architectures for a range of images, limiting their RD performance and adaptability. We thus anticipate a new paradigm, one that can balance the increasing cost of network complexity with representation capability while enabling an adaptive architecture for individual images.

This work is inspired by two key findings: (1) a randomly initialized neural network can act as a handcrafted prior, encoding significant image statistics and prior information within its network structure (Ulyanov et al., 2018); and (2) over-parameterized neural networks contain high-performing untrained subnetworks (Ramanujan et al., 2020). These insights motivate us to propose the *lottery codec hypothesis*. Specifically, consider a well-trained overfitted image codec $g_{\mathbf{W}}(\mathbf{z})$, which represents image $\mathbf{S}$ as a neural network parameterized by $\mathbf{W}$, with a latent representation $\mathbf{z}$ as its input. After quantization and entropy coding, the decoder reconstructs the image as $\mathbf{S}^* = g_{\hat{\mathbf{W}}}(\hat{\mathbf{z}})$ [1]. Next, consider an over-parameterized, randomly initialized network $g_{\mathbf{W}'}$, along with a learned binary mask $\tau' \in \{0,1\}^{|\mathbf{W}'|}$ and a latent $\mathbf{z}'$, where $|\mathbf{W}'|$ denotes the number of parameters of $g_{\mathbf{W}'}$. The source image is reconstructed via a subnetwork of $g_{\mathbf{W}'}$ as $\mathbf{S}' = g_{\mathbf{W}' \odot \tau'}(\hat{\mathbf{z}}')$, where $\odot$ represents the Hadamard product, identifying the subnetwork. We propose the following hypothesis:

Lottery codec hypothesis. *Let d denote a distortion function and H the entropy function. For any overfitted image codec $g_{\mathbf{W}}(\mathbf{z})$, there exists a pair $(\tau', \mathbf{z}')$ as the 'winning tickets' within a sufficiently over-parameterized and randomly initialized network $g_{\mathbf{W}'}$ satisfying $|\mathbf{W}'| > |\mathbf{W}|$, such that $d(\mathbf{S}, \mathbf{S}') \leq d(\mathbf{S}, \mathbf{S}^*)$ and $H(\hat{\mathbf{z}}') = H(\hat{\mathbf{z}})$.*

Note that the equality $H(\hat{\mathbf{z}}') = H(\hat{\mathbf{z}})$ implies that the number of bits required to represent $\hat{\mathbf{z}}'$ and $\hat{\mathbf{z}}$ after entropy coding is the same. We conjecture this hypothesis to hold for the following reasons: (1) Theoretical justification: prior studies (Pensia et al., 2020; da Cunha et al., 2022) suggest that any target network of width l_w and depth l_d can be approximated by pruning a random network that is a factor $O(\log(l_w l_d))$ wider and twice as deep, suggesting that sufficiently over-parameterized, randomly initialized networks contain some 'winning tickets' even without training. (2) Empirical evidence: extensive experiments conducted in this paper evaluate the hypothesis, and the results consistently support its validity.

[1] In this paper, variables with a hat notation, such as $\hat{\mathbf{z}}$, represent their quantized counterparts.

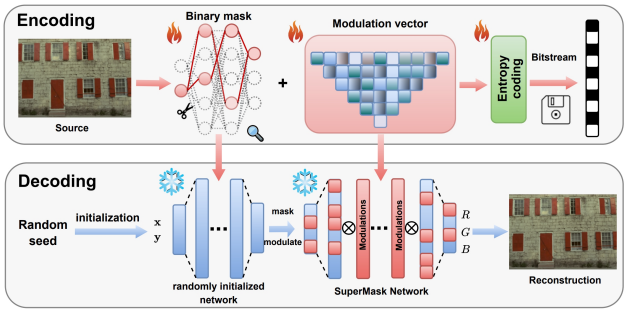


Figure 2. Illustration of LotteryCodec scheme: the source image is encoded into a binary mask and latent modulations. During decoding, the receiver initializes a common random network and uses a modulated subnetwork to reconstruct the source image.

The proposed *lottery codec hypothesis* highlights the potential of searching untrained but well-performing subnetworks as overfitted image codecs. While theoretical works and experiments demonstrate that sufficiently over-parameterized networks can contain untrained subnetworks that match the performance of well-trained networks, precise guidelines on the required level of over-parameterization, such as specific architectural depth or configurations, remain unclear for reliably obtaining a 'winning ticket'.

Based on the above hypothesis, we propose a novel overfitted image compression scheme, called LotteryCodec (see Fig. 2). This method utilizes a randomly initialized network as the synthesis network and learns a binary mask with latent modulations as the code. With a predefined random seed, network parameters are eliminated from transmissions; instead, it is sufficient to transmit only the binary mask and latent variables for the decoder to reconstruct the image.

To address the challenges of over-parameterization and subnetwork search, we propose a rewind modulation mechanism that introduces image information during the subnetwork search, simplifying the process and enhancing the RD performance. *To the best of our knowledge, the proposed LotteryCodec is the first overfitted image codec to surpass the RD performance of VTM while maintaining a low decoding complexity.* Furthermore, LotteryCodec sets a new state-of-the-art for overfitted neural codecs obtained from a single image. Its adaptive masking strategy enables flexible

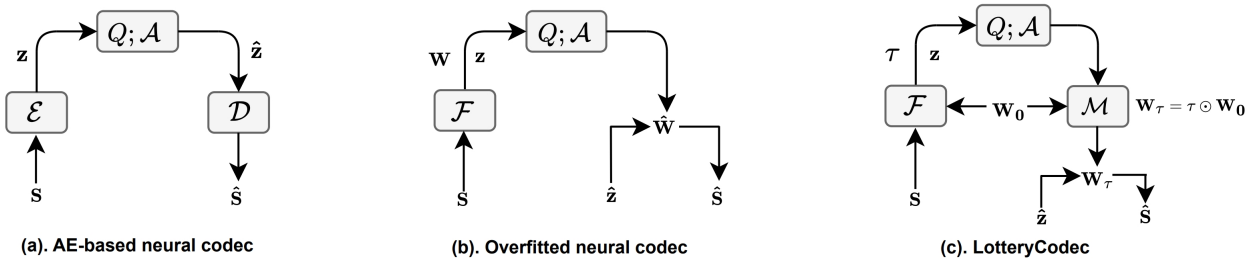


Figure 3. Operational structure of different compression schemes. (a) AE-based neural codecs: source image $\mathbf{S}$ is processed through a pair of encoder and decoder. (b) Overfitted neural codecs: $\mathbf{S}$ is fitted by parameters $\{\mathbf{W}, \mathbf{z}\}$ via a fitting operation $\mathcal{F}$. (c) LotteryCodec: $\mathbf{S}$ is fitted by parameters $\{\tau, \mathbf{z}\}$, identifying a subnetwork in a randomly initialized network, with masking operations $\mathcal{M}$.

model complexity adjustment based on varying mask ratios, balancing computational cost and performance, as shown in Fig. 1.

The contributions of this work are summarized as follows:

- We propose and experimentally verify the *lottery codec hypothesis*, which suggests that a subnetwork within a randomly initialized neural network can directly serve as a well-performing synthesis network for overfitted image compression. This hypothesis introduces a new paradigm for INR-based image compression, emphasizing the potential of encoding image statistics into the structure of a randomly initialized network.
- We propose a novel LotteryCodec scheme, which overfits a randomly initialized neural network to the source image by learning a binary mask and modulation vectors. To alleviate over-parameterization and simplify subnetwork search, a rewind modulation mechanism is introduced, significantly improving the RD performance.
- We show by extensive experiments that LotteryCodec achieves state-of-the-art performance among overfitted image codecs designed for single-image compression at a reduced computational cost. Additionally, LotteryCodec can adjust its decoding complexity by varying the mask ratio, thus providing flexible solutions for diverse computational and performance needs.

2. Related work

2.1. Neural data compression

Currently, there are two main paradigms for neural data compression (see Fig. 3): AE-based and overfitted neural codecs, both designed to balance the trade-off between the distortion and the rate (Cover, 1999).

AE-based neural codecs. As illustrated in Fig. 3a, an AE-based neural codec, e.g., (Ballé et al., 2018; Cheng et al., 2020a; He et al., 2022; Jiang et al., 2023; Wang et al., 2023), comprises a pair of encoding network $\mathcal{E}$ and decoding network $\mathcal{D}$, jointly optimized over a large dataset. A source image $\mathbf{S}$ is encoded by $\mathcal{E}$ into a latent representation $\mathbf{z}$, which is then compressed into R bits by quantization and entropy coding. At the decoder side, the quantized latent representation $\hat{\mathbf{z}}$ is first recovered and then used to reconstruct the source image. Resulting average distortion is $D = \mathbb{E}_{\mathbf{S}\sim p_s}\left[d(\mathbf{S}, \mathcal{D}(\hat{\mathbf{z}}))\right]$, where d denotes the distortion metric, and the compression rate R is given by:

$$R = \mathbb{E}_{\mathbf{S}\sim p_s}\left[-\log_2 p_{Q(\mathcal{E}(\mathbf{S}))}\left(\hat{\mathbf{z}}\right)\right], \tag{1}$$

where Q represents quantization operations. Although AE-based approaches achieve competitive performance, their decoding complexity is typically high due to reliance on large and complex architectures for robust latent representations (see Fig. 1).

Overfitted neural codecs. Overfitting a parametric function to each single image offers an alternative, where quantized function parameters serve as the compressed representation. A low-complexity network is often sufficient since generalization is not required. COIN (Dupont et al., 2021) pioneered this by training a lightweight MLP to map pixel coordinates to RGB values. Quantized parameters are entropy-coded into a bitstream for decoding. COIN++ (Dupont et al., 2022a) improved generalization with meta-learning. However, COIN's RD performance remains limited despite low complexity and remarkable generalization. Subsequently, the COMBINER series (Guo et al., 2023; He et al., 2024) introduces variational INRs and leverages relative entropy coding for RD-optimized compression.

A significant advancement came with COOL-CHIC (Ladune et al., 2023), which improves RD performance by integrating a latent representation and an entropy model while maintaining a low decoding complexity. As illustrated in Fig. 3b, COOL-CHIC jointly trains a parametric function $g_{\mathbf{W}}$ with a latent vector $\mathbf{z}$ as the input to overfit $\mathbf{S}$. Similarly to COIN, both $\mathbf{W}$ and $\mathbf{z}$ are compressed into R bits through quantization and entropy coding. During decoding, the quantized network and latent vector are used to recon-

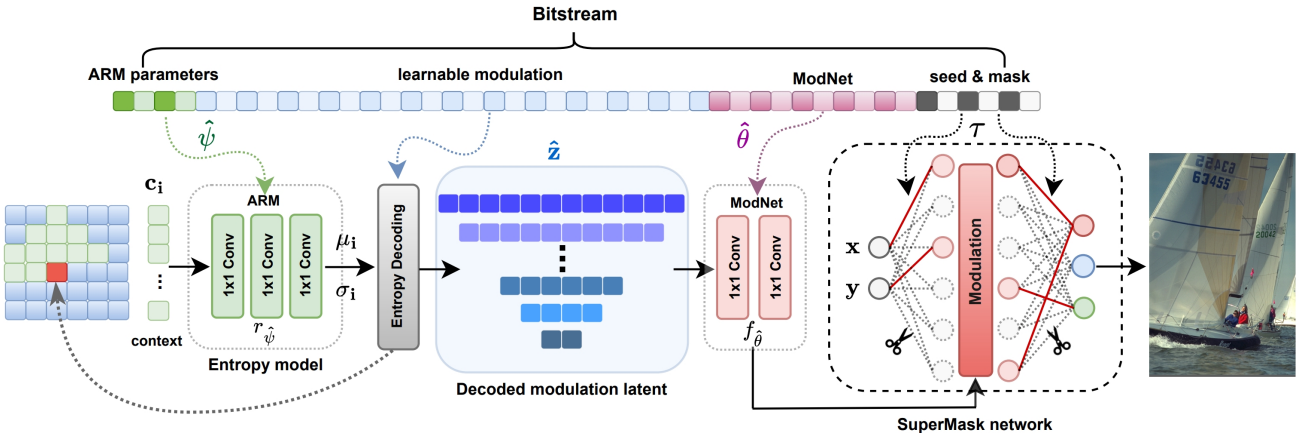


Figure 4. Illustration of the image decoding process in LotteryCodec. The ARM parameters $\hat{\psi}$ are first retrieved to regress the latent modulations $\hat{\mathbf{z}}$. Subsequently, the binary mask τ and initialization seed configure the synthesis network, while the modulation model parameters $\hat{\theta}$ are decoded to generate the modulations from $\hat{\mathbf{z}}$ to guide the synthesis network to reconstruct the image.

struct the image via the mapping $\hat{\mathbf{S}} = g_{\hat{\mathbf{W}}}(\hat{\mathbf{z}})$. This results in average distortion $D = \mathbb{E}_{\mathbf{S}\sim p_s}\left[d(\mathbf{S}, g_{\hat{\mathbf{W}}}(\hat{\mathbf{z}}))\right]$ at a rate

$$R = \mathbb{E}_{\mathbf{S}\sim p_s}\Big[-\log_2 p_{\hat{\psi}}(\hat{\mathbf{z}}) - \log_2 p(\hat{\mathbf{W}}) + R_{\hat{\psi}}\Big], \quad (2)$$

where $\mathcal{F}$ denotes the operations that overfit $(\mathbf{W}, \mathbf{z})$ parameters to the specific input $\mathbf{S}$, $p_{\hat{\psi}}(\cdot)$ is the estimated distribution with estimation model $\hat{\psi}$. In practice, entropy coding of $\hat{\mathbf{z}}$ relies on a lightweight auto-regressive entropy model (ARM) r_ψ for distribution estimation. In (2), $R_{\hat{\psi}}$ represents the extra bit overhead due to the transmission of this model parameters.

Extensions like COOL-CHICv2 (Leguay et al., 2023), C3 (Kim et al., 2024; Ballé et al., 2024), and COOL-CHICv3 (Blard et al., 2024) further enhance the RD performance with advanced architectures and techniques, including soft-rounding, Kumaraswamy noise, and conditional entropy models. These advancements allow COOL-CHIC to outperform widely used codecs such as BPG and HEVC.

2.2. Lottery ticket hypothesis

Frankle & Carbin (2019) introduced the *lottery ticket hypothesis* (LTH), stating that a randomly initialized, dense neural network contains a subnetwork that can match the test accuracy of the full network with equivalent training. Zhou et al. (2019) found that 'winning tickets' (i.e., masked subnetworks) can outperform random initialization without training, while Ramanujan et al. (2020) extended the idea by showing that even untrained subnetworks can achieve near state-of-the-art performance, a phenomenon referred to as the *strong lottery ticket hypothesis* (SLTH). Building on these ideas, Choi et al. (2023) applied LTH to video representation with multiple supermask overlays and unpruned biases. This LTH-based video representation method enhances expressiveness but increases complexity.

Recently, Malach et al. (2020) theoretically proved the SLTH for fully connected networks with ReLU activations, showing that any target network can be approximated by pruning a sufficiently over-parameterized random network of polynomial size relative to the target network. Orseau et al. (2020) and Pensia et al. (2020) relaxed the assumptions and improved these bounds to logarithmic order, which is later extended into convolutional neural networks (CNNs) in (da Cunha et al., 2022).

3. LotteryCodec

Inspired by the *lottery codec hypothesis*, we propose LotteryCodec, a novel image compression paradigm that encodes images by identifying structured subnetworks within a randomly initialized network. The detailed pseudocode for the method is provided in Appendix E.

3.1. Workflow

Encoder. As depicted in Fig 3(c), given an over-parameterized network $g_{\mathbf{W}_0}$ with randomly initialized parameters $\mathbf{W}_0$, each image $\mathbf{S}$ is overfitted using a subnetwork of $g_{\mathbf{W}_0}$ by learning a binary mask τ and a latent representation $\mathbf{z}$, such that the subnetwork $g_{\tau\odot\mathbf{W}_0}$ can well reconstruct $\mathbf{S}$ using $\hat{\mathbf{z}}$. The quantized latent $\hat{\mathbf{z}}$ and binary mask τ are then compressed via entropy coding, producing a bitstream representation of $\mathbf{S}$. Additional mechanisms, such as ARM r_ψ and the modulation model ModNet f_θ, are incorporated to further enhance the RD performance, as will be elaborated on later.

Decoder. As shown in Fig. 4, the decoder begins by ini-

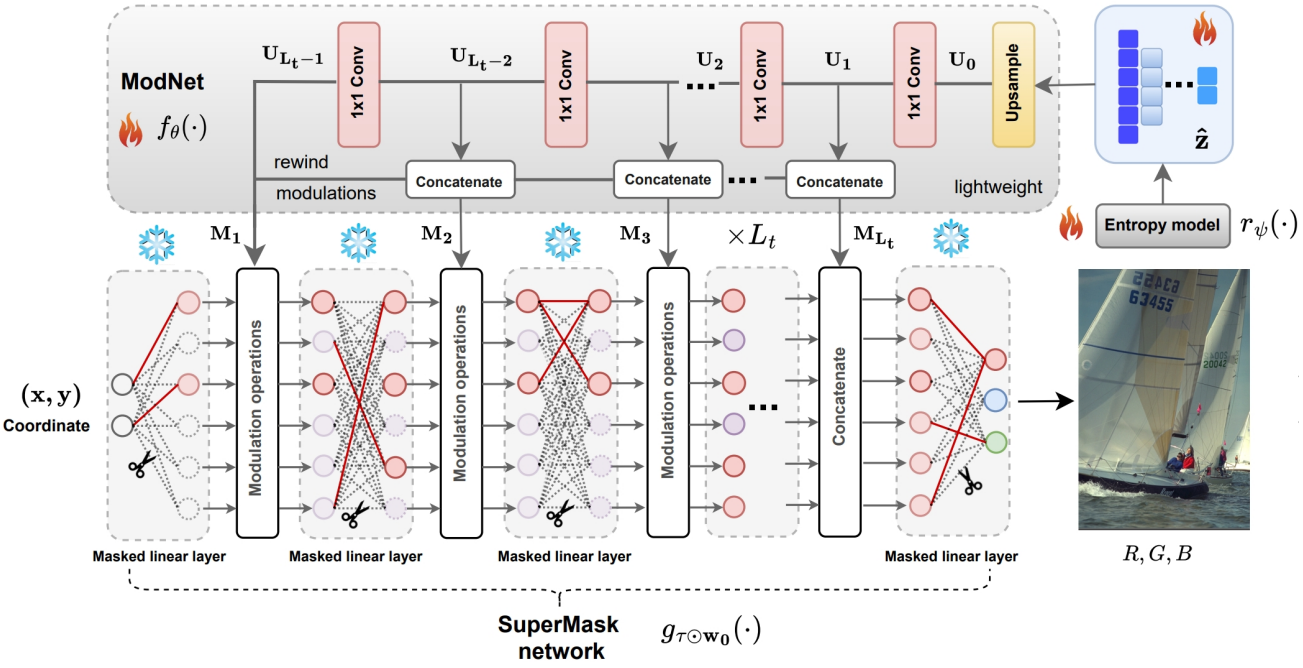


Figure 5. Illustration of the ModNet and SuperMask networks: SuperMask network maps pixel coordinates to RGB values by identifying subnetworks within a randomly initialized network, guided by modulations generated by the ModNet using input $\hat{\mathbf{z}}$. Solid red lines indicate active weights, while dashed lines represent masked weights. The weights of the SuperMask network remain frozen during training; only the binary mask τ and ModNet parameters are learned.

tializing the neural network $g_{\mathbf{W}0}$ using the same random seed as the encoder. Next, the binary mask τ is extracted to identify a subnetwork $g_{\tau\odot\mathbf{w_0}}$, and the ARM parameters $\hat{\psi}$ are retrieved to decode the latent modulation vector $\hat{\mathbf{z}}$. Finally, a modulation model $f_{\hat{\theta}}$ is retrieved and applied to modulate the image generation process based on $\hat{\mathbf{z}}$ as:

$$\hat{\mathbf{S}}(\mathbf{x}) = g_{\tau\odot\mathbf{w_0}}(f_{\hat{\theta}}(\hat{\mathbf{z}}), \mathbf{x}), \tag{3}$$

where $\mathbf{x}$ denotes the pixel coordinate vector of the image. The distortion of LotteryCodec is then measured by $D = \mathbb{E}_{\mathbf{S}\sim p_s}\left[d(\mathbf{S}, \hat{\mathbf{S}})\right]$. The decoding complexity remains low because $g_{\tau\odot\mathbf{w_0}}$ is a lightweight network. Moreover, LotteryCodec offers flexible decoding complexity by allowing adjustable mask ratios, which dynamically control the active size of the synthesis network.

RD cost optimization. To balance the rate-distortion trade-off, LotteryCodec is trained to fit a parameter set, denoted by $\mathbf{\Omega} \triangleq \{\mathbf{z}, \psi, \theta, \tau\}$, with the goal of minimizing the following loss function (i.e., the RD cost):

$$\mathcal{L}(\mathbf{\Omega}) = d(\mathbf{S}, g_{\tau\odot\mathbf{w_0}}(f_{\hat{\theta}}(\hat{\mathbf{z}}), \mathbf{x})) + \lambda R(\hat{\mathbf{z}}), \tag{4}$$

where $R(\hat{\mathbf{z}})$ represents the compression rate contributed by $\hat{\mathbf{z}}$, and λ is a hyperparameter that controls the trade-off between distortion and rate. During training, $\mathbf{z}$ undergoes soft-rounding with varying Kumaraswamy noise, whereas hard-rounding is applied during inference. Notably, the loss function (4) excludes the rate terms associated with ψ, θ, and τ, as their contribution to the overall bit rate is minimal due to their lightweight architectures. In practice, these parameters are quantized and entropy-coded using a non-learned distribution after the optimization process is completed, contributing to the final bitstream.

Overall, the LotteryCodec bitstream consists of four parts: the ARM parameters $\hat{\psi}$, learnable latent modulations $\hat{\mathbf{z}}$, modulation model parameters $\hat{\theta}$, and the binary mask τ. Hence, the total compression rate is given by

$$R = \mathbb{E}_{\mathbf{S}\sim p_s}\Big[-\log_2 p_{\hat{\psi}}(\hat{\mathbf{z}}) - \log_2 p(\tau) + R_{\hat{\theta}} + R_{\hat{\psi}}\Big], \tag{5}$$

where $R_{\hat{\theta}}$ and $R_{\hat{\psi}}$ denote the rate contributed by the transmission of quantized ModNet and ARM. As shown in Eqs. (2) and (5), the rate of overfitted codecs depends on $\{\hat{\mathbf{z}}, \hat{\psi}, \hat{\mathbf{W}}\}$, while the rate of our method is determined by $\{\hat{\mathbf{z}}, \hat{\psi}, \tau, \hat{\theta}\}$. According to the Lottery Codec Hypothesis, our bit cost for $\hat{\mathbf{z}}$ and $\hat{\psi}$ matches that of standard overfitted codecs. While each quantized parameter in $\hat{W}$ typically requires more than 13 bits, our binary mask τ uses up to 1 bits per entry. Despite its higher dimensionality, τ contributes significantly less to the total rate. Moreover, since $\hat{\theta}$ is lightweight, the combined rate of τ and $\hat{\theta}$ remains lower

than that of $\hat{\mathbf{W}}$, resulting in an improved compression efficiency. Note also that τ is transmitted in a lossless fashion as it does not need to be quantized. In practice, both τ and $\hat{\theta}$ are entropy-coded using offline-trained models or a static distribution.

3.2. Winning a lottery codec

The design of LotteryCodec incorporates four key architectural components: SuperMask network ($g_{\mathbf{W_0}\odot\tau}$), ModNet ($f_\theta$), latent modulation ($\mathbf{z}$), and ARM ($r_\psi$). As shown in Fig. 5, the SuperMask network $g_{\tau\odot\mathbf{W_0}}$ is a high-performing subnetwork that maps pixel coordinates to RGB values, obtained by applying the learned mask τ to the over-parameterized network $g_{\mathbf{W_0}}$. Specifically, $g_{\tau\odot\mathbf{W_0}}$ comprises L_t masked linear layers, with each masked linear layer followed by modulation operations. The ModNet f_θ includes an upsampling operation and (L_t-1) convolutional layers with 1×1 kernel to generate hierarchical modulation vectors for each layer of $g_{\tau\odot\mathbf{W_0}}$ from the retrieved $\hat{\mathbf{z}}$. This design significantly simplifies the subnetwork search while improving the RD performance. The latent modulation $\mathbf{z}$ is a set of learnable vectors that serve as a primary compression component, while the ARM (r_ψ) is an auto-regressive entropy model commonly used to compress latent representation $\mathbf{z}$ in overfitted codecs (Ladune et al., 2023), with architecture detailed in Appendix A.

Fourier initialization. Initialization plays a critical role in the SLTH problem to ensure the efficient identification of "winning tickets" (Ramanujan et al., 2020). In our LotteryCodec, parameters $\mathbf{W_0}$ are initialized using a Fourier initialization approach. This design is based on two main considerations: (1) From an INR perspective, the Fourier reparametrization method can address the low-frequency bias of MLPs and enhance INR with richer textures (Shi et al., 2024); (2) From an SLTH point of view, this initialization ensures the network retains rich sign information (Zhou et al., 2019; Oh et al., 2025) while maintaining constant variance between inputs and outputs (Glorot & Bengio, 2010; He et al., 2016).

Specifically, the weight matrix of the i-th MLP layer, denoted by $\mathbf{W^{(i)}} \in \mathbb{R}^{d_i\times d_{i-1}}$, is re-parameterized as a weighted combination of fixed Fourier bases:

$$\mathbf{W^{(i)}} = \mathbf{\Lambda^{(i)}}\mathbf{B^{(i)}}, \tag{6}$$

where $\mathbf{\Lambda^{(i)}} \in \mathbb{R}^{d_i\times M}$ is the coefficient matrix, and $\mathbf{B^{(i)}} \in \mathbb{R}^{M\times d_{i-1}}$ represents a set of M Fourier bases. Each element $b^{(i)}_{m,n}$ of $\mathbf{B^{(i)}}$ (the m-th row and n-column) is defined using a distinct frequency and phase as $b^{(i)}_{m,n} = \cos(w_m a^{(i)}_n + \varphi_m)$, where $\mathbf{a^{(i)}} = a^{(i)}_1, \ldots, a^{(i)}_{d_{i-1}}$ is a positional sequence uniformly sampled from $[-\pi, \pi]$, and $\mathbf{w}$ and φ are the frequency and phase vectors. We adopt P different phases and $2F$ different frequencies, hence $M = 2FP$. The phase vector is defined as $\varphi \triangleq \{0, 2\pi/P, \ldots, 2\pi(P-1)/P\}$. For each phase, the frequency vector is defined as $\mathbf{w} \triangleq \{\mathbf{w_{low}}, \mathbf{w_{high}}\}$, where $\mathbf{w_{low}} = \{1/F, 2/F, \ldots, 1\}$ and $\mathbf{w_{high}} = \{1, 2, \ldots, F, \}$ denote low-frequency and high-frequency bases, respectively. Each element $\lambda^{(i)}_{m,n}$ of $\mathbf{\Lambda^{(i)}}$ is sampled from:

$$\lambda^{(i)}_{m,n} \sim U\left(-\sqrt{\frac{6}{M\sum_{t=1}^{d_{i-1}}(b^{(i)}_{m,t})^2}}, \sqrt{\frac{6}{M\sum_{t=1}^{d_{i-1}}(b^{(i)}_{m,t})^2}}\right).$$

SuperMask network. We introduce a learnable matrix $\mathbf{P}$ to identify the subnetwork $g_{\tau\odot\mathbf{W}_0}$, defined by mask τ, that minimizes the loss function. Matrix $\mathbf{P}$ shares the same dimensions as $\mathbf{W}_0$, with each element representing a score of the corresponding weight in $\mathbf{W_0}$. During training, the randomly initialized $\mathbf{W_0}$ remains frozen while only $\mathbf{P}$ is updated. The top $r_a\%$ of weights with the highest scores across all layers are activated, while the remaining weights are set to zero.

More formally, let $\mathcal{V}^{(\mathbf{i})} \triangleq \{v^{(i)}_1, \ldots, v^{(i)}_{d_i}\}$ denote the values of nodes for the i-th masked linear layer with d_i nodes. The output of the k-th neuron in the i-th layer with the modulation function $m(\cdot)$ can then be written as:

$$v^{(i)}_k = \sigma\left(\sum_{j=1}^{d_{i-1}} \tau^{(i-1)}_{k,j} w^{(i-1)}_{kj} m(v^{(i-1)}_j)\right), \tag{7}$$

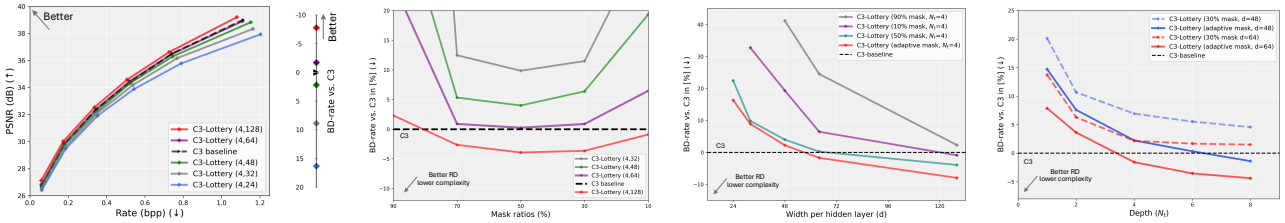
where σ is the activation function and $w^{(i-1)}_{kj}$ is the weight connecting the k-th neuron in i-th layer to the j-th neuron of the $(i-1)$-th layer. The mask function returning the binary mask $\tau^{(i-1)}_{k,j} = h(\mathbf{P})$ is defined as: $\tau^{(i-1)}_{k,j} = 1$ if the score $p^{(i-1)}_{kj}$ for the weight is among the top $r_a\%$ highest values, and $\tau^{(i-1)}_{k,j} = 0$ otherwise. Using the chain rule of gradient, the gradient of the loss $\mathcal{L}$ with respect to $p^{(i-1)}_{kj}$ can be estimated by a straight-through gradient estimator:

$$\frac{\partial\mathcal{L}}{\partial p^{(i-1)}_{kj}} \approx \frac{\partial\mathcal{L}}{\partial v^{(i)}_k}\frac{\partial v^{(i)}_k}{\partial\sigma} w^{(i-1)}_{kj} m(v^{(i-1)}_j) \triangleq \varsigma^{(i-1)}_{kj}. \tag{8}$$

The above formula holds because $\tau^{(i-1)}_{k,j}$ is a non-decreasing function of $p^{(i-1)}_{kj}$. The activation probability $p^{(i-1)}_{kj}$ can then be updated as:

$$p^{(i-1)}_{kj} \leftarrow p^{(i-1)}_{kj} - \alpha\varsigma^{(i-1)}_{kj}, \tag{9}$$

where α is the learning rate. When activation or deactivation swaps occur, the loss decreases for the mini-batch, according to (Ramanujan et al., 2020; Wortsman et al., 2019).



(a) (b) (c) (d)

Figure 6. Experimental verification of the *lottery codec hypothesis*, where C3-lottery (N_t, d) refers to the scheme using an over-parameterized network with N_t hidden layers and d dimensions per layer. (a) RD curve and BD rate for different over-parameterization configurations. (b) BD-rate versus different mask ratios. (c)-(d) BD-rate over varying width and depth configurations.

Rewind modulation mechanism. In practice, directly searching for a well-performing subnetwork from $g_{\mathbf{W_0}}$ requires a deep, wide random network, leading to high compression and computational costs. To address this, we introduce ModNet f_θ to generate modulations to simplify subnetwork search and improve the RD performance. Specifically, f_θ takes the quantized latent modulation $\hat{\mathbf{z}}$ as input, where $\mathbf{z} \triangleq \{\mathbf{z_1}, \mathbf{z_2}, \dots, \mathbf{z_L}\}$ comprises L learnable multi-resolution vectors, with each $\mathbf{z_i} \in \mathbb{Z}^{\frac{HW}{4^{i-1}}}$ being a learnable vector. In f_θ, $\hat{\mathbf{z}}$ is first upsampled using transpose convolutional operations as in (Blard et al., 2024), producing $\mathbf{U_0} = [\mathbf{u_1}; \mathbf{u_2}; \dots; \mathbf{u_L}] \in \mathbb{Z}^{L \times HW}$, where each $\mathbf{u_i} \triangleq \mathit{Upsample}(\hat{\mathbf{z}}_\mathbf{i}) \in \mathbb{Z}^{HW}$. The resultant $\mathbf{U_0}$ is then processed through convolutional layers as: $\mathbf{U_i} = f_\theta^{(i)}(\mathbf{U_{i-1}})$, $i = 1, \dots, L_t - 1$. Here, $f_\theta^{(i)}$ and $\mathbf{U_i}$ denote the operation and output of the i-th layer of ModNet, respectively.

Inspired by (Mehta et al., 2021; Perez et al., 2018; Zhou et al., 2019; Oh et al., 2025) and to leverage the network structure to encode the source image, we propose a rewind strategy to modulate the synthesis process by concatenating compensational structural information in a rewind fashion. The modulation vector for the i-th layer of the synthesis network $g_{\tau \odot \mathbf{W_0}}$ is defined as:

$$\mathbf{M_i} = \mathit{Concatenate}(\mathbf{U_{L_t-1}}, \mathbf{U_{L_t-2}}, \dots, \mathbf{U_{L_t-i}}), \quad (10)$$

where vectors $\mathbf{U_i}$'s from ModNet are concatenated in reverse order to facilitate the search for a high-performing subnetwork.

The modulation operation concatenates $\mathbf{M_i}$ with the intermediate layer output of $g_{\tau \odot \mathbf{W_0}}$. Consequently, the output of each masked linear layer after modulations is given by:

$$\mathbf{G_i} = m(\mathbf{F_i}) = \mathit{Concatenate}(\mathbf{M_i}, \mathbf{F_i}), \quad (11)$$

where $\mathbf{F_i}$ represents the output of the i-th layer of $g_{\tau \odot \mathbf{W_0}}$, and $\mathbf{G_i}$ is the output after corresponding modulation operations. Intuitively, this concatenation enriches the structure of the SuperMask network with both sign and magnitude information, allowing for the reactivation of features in deeper layers while preserving high-level features. Note that the proposed LotteryCodec scheme with its rewind modulation mechanism serves as a general framework, allowing for alternative modulation operations, such as FiLM (Perez et al., 2018), which are discussed in Appendix D.4.

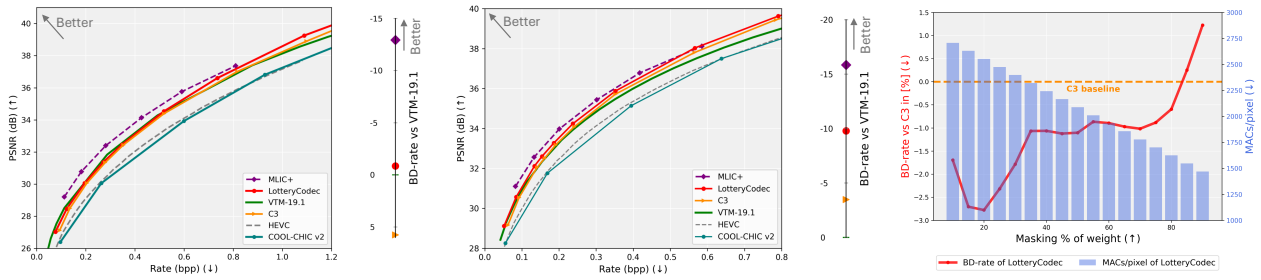
4. Experimental results

We evaluate our model on the Kodak (24 images) (Kodak, 1991) and CLIC2020 (41 images) (Toderici et al., 2020) datasets. The mask ratio is selected from $[0.1, 0.9]$. LotteryCodec is compared with classical codecs (VTM-19.1 (Bross et al., 2021a), HEVC (Sullivan et al., 2012)), AE-based neural codecs (CST (Cheng et al., 2020b), EVC (Wang et al., 2023), MLIC+ (Jiang et al., 2023)), and other overfitted image codecs (including C3 (Kim et al., 2024) and COOL-CHIC (Leguay et al., 2023)). The peak signal-to-noise ratio (PSNR) on RGB channels and BD-rate (Gisle, 2001) are used to evaluate the RD performance. The evaluation details and additional experiments, including ablation study and visualizations, are presented in the Appendix F.

4.1. Verification of the *lottery codec hypothesis*

We first validate the *lottery codec hypothesis* through a series of experiments on the first 10 images of the Kodak dataset. We implement the C3 scheme with a 16-dimensional ARM as the baseline. For comparison, we search for a subnetwork within a randomly initialized network and replace the well-trained synthesis network in C3 with this subnetwork, keeping all other components unchanged. This method is referred to as the C3-Lottery scheme. See Appendix B.4 for more details.

Fig. 6 presents the RD trade-off across varying network depths and widths. The results demonstrate that when the randomly initialized network is sufficiently over-parameterized, the C3-Lottery scheme successfully identifies a subnetwork and an associated latent representation that achieves the same level of distortion as C3, but at a lower bitrate. In particular, Fig. 6a presents PSNR versus the rate



(a) (b) (c)

Figure 7. Performance of LotteryCodec and other schemes. (a) RD curve and BD rate on Kodak dataset. (b) RD curve and BD rate on CLIC2020 dataset. (c) BD-rate and decoding complexity across different mask ratios on Kodak dataset.

contributed by $\hat{\mathbf{z}}$, demonstrating that RD performance (in terms of the PSNR-rate curve and BD-rate) improves as the network width increases. In particular, when depth $N_t = 4$, the C3-Lottery scheme matches or even surpasses the performance of well-trained C3 baselines for width $d \geq 64$. Additional results for different depth settings can be found in Appendix B.4. Fig. 6b illustrates the impact of the mask ratio on performance, revealing that the optimal LotteryCodec is achieved at a mask ratio of approximately 50%. Intuitively, this occurs because a 50% mask ratio maximizes the entropy of the network structure, allowing more information to be encoded into the structure. Figs. 6c and 6d indicate that increasing both the depth and the width of the random network can significantly enhance RD performance. Furthermore, employing adaptive mask ratios for different images could further boost performance.

4.2. RD performance of the LotteryCodec

We then compare the RD performance of LotteryCodec with the existing methods. As shown in Fig. 7a, LotteryCodec outperforms VTM and significantly surpasses other overfitted codecs on the Kodak dataset, achieving BD-rate reductions of -6.4% over C3 and -3.73% over its adaptive variant, C3-adapt (relative to VTM-19.1). One of the substantial performance gains occurs in the low-bpp regime, where network parameters dominate the compression rate in an overfitted codec. By contrast, LotteryCodec achieves a lower rate by compressing a binary mask instead of real-valued parameters, resulting in a superior RD performance (more details are provided in Appendix B.5). Additional MS-SSIM evaluations are provided in the appendix (see Fig. 13), where LotteryCodec closely approaches ELIC and has a $+10.72\%$ BD-rate gap (versus VTM-19.1) compared to MLIC+.

Fig. 7b shows the results on the CLIC2020 dataset, where LotteryCodec demonstrates an even greater advantage. In particular, the performance gain over VTM-19.1 reaches up to -9.79% (see Appendix B.5 for details). Consistent improvements are also observed from Fig. 7b over C3 (-5.69% BD-rate), C3-adpt (-2.90% BD-rate) and COOL-CHIC v2 (-25.57% BD-rate), respectively. From experiments on both the Kodak and CLIC2020 datasets, the LotteryCodec scheme demonstrates state-of-the-art RD performance among current overfitted codecs. *To the best of our knowledge, LotteryCodec is the first neural codec to surpass VTM in RD performance while maintaining low decoding complexity, establishing it as the state-of-the-art for single-image compression.* Like other overfitted codecs, LotteryCodec still falls short of state-of-the-art AE-based neural codecs such as MLIC+ in RD performance. But it achieves a BD-rate over VTM that is comparable to MLIC, with only a $+2.33\%$ gap. As shown in Fig. 1, LotteryCodec demonstrates a significant advantage over MLIC+ in terms of decoding complexity, requiring two orders of magnitude fewer decoding operations. Further discussion on complexity is provided later.

4.3. The mask ratio and decoding complexity

The impact of the mask ratio is examined in Fig. 7c, which illustrates the performance of LotteryCodec under different mask ratios. As expected, the decoding complexity decreases linearly as the mask ratio increases, since masking more weights reduces the overall computation required by the network. However, RD performance does not follow a monotonic trend with respect to the mask ratio. Interestingly, with the introduction of the rewind modulation, the optimal mask ratio for LotteryCodec to achieve the best BD-rate is around 20%, different from 50% observed in Fig. 6. Intuitively, decreasing the mask ratio from 50% to 20% significantly reduces the number of possible subnetworks, demonstrating that rewind modulation effectively simplifies the subnetwork search process while maintaining strong compression performance.

Additionally, Fig. 7c demonstrates that LotteryCodec can flexibly balance RD performance and decoding complexity, allowing for adaptable trade-offs based on specific require-

ments. Notably, even with a high mask ratio of 80%, which results in low decoding complexity, LotteryCodec still outperforms the C3 baseline scheme. More detailed ablation studies and additional experiments on varying mask ratios are presented in Appendix B.5

A more comprehensive comparison of decoding complexity between LotteryCodec and other neural codecs is presented in Fig. 1 and Fig. 10a. In particular, compared to most AE-based schemes, LotteryCodec achieves similar or superior RD performance with at least an order of magnitude fewer MACs. For more advanced codecs like ELIC and MLIC+, while LotteryCodec does not exceed their RD performance, it reduces MACs by over two orders of magnitude while maintaining acceptable quality. Compared to other overfitted image codecs, such as the COOL-CHIC/C3 family, LotteryCodec achieves better RD performance with lower decoding complexity.

Notably, the current figure shows the theoretical minimum decoding complexity, excluding masked operations. This lower bound can be approached with sparsity-aware libraries (e.g., TVM (Chen et al., 2018), cuSparse (Naumov et al., 2010), DeepSparse (Kurtz et al., 2020)) on compatible hardware. For a comprehensive analysis, we also report real coding times using structured pruning (see Tables 7 and 8 in the appendix). Additionally, we provide both theoretical upper and lower bounds on decoding complexity, corresponding to unpruned and active operations, respectively, with practical complexity lying in between. As shown in Fig. 11, even without pruning, LotteryCodec outperforms C3 in BD-rate with similar complexity.

5. Conclusion and future work

This paper introduces and validates the *lottery codec hypothesis*, which proposes a new paradigm for image compression: encoding images into structures of randomly initialized networks. Building on this hypothesis, we propose LotteryCodec, a novel overfitted codec that compresses an image into modulation vectors and a binary mask for an over-parameterized, randomly initialized network. The proposed LotteryCodec achieves state-of-the-art RD performance among existing overfitted codecs while maintaining adaptive and low decoding complexity. Our work advances the field of overfitted image compression by addressing the critical challenge of achieving high compression efficiency with minimal computational cost.

Limitations. LotteryCodec's low and flexible decoding cost is particularly beneficial in multi-user streaming scenarios, where encoding can be done once and offline, to support many users in decoding the same content. While high encoding complexity remains a key bottleneck for all overfitted codecs, including ours, several potential acceleration strategies exist, such as meta-learning, mixed-precision training, and neural architecture search. In particular, LotteryCodec also opens the door to parallel encoding of overfitted codecs by reparameterizing distinct network learning processes into a batch of mask learning processes.

Future work. As a novel paradigm in image compression, LotteryCodec opens several avenues for future research. First, its performance and efficiency could be further optimized by incorporating advanced strategies, such as those proposed in (Kim et al., 2024; Ladune et al., 2024). Second, LotteryCodec can be extended as a flexible alternative for video coding. By sharing modulation across adjacent groups of frames (GoF) and applying distinct masks, it can also encode temporal information into the network structure. Moreover, video coding enables adaptive mask ratio selection across GoF, offering greater flexibility in both computational complexity and rate control. Its low decoding complexity positions LotteryCodec as a promising approach for real-world neural compression applications, enabling efficient deployment in resource-constrained environments.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

Acknowledgments

This work received funding from the UKRI for the projects INFORMED-AI (EP/Y028732/1) and AI-R (ERC Consolidator Grant, EP/X030806/1).

References

Atzmon, M. and Lipman, Y. Sal: Sign agnostic learning of shapes from raw data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2565–2574, 2020.

Ballé, J., Minnen, D., Singh, S., Hwang, S. J., and Johnston, N. Variational image compression with a scale hyperprior. *ICLR 2018-6th International Conference on Learning Representations*, 2018.

Ballé, J., Versari, L., Dupont, E., Kim, H., and Bauer, M. Good, cheap, and fast: Overfitted image compression with wasserstein distortion. *arXiv preprint arXiv:2412.00505*, 2024.

Bégaint, J., Racapé, F., Feltman, S., and Pushparaja, A. Compressai: A PyTorch library and evaluation platform

for end-to-end compression research. *arXiv preprint arXiv:2011.03029*, 2020.

Bellard, F. Bpg image format. *URL https://bellard. org/bpg*, 1(2):1, 2015.

Blard, T., Ladune, T., Philippe, P., Clare, G., Jiang, X., and Déforges, O. Overfitted image coding at reduced complexity. In *2024 32nd European Signal Processing Conference (EUSIPCO)*, pp. 927–931, 2024. doi: 10.23919/EUSIPCO63174.2024.10714961.

Bross, B., Chen, J., Ohm, J.-R., Sullivan, G. J., and Wang, Y.-K. Developments in international video coding standardization after avc, with an overview of versatile video coding (vvc). *Proceedings of the IEEE*, 109(9):1463–1493, 2021a.

Bross, B., Wang, Y.-K., Ye, Y., Liu, S., Chen, J., Sullivan, G. J., and Ohm, J.-R. Overview of the versatile video coding (vvc) standard and its applications. *IEEE Transactions on Circuits and Systems for Video Technology*, 31 (10):3736–3764, 2021b.

Chen, T., Moreau, T., Jiang, Z., Zheng, L., Yan, E., Shen, H., Cowan, M., Wang, L., Hu, Y., Ceze, L., et al. TVM: An automated end-to-end optimizing compiler for deep learning. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pp. 578–594, 2018.

Cheng, Z., Sun, H., Takeuchi, M., and Katto, J. Learned image compression with discretized gaussian mixture likelihoods and attention modules. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020a.

Cheng, Z., Sun, H., Takeuchi, M., and Katto, J. Learned image compression with discretized gaussian mixture likelihoods and attention modules. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 7939–7948, 2020b.

Choi, H. M., Kang, H., and Oh, D. Is overfitting necessary for implicit video representation? In *International Conference on Machine Learning*, pp. 5748–5770. PMLR, 2023.

Cover, T. M. *Elements of information theory*. John Wiley & Sons, 1999.

da Cunha, A., Natale, E., and Viennot, L. Proving the strong lottery ticket hypothesis for convolutional neural networks. In *ICLR 2022-10th International Conference on Learning Representations*, 2022.

Dupont, E., Goliński, A., Alizadeh, M., Teh, Y. W., and Doucet, A. Coin: Compression with implicit neural representations. *ICLR 2021-International Conference on Learning Representations Workshop Neural Compression 2021*, 2021.

Dupont, E., Loya, H., Alizadeh, M., Golinski, A., Teh, Y., and Doucet, A. Coin++: neural compression across modalities. *Transactions on Machine Learning Research*, 2022(11), 2022a.

Dupont, R., Amine Alaoui, M., Sahbi, H., and Lebois, A. Extracting effective subnetworks with gumbel-softmax. In *2022 IEEE International Conference on Image Processing (ICIP)*, pp. 931–935, 2022b. doi: 10.1109/ICIP46576.2022.9897718.

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJl-b3RcF7.

Gisle, B. Calculation of average psnr differences between rd curves. In *ITU-T SG16 Doc. VCEG-M33*, 2001.

Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.

Guo, Z., Flamich, G., He, J., Chen, Z., and Hernández-Lobato, J. M. Compression with Bayesian implicit neural representations. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

Han, S., Mao, H., and Dally, W. J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015a.

Han, S., Pool, J., Tran, J., and Dally, W. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015b.

He, D., Yang, Z., Peng, W., Ma, R., Qin, H., and Wang, Y. Elic: Efficient learned image compression with unevenly grouped space-channel contextual adaptive coding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5718–5727, 2022.

He, J., Flamich, G., Guo, Z., and Hernández-Lobato, J. M. Recombiner: Robust and enhanced compression with Bayesian implicit neural representations. In *International Conference on Learning Representations*, 2024.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Jiang, W., Yang, J., Zhai, Y., Ning, P., Gao, F., and Wang, R. Mlic: Multi-reference entropy model for learned image compression. In *Proceedings of the 31st ACM International Conference on Multimedia*, pp. 7618–7627, 2023.

Kim, H., Bauer, M., Theis, L., Schwarz, J. R., and Dupont, E. C3: High-performance and low-complexity neural compression from a single image or video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9347–9358, 2024.

Kodak. Kodak dataset. 1991. URL http://r0k.us/graphics/kodak/.

Kurtz, M., Kopinsky, J., Gelashvili, R., Matveev, A., Carr, J., Goin, M., Leiserson, W., Moore, S., Nell, B., Shavit, N., and Alistarh, D. Inducing and exploiting activation sparsity for fast inference on deep neural networks. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5533–5543, Virtual, 13–18 Jul 2020. PMLR. URL http://proceedings.mlr.press/v119/kurtz20a.html.

Ladune, T., Philippe, P., Henry, F., Clare, G., and Leguay, T. Cool-chic: Coordinate-based low complexity hierarchical image codec. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 13515–13522, 2023.

Ladune, T., Philippe, P., Clare, G., Henry, F., and Leguay, T. Cool-chic: Perceptually tuned low complexity overfitted image coder. In *2024 Data Compression Conference (DCC)*, pp. 565–565. IEEE, 2024.

Leguay, T., Ladune, T., Philippe, P., Clare, G., Henry, F., and Déforges, O. Low-complexity overfitted neural image codec. In *2023 IEEE 25th International Workshop on Multimedia Signal Processing (MMSP)*, pp. 1–6. IEEE, 2023.

Malach, E., Yehudai, G., Shalev-Schwartz, S., and Shamir, O. Proving the lottery ticket hypothesis: Pruning is all you need. In *International Conference on Machine Learning*, pp. 6682–6691. PMLR, 2020.

Mehta, I., Gharbi, M., Barnes, C., Shechtman, E., Ramamoorthi, R., and Chandraker, M. Modulated periodic activations for generalizable local functional representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14214–14223, 2021.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Miles, R. and Mikolajczyk, K. Cascaded channel pruning using hierarchical self-distillation. In *British Machine Vision Conference (BMVC)*, 2020.

Minnen, D., Ballé, J., and Toderici, G. D. Joint autoregressive and hierarchical priors for learned image compression. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Naumov, M., Chien, L., Vandermersch, P., and Kapasi, U. Cusparse library. In *GPU technology conference*, volume 12, 2010.

Oh, J., Baik, S., and Lee, K. M. Find a winning sign: Sign is all we need to win the lottery. *arXiv preprint arXiv:2504.05357*, 2025.

Orseau, L., Hutter, M., and Rivasplata, O. Logarithmic pruning is all you need. *Advances in Neural Information Processing Systems*, 33:2925–2934, 2020.

Pensia, A., Rajput, S., Nagle, A., Vishwakarma, H., and Papailiopoulos, D. Optimal lottery tickets via subset sum: Logarithmic over-parameterization is sufficient. *Advances in neural information processing systems*, 33: 2599–2610, 2020.

Perez, E., Strub, F., de Vries, H., Dumoulin, V., and Courville, A. C. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018.

Ramanujan, V., Wortsman, M., Kembhavi, A., Farhadi, A., and Rastegari, M. What's hidden in a randomly weighted neural network? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11893–11902, 2020.

Shi, K., Zhou, X., and Gu, S. Improved implicit neural representation with fourier reparameterized training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 25985–25994, 2024.

Sitzmann, V., Martel, J., Bergman, A., Lindell, D., and Wetzstein, G. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020.

Sullivan, G. J., Ohm, J.-R., Han, W.-J., and Wiegand, T. Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on circuits and systems for video technology*, 22(12):1649–1668, 2012.

Toderici, G., Shi, W., Timofte, R., Theis, L., Ballé, J., Agustsson, E., Johnston, N., and Mentzer, F. Workshop and challenge on learned image compression (clic2020). In *CVPR*, 2020.

Ulyanov, D., Vedaldi, A., and Lempitsky, V. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9446–9454, 2018.

Wang, G.-H., Li, J., Li, B., and Lu, Y. Evc: Towards real-time neural image compression with mask decay. *International Conference on Learning Representations*, 2023.

Wortsman, M., Farhadi, A., and Rastegari, M. Discovering neural wirings. *Advances in Neural Information Processing Systems*, 32, 2019.

Yang, Y., Mandt, S., Theis, L., et al. An introduction to neural data compression. *Foundations and Trends® in Computer Graphics and Vision*, 15(2):113–200, 2023.

Zhou, H., Lan, J., Liu, R., and Yosinski, J. Deconstructing lottery tickets: Zeros, signs, and the supermask. *Advances in neural information processing systems*, 32, 2019.

Appendix

A. Quantization and entropy coding methods

For compression, the latent modulation $\mathbf{z}$ and network parameters θ, ψ are quantized into $\hat{\mathbf{z}}$, $\hat{\theta}$ and $\hat{\psi}$, respectively, and subsequently entropy-coded into a bitstream, using standard methods, as in (Kim et al., 2024; Ladune et al., 2024). More details can be seen in the Table 1.

A.1. Latent modulation

Quantization. Similar to (Kim et al., 2024), we adopt a two-stage quantization-aware optimization approach for optimizing $\mathbf{z}$. During the training stage, $\mathbf{z}$ is learned in a continuous space for discrete optimization, with quantization approximated using Kumaraswamy noise. This soft-rounding technique ensures that the quantization process remains differentiable with respect to $\mathbf{z}$. In the inference stage, uniform quantization with hard-rounding is applied to $\mathbf{z}$ as:

$$\hat{\mathbf{z}} = \begin{cases} \mathcal{S}_T(\mathbf{z}) + \mathbf{u}_{kum}, & \text{Training Stage I} \\ Q(\mathbf{z}), & \text{Training Stage I \& Inference Stage,} \end{cases} \tag{12}$$

where $\mathcal{S}_T$ denotes soft-rounding operation with temperature T, Q represents hard-rounding, and $\mathbf{u}_{kum}$ is the Kumaraswamy noise term. The temperature and noise strengths are controlled to shape the Kumaraswamy distribution, transitioning from a peaked form (low noise) at the beginning of the training stage to a uniform distribution by its end.

Entropy coding. Similar to (Ballé et al., 2018; Ladune et al., 2023; Minnen et al., 2018), we introduce a factorized auto-regressive model r_ψ to estimate the distribution of $\hat{\mathbf{z}}$, which is necessary in the entropy coding algorithm. The distribution of each latent element of $\hat{z}_{i,j}$ (the j-th element of $\mathbf{z}_i$) is conditioned on C spatially neighboring elements $\mathbf{c_{i,j}} \in \mathbb{Z}^C$ as:

$$p_\psi(\hat{\mathbf{z}}) = \prod_{i,j} p_\psi(\hat{z}_{i,j}|\mathbf{c_{i,j}}), \tag{13}$$

where $p_\psi(\hat{\mathbf{z}})$ is modeled by an integrated Laplace distribution as

$$p_\psi(\hat{z}_{i,j}|\mathbf{c_{i,j}}) = \int_{\hat{z}_{i,j}-0.5}^{\hat{z}_{i,j}+0.5} g(z)dz. \tag{14}$$

The expectation and scale parameters are estimated via context elements as $g \sim \mathcal{L}(\mu_{i,j}, \sigma_{i,j})$, where $\mu_{i,j}, \sigma_{i,j} = r_\psi(\mathbf{c_{i,j}})$.

With this estimated distribution of $\hat{\mathbf{z}}$, the range coding algorithm is used (range-coder in PyPI), as (Kim et al., 2024) [2], to compress $\hat{\mathbf{z}}$. The resulting rate contributed by $\hat{\mathbf{z}}$ is then given by:

$$R(\hat{\mathbf{z}}) = -\log_2 p_\psi(\hat{\mathbf{z}}) = -\sum_{i,j} \log_2 p_\psi(\hat{z}_{i,j}|\mathbf{c_{i,j}}). \tag{15}$$

A.2. Model compression

The parameters of ModNet and ARM are essential for decoding and are therefore compressed. Specifically, we first quantize ψ, θ using a scalar quantizer $Q(\cdot, \Delta)$ with step size Δ as:

$$\hat{\theta} = Q(\theta, \Delta_\theta), \text{and} \quad \hat{\psi} = Q(\psi, \Delta_\psi). \tag{16}$$

The quantized parameters $\hat{\theta}$ and $\hat{\psi}$ are then entropy-coded, where the discrete distribution of each quantized parameter is modeled by a continuous Laplace distribution. Specifically, the probability of a quantized parameter from θ (similarly for ψ) is given by:

$$p(\hat{\theta}_i) = \int_{\hat{\theta}_i-0.5}^{\hat{\theta}_i+0.5} g(\theta)d\theta, \quad \text{with} \quad g \sim \mathcal{L}(0, \text{stddev}(\hat{\theta})), \tag{17}$$

where g follows a Laplace distribution with zero mean and a standard deviation stddev ($\hat{\theta}$) of the quantized parameters.

[2] An optimized C++ implementation is open-sourced on COOL-CHIC project page.

Then, the total rate contribution from both ARM and ModNet can be expressed as:

$$R_{\text{MLP}} = R_{\hat{\theta}} + R_{\hat{\psi}} = \sum_i -\log_2 p(\hat{\theta}_i) + \sum_i -\log_2 p(\hat{\psi}_i). \tag{18}$$

A greedy search over quantization steps selects optimal Δ_θ and Δ_ψ by minimizing the following rate-distortion cost:

$$\min_{\Delta_\psi, \Delta_\theta} D(\mathbf{S}, g_{\tau \odot \mathbf{w_0}}(f_{\hat{\theta}}(\hat{\mathbf{z}}), \mathbf{x})) + \lambda R, \tag{19}$$

where R is the same as that in Eqn. (5).

B. Implementation details.

B.1. Baseline choices

LotteryCodec is compared against classical codecs, including VTM (Bross et al., 2021a) and HEVC (Sullivan et al., 2012), as well as autoencoder-based neural codecs: CST (Cheng et al., 2020b) (a competitive neural codec), EVC (Wang et al., 2023) (optimized RD performance with low decoding complexity), and MLIC+ (Jiang et al., 2023) (one of the state-of-the-art neural codec). Additionally, we compare with overfitted INR-based codecs such as C3 (Kim et al., 2024) (state-of-the-art overfitted image codec) and COOL-CHICv2 (Leguay et al., 2023) (an optimized version of COOL-CHIC version). We measure PSNR on RGB channels and quantify RD performance using the BD-rate metric (Gisle, 2001). The baseline results were obtained using their official implementations or directly using the reported results (C3, MLIC+) from their papers. For VTM, we use the CompressAI library (Bégaint et al., 2020) for an updated VTM-19.1 (YUV 10 bits) version, where code and datapoints are provided on our project website.

B.2. Datasets

Unless specified otherwise, we use the Kodak dataset and CLIC2020 professional validation sets for evaluation. Specifically, the Kodak dataset consists of 24 images, each with a resolution of 512×768. The CLIC2020 professional validation set includes 41 images with resolutions ranging from 439×720 to 1370×2048.

B.3. Model architecture

We provide the detailed architecture setting for the proposed LotteryCodec scheme in Table 1.

For the ARM model with c contextual elements as input, denoted as ARM-c model, there are three linear or 1×1 convolutional layer, followed by GELU activation functions, with input and output dimension given as $(c \times c) \rightarrow$ GELU $\rightarrow$ $(c \times c) \rightarrow$ GELU $\rightarrow (c \times 2)$. For the proposed LotteryCodec, we employ $c \in \{8, 16, 24, 32\}$.

The ModNet model, using L-dimensional latent modulation as the input, comprising $L_t - 1$ layers ($L_t = 4$ and $L = 7$ in this paper), the input and output dimension are given as: $(7 \times d) \rightarrow$ GELU $\rightarrow (d \times 3) \rightarrow$ GELU $\rightarrow (3 \times 3)$. For the proposed LotteryCodec, we employ $c \in \{32, 48\}$ for Kodak and CLIC2020 dataset.

For the SuperMask model using pixel coordinates as the input, comprising $L_t = 4$ layers, the input and output dimension are given as: $(2 \times 32) \rightarrow$ GELU $\rightarrow ([32 + 3] \times 24) \rightarrow$ GELU $\rightarrow ([24 + 3 + 3] \times 16) \rightarrow ([16 + 3 + 3 + d] \times 3) \rightarrow$ Tanh.

To optimize the RD performance, we can further add a 3×3 convolutional operation in ModNet: $(7 \times d) \rightarrow$ GELU $\rightarrow$ $(d \times 3) \rightarrow$ GELU $\rightarrow (3 \times 3) \rightarrow$ GELU $\rightarrow (3 \times 3)$, where the last two output are concatenated as the first modulation input. As a result, the SuperMask mapping becomes: $(2 \times 32) \rightarrow$ GELU $\rightarrow ([32 + 3 + 3] \times 24) \rightarrow$ GELU $\rightarrow ([24 + 3 + 3 + 3] \times 16) \rightarrow$ $([16 + 3 + 3 + 3 + d] \times 3)$. This additional modification can be omitted if lower complexity is prioritized.

B.4. over-parameterization experiments

This section outlines the detailed experimental settings of Section 4.1.

Datasets. Section. 4.1 trains over-parameterized models with varying mask ratios $\{10\%, 20\%, 30\%, 50\%, 70\%, 90\%\}$ and different $\lambda \in \{2e-2, 1e-2, 5e-3, 1e-3, 5e-4, 2e-4\}$ to compute the BD rate, which yields 36 models for each image. Considering extensive experiments, we use the first 10 images of Kodak dataset to compare the average performance, which indicates that each point in the figure corresponds to an average performance of 360 well-trained INR models.

Hyper parameter	**Initial values**	**Final values**
Values of λ	$\{2e^{-2}, 1e^{-2}, 5e^{-3}, 1e^{-3}, 5e^{-4}, 2e^{-4}\}$	
Quantization – Stage I		
Number of encoding steps	10^5	
Learning rate β	10^{-2}	0
Scheduler for learning rate	Cosine scheduler	
Temperature T for soft rounding	0.3	0.1
Noise strength α for Kumaraswamy noise	2.0	1.0
Scheduler for Soft-rounding and Kumaraswamy noise	Linear scheduler	
Quantization – Stage II		
Number encoding steps	10^4	
Learning rate	10^{-4}	10^{-8}
Decay learning rate if loss has not improved for this many steps	40	
Decay factor	0.8	
Temperature T for soft rounding	10^{-4}	
Architecture – Latent modulations	**Values**	
Number of latent vectors L	7	
Initialization of $\mathbf{z}$	0	
Architecture – ModNet	**Values**	
Output channels of the 1×1 convolutions	$\{48, 3\}$ vs. $\{32, 3\}$	
Number of 3×3 residual convolutions	$\{1\}$ vs $\{2\}$	
Architecture – Entropy model		
Alternative widths of the 3 layers residual 1×1 convolutions (ARM-c)	$c \in \{8/16/24/32\}$	
Activation function	GELU	
Log-scale of Laplace is shifted before $\exp$	4	
Scale parameter of Laplace is clipped to	$[10^{-2}, 150]$	
Architecture – SuperMask network	**Values**	
Output dimensions of MLP layers	$\{32, 24, 16, 3\}$	
Mask ratios	$\{0.1, 0.9\}$ with 0.05 intervals	
Initialization of MLP	FFN initialization	
FFN initialization phase number P	32	
FFN initialization low/high frequency number F	$64/64$	
Initialization of score matrix $\mathbf{P}$	Kaiming uniform initialization	
Learning rate α for $\mathbf{P}$	0.1	
Scheduler	Cosine scheduler	

Table 1. Hyper-parameter settings

Model architecture As illustrated in Fig. 8, we replace the synthesis network of C3 with an over-parameterized network, denoted as C3-Lottery. We present architecture with different over-parameterization levels, as in Table 3 and Table 4.

For a fair comparison, we employ the same ARM model for the C3 in Section 4.1 as the target network. To ensure a fair evaluation of our hypothesis, only the synthesis network is replaced while all other components are kept unchanged for both schemes, as seen in Fig. 8.

Fast implementation. Given that LotteryCodec requires adaptive adjustments to mask ratios and model architectures across images, which may incur extra computational costs, we provide a recommended configuration in Table 2 that achieves near state-of-the-art performance with less adaptation and faster training.

More experimental results. We also provide additional experimental results on the impact of increasing the depth of the over-parameterized network, as shown in Fig. 9. Specifically, Fig. 9a and Fig. 9b present the PSNR vs. rate term of $\hat{\mathbf{z}}$ (bpp) and the corresponding BD-rate performance for randomly initialized networks with hidden dimensions of 48 and 64, respectively. We observe that, in both cases, a deeper network generally increases the likelihood of finding a winning lottery

Architecture – ModNet	**Values**
Output channels of the 1×1 convolutions	$\{48, 3\}$
Number of 3×3 residual convolutions	$\{2\}$
Architecture – Entropy model	
Alternative widths of the 3 layers residual 1×1 convolutions (ARM-c)	$c \in \{16/24\}$
SuperMask network	**Values**
Mask ratios	$\{0.15, 0.4\}$ with 0.05 intervals

Table 2. Fast implementation of LotteryCodec

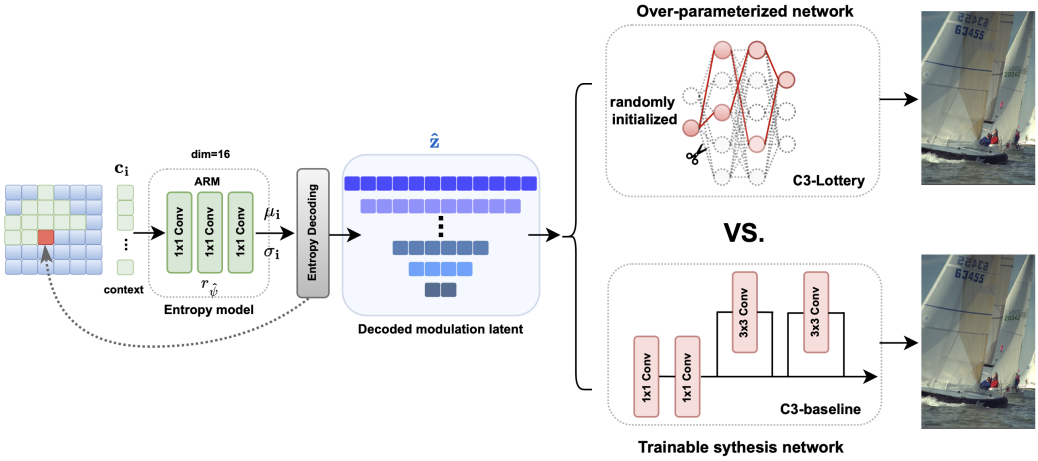


Figure 8. Illustration of the experiments in Section 4.1: the synthesis network is replaced with a randomly initialized over-parameterized network, where only a binary mask is learned, while all other components remain unchanged for a fair comparison.

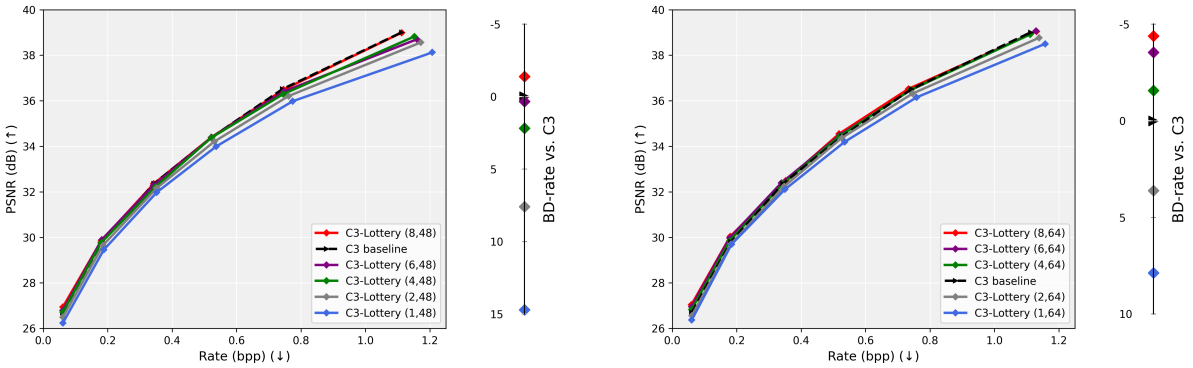
ticket, leading to improved RD performance. Notably, for $d = 64$, lower distortion is achieved with fewer layers.

C3-Lottery model	C3-Lottery $(4, 24)$	C3-Lottery $(4, 32)$	C3-Lottery $(4, 64)$	C3-Lottery $(4, 128)$
Masked linear layers	$(7, 24)$ $(24, 24) \times 4$ $(24, 3)$	$(7, 32)$ $(32, 32) \times 4$ $(32, 3)$	$(7, 64)$ $(64, 64) \times 4$, $(64, 3)$	$(7, 128)$ $(128, 128) \times 4$ $(128, 3)$
Baseline C3	$(7, 18); (18, 3); (3, 3); (3, 3)$			

Table 3. Going wider: Model architectures (input-output channels) for over-parameterization experiments, where all models employ an auto-regressive entropy model (ARM-16). The mask ratios for the experiments are $\{0.1, 0.3, 0.5, 0.7, 0.9\}$.

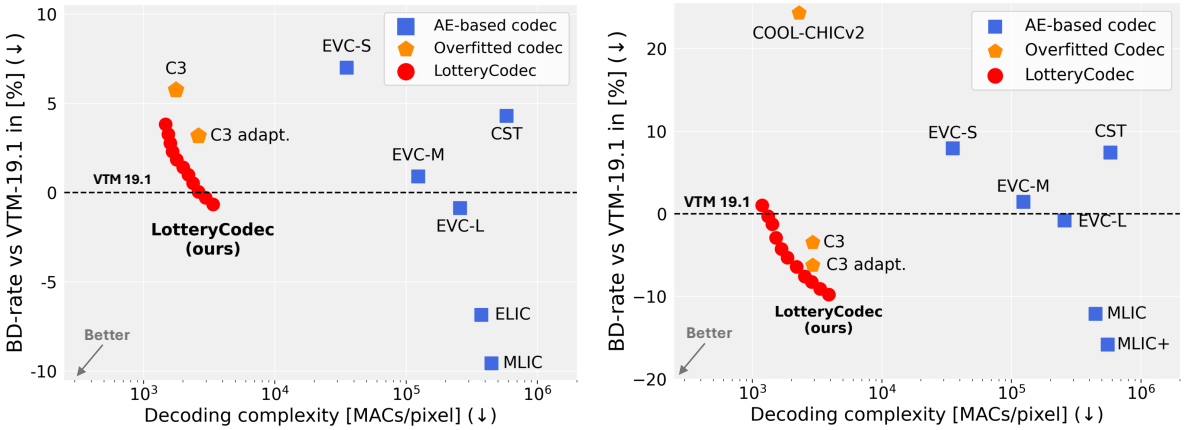
C3-Lottery model	C3-Lottery (1,48)	C3-Lottery (2,48)	C3-Lottery (4,48)	C3-Lottery (6,48)	C3-Lottery (8,48)
Masked linear layers	$(7, 48)$ $(48, 48) \times 1$ $(48, 3)$	$(7, 48)$ $(48, 48) \times 2$ $(48, 3)$	$(7, 48)$ $(48, 48) \times 4$, $(48, 3)$	$(7, 48)$ $(48, 48) \times 6$ $(48, 3)$	$(7, 48)$ $(48, 48) \times 8$ $(48, 3)$
C3-Lottery model	O_{64-1}	O_{64-2}	O_{64-4}	O_{64-6}	O_{64-8}
Masked linear layers	$(7, 48)$ $(64, 64) \times 1$ $(64, 3)$	$(7, 64)$ $(64, 64) \times 2$ $(64, 3)$	$(7, 64)$ $(64, 64) \times 4$, $(64, 3)$	$(7, 64)$ $(64, 64) \times 6$ $(64, 3)$	$(7, 64)$ $(64, 64) \times 8$ $(64, 3)$
Baseline C3	$(7, 18); (18, 3); (3, 3); (3, 3)$				

Table 4. Going deeper: Model architectures (input-output channels) for over-parameterization experiments, where all models employ an auto-regressive entropy model (ARM-16). The mask ratios for the experiments are $\{0.3, 0.5, 0.7\}$.



(a) (b)

Figure 9. Verification of the *lottery codec hypothesis* across varying network depths when hidden dimension is 48 and 64.



(a) Kodak (b) CLIC2020

Figure 10. Performance of LotteryCodec (mask ratios $\in [0.15 : 0.05 : 0.9]$) across different decoding complexities (a) BD-rate across different decoding complexities on Kodak dataset. (b) BD-rate across different decoding complexities on the CLIC2020 dataset.

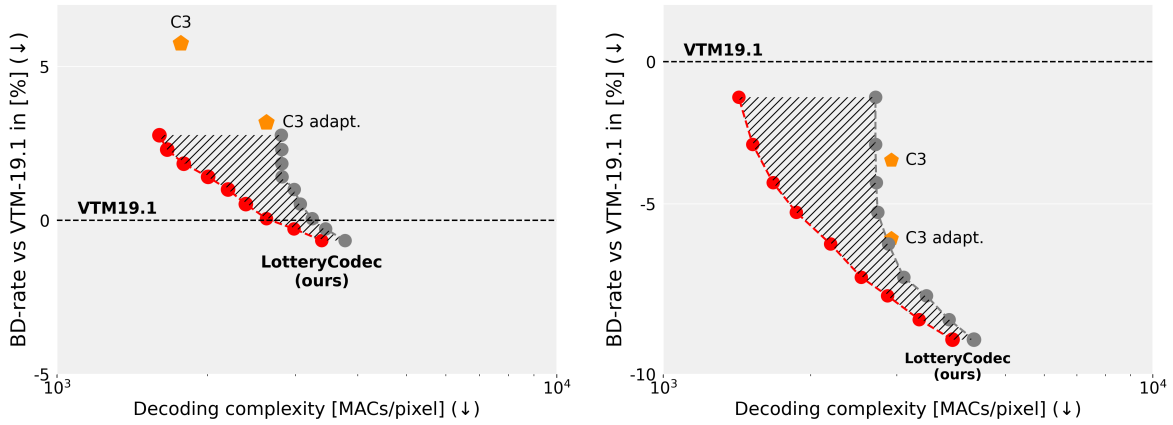
B.5. More experiments: BD rate vs. decoding complexity

We report the BD-rate (vs. VTM 19.1) on the Kodak and CLIC dataset in Fig. 10. We note that VTM configurations vary between implementations (Kim et al., 2024; Blard et al., 2024), and the BD-rate computation depends on both configurations and datapoints. For a fair and more aligned comparison, we update VTM baseline into VTM-19.1 from CompressAI (Bégaint et al., 2020) and recompute BD-rates for all codecs under similar λ settings. We also open-resourced all above baselines and datapoints in our project page for future alignment.

Due to computational constraints, the optimal BD-rate is evaluated over ratio $[0.15, 0.9]$ and $\lambda \in \{1e^{-2}, 5e^{-3}, 1e^{-3}, 5e^{-4}, 2e^{-4}, 1e^{-4}\}$. Other datapoints in Fig. 7 (especially CLIC2020) are trained with a narrower range $[0.15, 0.45]$, yet LotteryCodec still achieves strong performance. Further optimized RD–complexity trade-offs are expected with broader mask ratio selection (see our updates on the project page). Detailed datapoints, lambda, and complexity are provided on our project page.

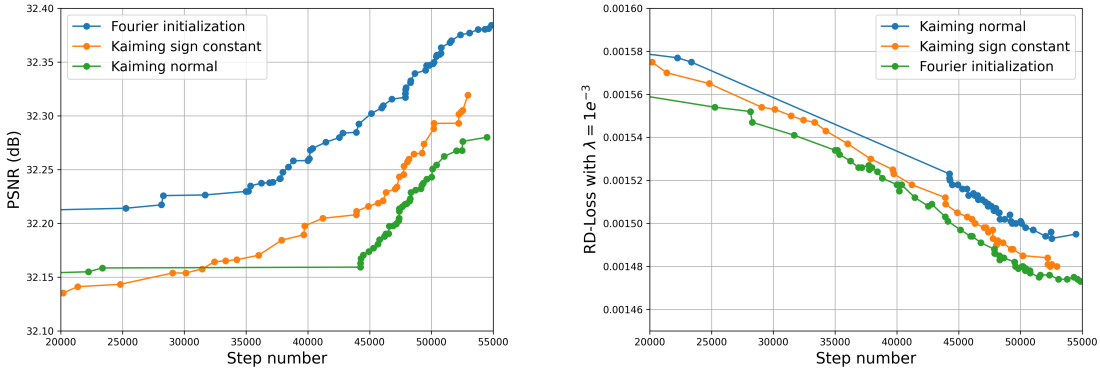
B.6. Theoretical vs. practical decoding complexity

The current figure reports the theoretical minimum decoding complexity, excluding the multiplication operations of masked parameters, an evaluation approach also adopted in (Han et al., 2015a;b). This lower bound can be approached with sparsity-aware libraries (e.g., TVM (Chen et al., 2018), cuSparse (Naumov et al., 2010), DeepSparse (Kurtz et al., 2020)) on



(a) Kodak (b) CLIC2020

Figure 11. Flexible complexity region for Kodak and CLIC2020, where the dashed region is achievable via varying the mask ratios.



(a) (b)

Figure 12. Ablation study on initialization methods: (a) PSNR performance vs. coding step. (b) RD performance vs. coding step. The model is evaluated every 10 coding steps, and the best-performing model at each step is plotted. Results are presented for kodim01 as an example.

appropriate hardware. We adopt this metric because practical MACs per pixel and runtime for unstructured sparse networks are highly dependent on implementation-specific engineering factors, making fair comparisons difficult to conduct.

For a comprehensive analysis, we report real coding times using a simple structured pruning strategy (see Tables 7 8), highlighting LotteryCodec's efficiency, especially on high-resolution images. Additionally, we also provide both theoretical upper and lower bounds on decoding complexity: the upper bound accounts for all operations without pruning, while the lower bound considers only active components. Practical complexity lies between these bounds, depending on implementation. As shown in Fig. 11, even without pruning, LotteryCodec achieves better BD-rate than C3 scheme, with comparable complexity.

C. Qualitative analysis

Justification of Lottery Codec Hypothesis (LCH). Although a rigorous bound supporting the LCH is not available, we can provide a rough validation based on existing proofs for Strong Lottery Tickets Hypothesis (SLTH). Suppose a codec $g_{\mathbf{W}}(\mathbf{z})$ is overfitted to an image $\mathbf{S}$, resulting in distortion σ. According to SLTH, for any $\epsilon > 0$, there exists a subnetwork within a sufficiently over-parameterized network $g_{\mathbf{W}'}$, defined by a supermask τ, such that $d(g_{\mathbf{W}}(\mathbf{z}), g_{\mathbf{W}'\odot\tau}(\mathbf{z})) \leq \epsilon$. Thus,

reconstructing the image $\mathbf{S}$ using $g_{\mathbf{W}'\odot\tau}(\mathbf{z})$ results in a distortion of at most $\sigma+\epsilon$. Now, we can further decrease the distortion by optimizing the latent vector over a set of $\mathbf{z}'$ satisfying $H(\mathbf{z}')=H(\mathbf{z})$, along with the supermask τ. Since ϵ can be made arbitrarily small, it is highly likely that we can find a pair of $(\tau',\mathbf{z}')$ such that $d(\mathbf{S},g_{\mathbf{W}'\odot\tau'}(\mathbf{z}'))\leq\sigma$.

Intuition Behind the Advantages of LotteryCodec. Based on the LCH, we can intuitively justify why the proposed LotteryCodec outperforms previous overfitted codecs in terms of the rate-distortion performance. The rate formulations for overfitted codecs and our LotteryCodec are given in Eqs. (2) and (5), respectively. They show that the rate of overfitted codecs depends on $\{\hat{\mathbf{z}},\hat{\psi},\hat{\mathbf{W}}\}$, while our method is determined by $\{\hat{\mathbf{z}},\hat{\psi},\tau,\hat{\theta}\}$. According to LCH, to achieve the same level of distortion, we can find a pair of $(\hat{\mathbf{z}},\tau)$ such that the bit cost for $\hat{\mathbf{z}}$ and $\hat{\psi}$ is equal to that of overfitted codecs. While each quantized parameter in $\hat{\mathbf{W}}$ typically requires over 13 bits, our binary mask τ uses just 1 bit per entry. Despite its higher dimensionality, τ contributes significantly less to the total rate. Moreover, since $\hat{\theta}$ is lightweight, the combined rate of τ and $\hat{\theta}$ remains lower than that of $\hat{\mathbf{W}}$, resulting in a lower compression rate and improved RD performance.

D. Ablation studies

D.1. Impact of each component in LotteryCodec

To clarify the impact of each component in our design, we conduct an ablation study on each component. As shown in Table 5 below, removing the Supermask network and using only the modulation network increases BD-rate by $+12.45\%$, highlighting the importance of the random network. Removing ModNet and directly feeding $\mathbf{z}$ into the random network (with different overparameterization configurations) results in a performance drop of up to $+14.99\%$ due to high overparameterization costs. Additional ablation studies and visualizations of other components are provided in Table 6.

LotteryCodec	**w/o SuperMask**	Random network **w/o ModNet**: $(4,32)/(4,48)/(4,64)$
0	$+12.45\%$	$+13.02\%/+11.98\%/+14.99\%$

Table 5. Change in BD-rate due to removal of individual components from LotteryCodec, ARM dimension 16.

D.2. MS-SSIM distortion metric

In addition, we evaluate LotteryCodec against baseline methods in terms of MS-SSIM distortion on the Kodak dataset, as shown in Fig. 13. Specifically, we train our method with an MS-SSIM loss with $\lambda\in\{3e^{-1},1e^{-1},3e^{-2},2e^{-2},1e^{-2}\}$. It achieves up to a -43.39% BD-rate reduction over VTM-19.1, closely approaching ELIC and showing a $+10.41\%$ BD-rate gap compared to MLIC+.

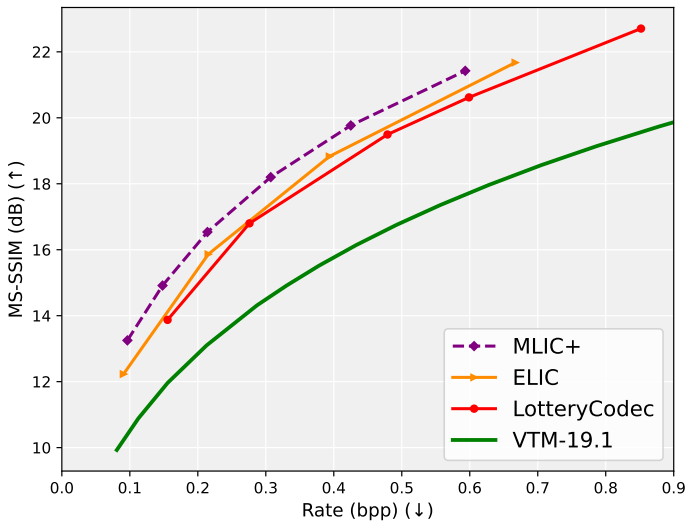


Figure 13. Rate-distortion performance for MS-SSIM metric.

D.3. Other initialization methods

We conduct different initialization methods, the overall BD performance is presented in Table 6. To simplify the validation, here we employ the same training steps of 60000 in Stage I and 6000 for Stage II for all schemes. We also present the PSNR performance and loss values during the coding process in Fig. 12. We can observe that the Fourier initialization we employed can enhance the performance and coding process.

Model Variant	BD rate vs. *LotteryCodec*
LotteryCodec scheme	0.0%
⇒ Kaiming normal initialization (He et al., 2016)	+1.55%
⇒ Signed Kaiming constant initialization (Ramanujan et al., 2020)	+0.63%
⇒ FiLM modulation methods (Perez et al., 2018)	+2.21%
⇒ Score-based same layer masking algorithm (Ramanujan et al., 2020)	+1.77%
⇒ NeRF positional encoding module (Mildenhall et al., 2021)	**Not work well**
⇒ Xavier normal initialization (He et al., 2016)	**Not work well**
⇒ Gumble-softmax for mask ratio learning (Miles & Mikolajczyk, 2020; Dupont et al., 2022b)	**Not work well**
⇒ Bernoulli-based masking algorithm (Zhou et al., 2019)	**Not work well**

Table 6. Ablation study on different mechanisms, conducted on the first 10 Kodak images using an ARM-16 model and a mask ratio of 20%, where $\lambda \in \{2e^{-2}, 1e^{-2}, 5e^{-3}, 1e^{-3}, 2e^{-4}, \}$. A higher BD rate indicates worse RD performance, and "Not work well" signifies lack of robustness and unsatisfactory results

D.4. Other Modulation methods

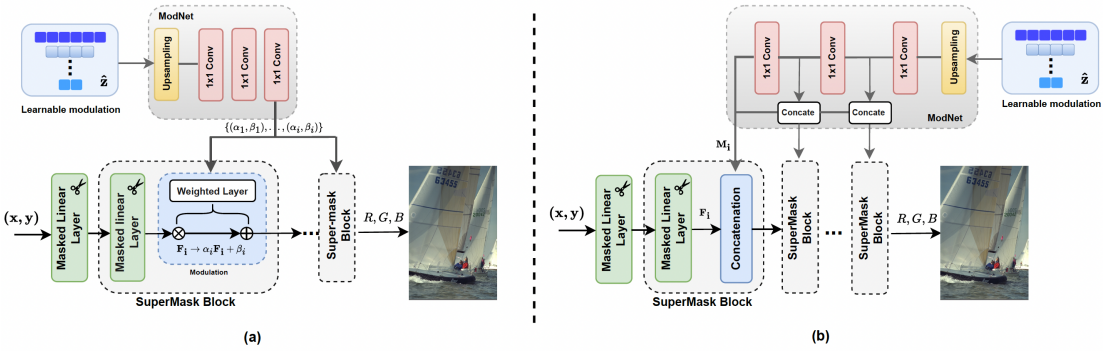


Figure 14. Different alternative modulation methods of the LotteryCodec. (a). A FiLM-based modulation approach. (b). Concatenation-based approach

Note that LotteryCodec is a flexible framework that supports various modulation methods. We also implement a FiLM-based modulation (Perez et al., 2018), as shown in Fig. 14 (a), whose performance is presented in the Table 6. Interestingly, experiments show that additive bias alone achieves competitive performance, although it falls short of the proposed concatenation-based method. This approach, however, provides the potential of reducing inference complexity and exploring alternative modulation strategies in future research.

D.5. Alternative masking learning approach

This section presents various trials we explored. While some did not yield satisfactory results, we include them for completeness and future reference. Specifically, to achieve an adaptive mask ratio, we employed a Gumbel-Softmax scheme for dynamically controlling the pruning rate, following (Miles & Mikolajczyk, 2020; Dupont et al., 2022b), but it did not produce satisfactory results. A similar Bernoulli-based approach, as in (Zhou et al., 2019), also underperformed compared to the current LotteryCodec. Additionally, we evaluated a mask learning algorithm applying a fixed pruning rate across each layer (Ramanujan et al., 2020), which resulted in worse performance. Lastly, positional encoding, similar to (Shi et al., 2024), did not yield performance gains in the context of overfitted codecs.

D.6. Coding latency across various resolutions

We present the coding cost of various schemes in Table 7, and report resolution-dependent coding costs in Table 8. The proposed method shows scalability to ultra-high-resolution images, albeit with increased coding time. An additional example of 2K image encoding is shown in Table 9. Overall, our method has a slightly higher encoding time than other overfitted codecs due to additional gradient-based mask learning, but it offers greater flexibility and faster decoding. Notably, the lottery codec hypothesis provides potential for parallel encoding by re-parameterizing distinct network optimizations into batch-wise mask learning, highlighting its advantage of scalability for efficient large-scale image encoding.

All of these results are based on unoptimized code and current hardware, which can be significantly improved with proper engineering optimization. For example, we can accelerate inference via ONNX and DeepSparse libraries to reduce the decoding time to 20–80 ms on a CPU. Additional techniques (Blard et al., 2024), such as symmetric/separable kernels, filter-based upsampling, and wavefront decoding, can further enhance the speed of overfitted codecs.

Models	Encoding time	Decoding time
VTM 19.1	85.53 (s)	352.52 (ms)
EVC (S/M/L)	20.23/32.21/51.35 (ms)	18.82/23.73/32.56 (ms)
MLIC+	205.60 (ms)	271.31 (ms)
LotteryCodec ($d = 8/16/24$)	13.86/14.64/14.92 (sec/1k steps)	261.33/267.58/278.31 (ms)
C3 ($d = 12/18/24$)	13.10/13.98/14.32 (sec/1k steps)	272.15/284.67/295.03 (ms)

Table 7. Coding time for Kodak images on NVIDIA L40S (GPU) and Intel Xeon Platinum 8358 (CPU) with a masking ratio of 0.8 under structured pruning. Orange indicates GPU computation; blue indicates CPU computation.

	LotteryCodec vs. C3		**LotteryCodec vs. C3 vs. MLIC+**
Input resolution	**GPU Encoding** (sec/1k steps)	**CPU Decoding** (ms)	**Peak Memory usuage during the training** (GB)
512×512	10.71 vs. 10.43	232.46 vs. 228.43	0.56 vs. 0.31 vs. 1.98
1024×1024	56.81 vs. 38.54	565.22 vs. 576.51	2.15 vs. 1.24 vs. 3.61
1536×1536	136.81 vs. 84.79	984.01 vs. 1086.92	4.82 vs. 2.78 vs. 9.15
2048×2048	257.93 vs. 155.02	1595.86 vs. 1807.35	8.53 vs. 4.95 vs. 24.37
2560×2560	407.68 vs. 237.45	3003.24 vs. 3269.02	13.36 vs. 7.72 vs. OM
3840×2160	446.09 vs. 301.56	4014.21 vs. 4216.11	16.89 vs. 9.84 vs. OM

Table 8. Encoding time for images across different resolutions, where mask ratio 0.8 and ARM model of $d = 16$ are employed. OM means out of memory (> 32 GB).

Training steps	**Training time (s)**	**bpp**	**PSNR (dB)**
5k	678	0.24	36.51
10k	1347	0.22	36.92
20k	2685	0.21	37.02
30k	4026	0.20	37.10
50k	6733	0.199	37.14

Table 9. Encoding cost for a 2K image (size 1292×1945), "davide-ragusa-716 in CLIC2020 with optimal result PSNR 37.18 at bpp 0.196" ($d = 24$, ratio 0.2, peak memory 5.64 G), where 10-20k steps can yield a descent performance.

E. Pseudocode for the algorithm

This section provides detailed encoding and decoding algorithm of LotteryCodec, as shown in Algorithm 1 and Algorithm 2.

F. Visualization

Compression Cost. This section visualizes the compression cost of each component in the bitstream of LotteryCodec. As shown in Fig. 15, thanks to the introduction of ModNet, bit cost of the binary mask can be minimal, particularly for

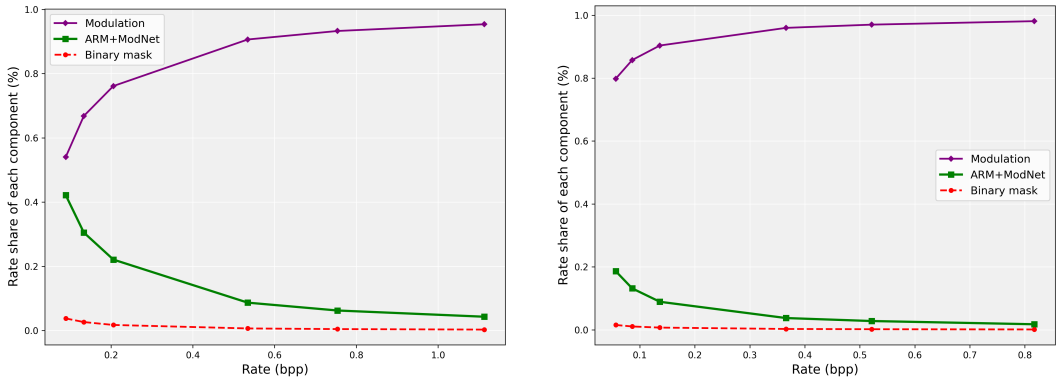


Figure 15. Visualization of the compression cost distribution across the rate within the LotteryCodec scheme using a ARM-24 model and a mask ratio of 0.2: (a) Rate share of compression cost on the Kodak. (b) Rate share of compression cost on the CLIC2020.

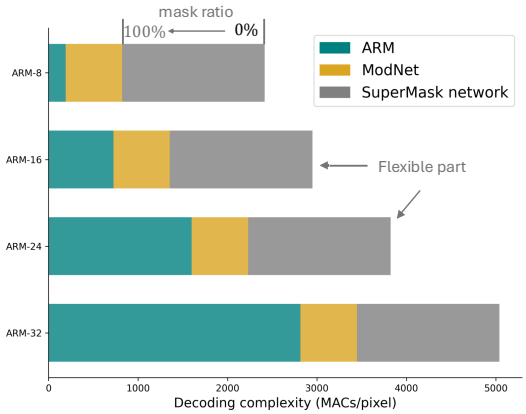


Figure 16. Decoding complexity of each component in LotteryCodec, where ARM-d denotes ARM model with a hidden dimension of d.

higher-resolution images like those in the CLIC2020 dataset, where the same network is used with a lower bpp contribution from binary mask. As bpp increases, modulation bit cost rises, allowing finer image details to be preserved within the given bit budget. Since the network architecture is fixed, then the relative bit cost (%) contribution from the network and binary mask decreases as bpp increases.

Decoding Complexity Analysis. To illustrate the adaptive complexity advantage, we visualize the decoding complexity of each component in the LotteryCodec scheme in Fig. 16. The SuperMask network contributes significantly to the total decoding complexity, which can be adaptively controlled via different mask ratios, enabling flexible complexity management across various regimes. Specifically, for LotteryCodec using ARM-8, the SuperMask accounts for more than 60% of the total complexity, demonstrating its remarkable adaptability and efficiency, particularly in low-bpp regimes.

Effect of modulations in different SuperMask layers. We visualize the effect of different modulations from each ModNet layer in Fig. 17, where we employ a three-layer ModNet and a four-layer SuperMask network. When considering the mask ratio: a lower mask ratio tends to require more complex modulations to aid the LotteryCodec to search the subnetwork, which is reflected in the increased variability of signs and entropy, leading to stronger representational ability. The variability of signs also indicates a greater dependence on the randomly initialized network, especially with a low mask ratio. At lower mask ratios and lower compression rates, the ModNet output also exhibits a more complex distribution, demonstrating the utilization of the random network for representation.

Additionally, the shallow ModNet layers play a more crucial role, as they are directly connected to the deeper SuperMask

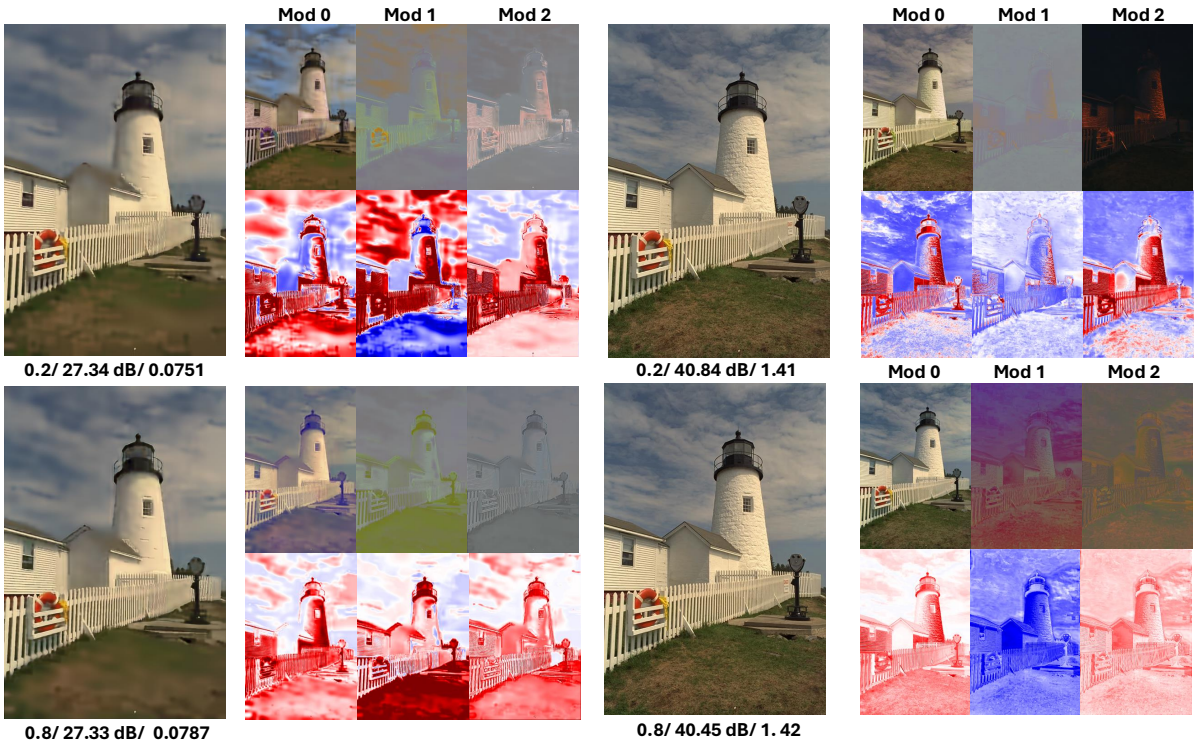


Figure 17. Visualization of ModNet outputs and their effects in LotteryCodec scheme, where $0.2/27.34/0.0751$ represent reconstruction with a 0.2 mask ratio, 27.34 PSNR, and 0.0751 bpp. Mod i represents the visualization of the i-th layer of ModNet. The first row shows reconstructions with all but the selected latent outputs set to zero, while the second row displays the averaged feature outputs, upscaled to match the resolution of the reconstructed image. Visualizations are provided for both low bit-rate (a) and high bit-rate (b) scenarios.

layers without rewinding, significantly influencing the synthesis process. In contrast, the deeper ModNet layers provide coarser information, such as lighting and overall structure, which is repeatedly fed into different layers in a rewind fashion.

Effect of different input latent modulations. We also visualize the effect of the input latent modulations in Fig. 18. As shown in the figure, we observe that the highest resolution latent grid primarily captures luminance and structural details of the image, while lower resolution latent grids provide complementary information for the overall image reconstruction.

Effect of latent modulations across different bit rates. Comparing Fig. 18 (a) and Fig. 18 (b), we observe that in low-bpp scenarios, lower-resolution modulations play a dominant role in reconstruction, while higher-resolution modulations contribute minimally. This highlights an RD trade-off, where lower-resolution modulations prioritize rate efficiency at the cost of higher distortion. Conversely, in high-bpp scenarios, reconstruction is primarily influenced by higher-resolution modulations, capturing finer image details.

Effect of different mask ratios. Comparing a mask ratio of 0.2 with 0.8, we observe that a lower mask ratio distributes more details across each layer of modulation (from $\mathbf{z_1}$ to $\mathbf{z_7}$), allowing the synthesis process to utilize richer structural information. In contrast, at a higher mask ratio of 0.8, the generation process is predominantly influenced by high-resolution latent representations, with less contribution from lower-resolution modulations.

Interpretation of the design. From the above visualizations, we interpret modulation as additional "bias", where concatenating neurons with a learned weight mask is equivalent to adding a flexible bias to the subsequent layers. This design demonstrates that the SuperMask network aims to encode images into the network structure, while ModNet introduces a flexible "adaptive bias" to enhance representation flexibility. As shown in Fig. 17, modulation outputs from different ModNet layers contribute varying compensatory information for image reconstruction.

Algorithm 1 Encoding stage of the LotteryCodec

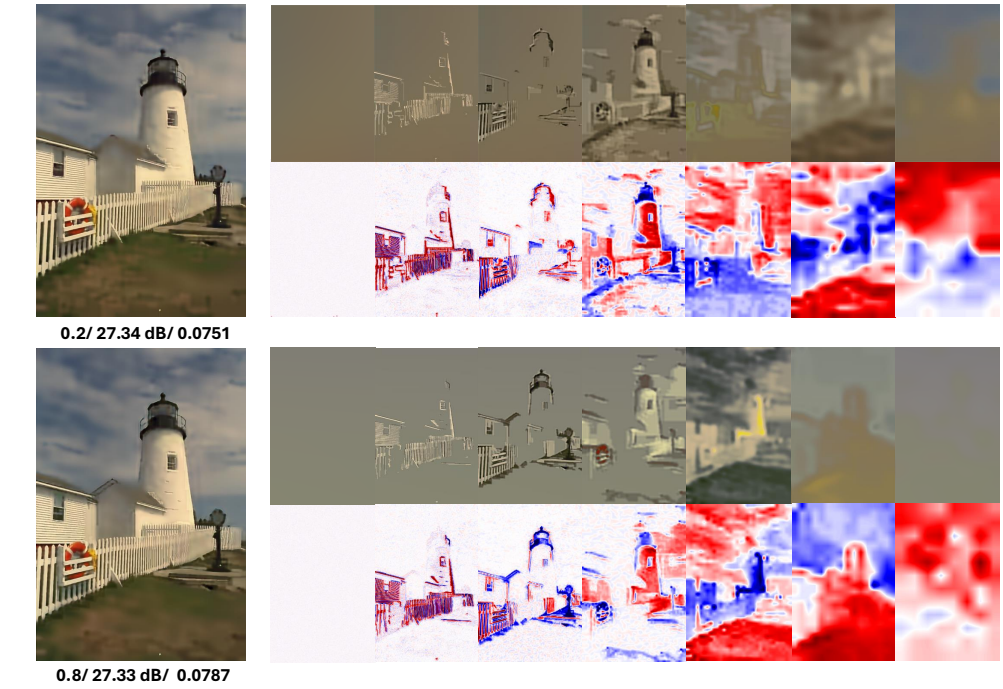
Input: Source image $\mathbf{S}$, coordinate vector $\mathbf{x}$, random seed e,
learning rates α for mask scores ($\mathbf{P}$), β for modulations ($\mathbf{z}$), and networks (ModNet $\theta$and ARM ψ).
cosine scheduler for learning rate for Stage I and II: $\mathcal{C}_1$, $\mathcal{C}_2$,
linear scheduler for soft-rounding temperature and Kumaraswamy noise strength for Stage I and II: l_1, l_2.
Greedy search for quantization step: $\mathcal{G}$.
Output: Bits stream of $\mathbf{z}$, ψ, θ, τ: $\mathbf{b}_\mathbf{z}$, $\mathbf{b}_\psi$, $\mathbf{b}_\theta$, $\mathbf{b}_\tau$.

$\mathbf{W_0} = \mathbf{\Lambda B}$, *// Fourier initialization for $g(\cdot)$ based on e*
$\mathbf{P} \sim \mathcal{U}_k$, *// Kaiming uniform initialization*
for the i-th step within the Stage I **do**
  $\tau = h(\mathbf{P})$, $g = g_{\tau \odot \mathbf{w_0}}$ *// Configure network by masking $k\%$ weights based on $\mathbf{P}$*
  $\hat{\mathbf{z}} = \mathcal{S}_T(\mathbf{z}) + \mathbf{u}_{kum}$, *// Quantization-aware training*
  $\hat{\mathbf{S}} = g_{\tau \odot \mathbf{w_0}}(f_\theta(\hat{\mathbf{z}}), \mathbf{x})$, *// Modulate the reconstruction*
  $\mathcal{L} = D(\mathbf{S}, \hat{\mathbf{S}}) + \lambda R_\psi(\hat{\mathbf{z}})$, *// Compute the RD cost loss function*
  $\mathbf{P} \leftarrow \mathbf{P} - \alpha \nabla_\mathbf{P} \mathcal{L}$, *// Update the scores for mask*
  $\mathbf{z} \leftarrow \mathbf{z} - \beta \nabla_\mathbf{z} \mathcal{L}$, *// Update latent modulations*
  $\theta \leftarrow \theta - \beta \nabla_\theta \mathcal{L}$, *// Update ModNet*
  $\psi \leftarrow \psi - \beta \nabla_\psi \mathcal{L}$, *// Update ARM*
  $\alpha = \mathcal{C}_1(\alpha, i)$, $\beta = \mathcal{C}_1(\beta, i)$ *// Update learning rate*
  $T = l_1(T, i)$, $u_{kum} = l_1(u_{kum}, i)$ *// Update noise strength*
end for
for the i-th step within the Stage II **do**
  $\tau = h(\mathbf{P})$, $g = g_{\tau \odot \mathbf{w_0}}$ *// Mask $k\%$ weights based on updated $\mathbf{P}$*
  $\hat{\mathbf{z}} = Q(\mathbf{z})$, *// Hard rounding*
  $\hat{\mathbf{S}} = g_{\tau \odot \mathbf{w_0}}(f_\theta(\hat{\mathbf{z}}), \mathbf{x})$, *// Modulate the reconstruction*
  $\mathcal{L} = D(\mathbf{S}, \hat{\mathbf{S}}) + \lambda R_\psi(\hat{\mathbf{z}})$, *// Compute the RD cost loss*
  $\mathbf{P} \leftarrow \mathbf{P} - \alpha \nabla_\mathbf{P} \mathcal{L}$, *// Update the scores for mask*
  $\mathbf{z} \leftarrow \mathbf{z} - \beta \nabla_\mathbf{z} \mathcal{L}$, *// Update latent modulations*
  $\theta \leftarrow \theta - \beta \nabla_\theta \mathcal{L}$, *// Update ModNet*
  $\psi \leftarrow \psi - \beta \nabla_\psi \mathcal{L}$, *// Update ARM*
  $\alpha = \mathcal{C}_2(\alpha, i)$, $\beta = \mathcal{C}_2(\beta, i)$ *// Update learning rate*
  $T = l_2(T, i)$, $u_{kum} = l_2(u_{kum}, i)$ *// Update noise strength*
end for
$\mathbf{b}_\mathbf{z} = \mathcal{A}(Q(\mathbf{z}))$ *// Bit stream of $\mathbf{z}$ after quantization and entropy coding*
$\mathbf{b}_\tau = \mathcal{A}(\tau)$ *// Bit stream of τ after entropy coding*
$\Delta_\theta, \Delta_\psi = \mathcal{G}(\theta, \psi)$ *// Search for a optimal quantization step for networks*
$\mathbf{b}_\theta = \mathcal{A}(Q(\theta, \Delta_\theta))$ *// Bit stream of θ after quantization and entropy coding*
$\mathbf{b}_\psi = \mathcal{A}(Q(\psi, \Delta_\psi))$ *// Bit stream of ψ after quantization and entropy coding*

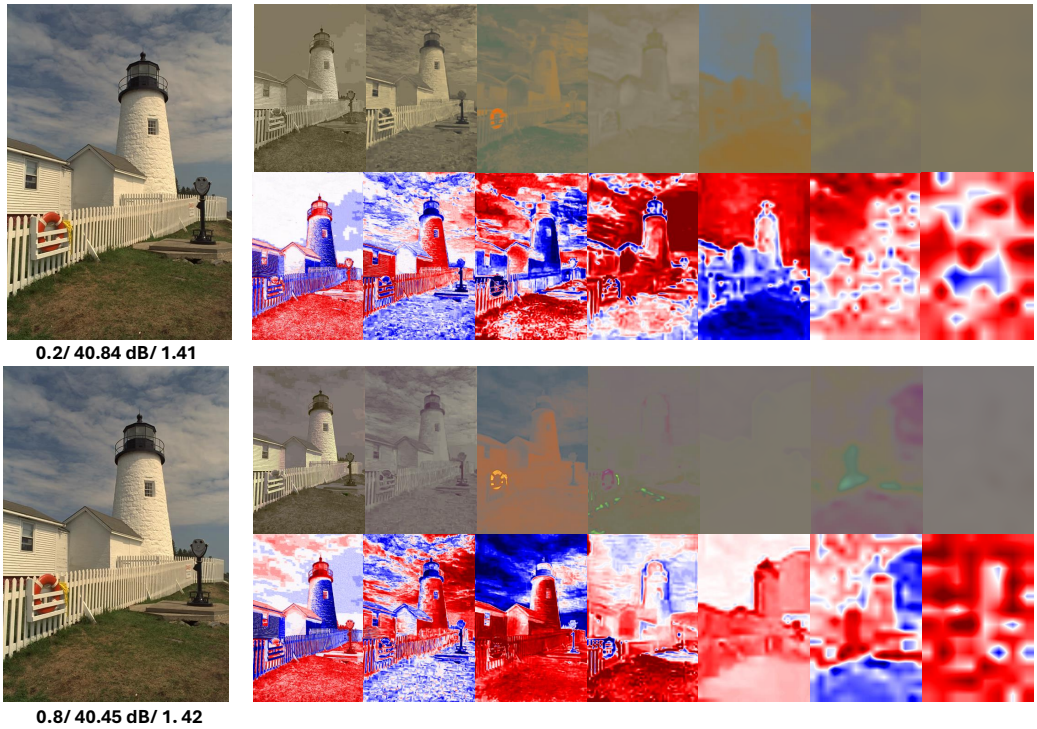
Algorithm 2 Decoding stage of the LotteryCodec

Input: Coordinate vector $\mathbf{x}$, random seed e, Bits stream: $\mathbf{b}_\mathbf{z}$, $\mathbf{b}_\psi$, $\mathbf{b}_\theta$, $\mathbf{b}_\tau$.
Output: Reconstruction of image $\hat{\mathbf{S}}$.

$\mathbf{W_0} = \mathbf{\Lambda B}$, *// Fourier initialization for g based on a given seed e.*
$\tau = \mathcal{A}(\mathbf{b}_\tau)$, $g = g_{\tau \odot \mathbf{w_0}}$ *// Entropy decode mask and configure SuperMask network locally*
$\hat{\psi} = Q(\mathcal{A}(\mathbf{b}_\psi))$, $\hat{\theta} = Q(\mathcal{A}(\mathbf{b}_\theta))$, *// Entropy decode and de-quantize parameters $\hat{\psi}$ and $\hat{\theta}$*
$\hat{\mathbf{z}} = \mathcal{A}(f_{\hat{\psi}}, \mathbf{b}_\mathbf{z})$, *// Entropy decode $\mathbf{z}$*
$\hat{\mathbf{S}} = g_{\tau \odot \mathbf{w_0}}(f_{\hat{\theta}}(\hat{\mathbf{z}}), \mathbf{x})$, *// Reconstruct the source image*



(a)



(b)

Figure 18. Visualization of the LotteryCodec scheme using kodim19 as an example, where $0.2/27.34/0.0751$ represent reconstruction with a 0.2 mask ratio, 27.34 PSNR, and 0.0751 bpp. From left to right: reconstructed image followed by latent modulations $\mathbf{z_1}$ to $\mathbf{z_7}$, arranged from high to low resolution. The first row shows reconstructions where all but $\mathbf{z_i}$ latent are set to zero. The second row displays the raw latents, upscaled to match the output resolution. (a). Reconstruction and visualization at a low bit-rate (around 0.078 bpp). (b). Reconstruction and visualization at a high bit-rate (around 1.41 bpp).