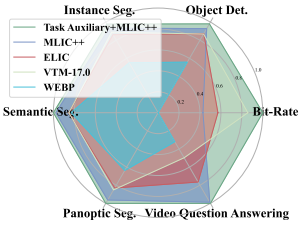
WHICH TASKS SHOULD BE COMPRESSED TOGETHER? A CAUSAL DISCOVERY APPROACH FOR EFFICIENT MULTI-TASK REPRESENTATION COMPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

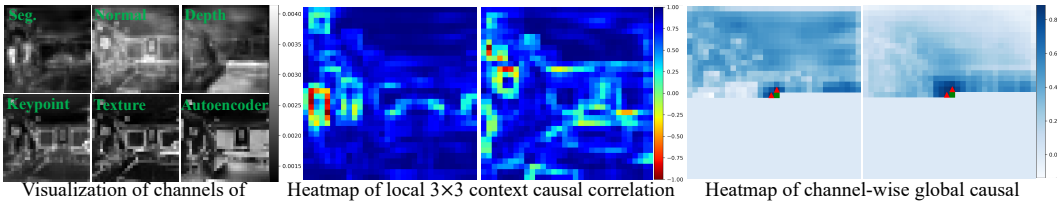
Conventional image compression methods are inadequate for intelligent analysis, as they overemphasize pixel-level precision while neglecting semantic significance and the interaction among multiple tasks. This paper introduces a Taskonomy-Aware Multi-Task Compression framework comprising (1) inter-coherent task grouping, which organizes synergistic tasks into shared representations to improve multi-task accuracy and reduce encoding volume, and (2) a conditional entropy-based directed acyclic graph (DAG) that captures causal dependencies among grouped representations. By leveraging parent representations as contextual priors for child representations, the framework effectively utilizes cross-task information to improve entropy model accuracy. Experiments on diverse vision tasks, including Keypoint 2D, Depth Zbuffer, Semantic Segmentation, Surface Normal, Edge Texture, and Autoencoder, demonstrate significant bitrate-performance gains, validating the method's capability to reduce system entropy uncertainty. These findings underscore the potential of leveraging representation disentanglement, synergy, and causal modelling for compact representation learning, enabling efficient multi-task compression in intelligent systems. Code will be available.



(a)

		Relative Performance Inferenced On					
		ObjectDet.	InstanceSeg.	PanoSeg.	Keypoints	Classifier	SemSeg.
Trained With	ObjectDet.	5.19%	5.13%	3.34%	5.57%	-0.39%	1.62%
	InstanceSeg.	4.90%	5.50%	5.20%	3.00%	-1.00%	2.10%
	PanoSeg.	4.70%	5.10%	5.70%	3.10%	-1.20%	4.50%
	Keypoints	4.00%	3.80%	3.90%	5.80%	1.50%	3.10%
	Classifier	4.50%	3.20%	2.80%	3.60%	5.50%	2.00%
	SemSeg.	3.70%	4.20%	4.60%	2.90%	2.10%	5.70%

(b)



Visualization of channels of latent representation

Heatmap of local 3×3 context causal correlation of Semantic Seg. and Surface Normal

Heatmap of channel-wise global causal correlation of Semantic Seg. and Surface Normal

(c)

Figure 1: **Motivation of Taskonomy-Aware Multi-Task Compression. (a)** *Multi-task Compression Performance*: Normalized performance (0=worst, 1=best for clarity) of compressors in diverse tasks. **(b)** *Necessity of Task Grouping*: Relative performance (%) when compressors are trained on one task and tested on another. Positive values indicate collaboration, negative values highlight conflicts, underscoring the need for task grouping. **(c)** *Necessity of Taskonomy-Aware Causal Modeling:* Visualization of local context correlation (middle) and global channel-wise correlation (right) of Semantic Seg. and Surface Normal tasks, indicating the potential for capturing redundancy and achieving bit savings through task-aware causal modeling. The ■ indicates the current decoding point, while the ▲ represents the most similar reference positions.

1 INTRODUCTION

Multimodal models like CLIP (Radford et al., 2021), GPT-4 (Achiam et al., 2023), and Sora (Liu et al., 2024a) exhibit human-level comprehension and reasoning(Achiam et al., 2023; Chang et al., 2024; Zheng et al., 2023; Laskar et al., 2023), making them potential consumers of visual and multimedia content. This highlights the need for semantic representation compression to support efficient multi-task processing. However, current compression techniques, including both handcrafted video codecs (Bross et al., 2021; Pennebaker & Mitchell, 1992; Si & Shen, 2016) and end-to-end learning-based approaches (Jiang et al., 2023; He et al., 2022; Zou et al., 2022; Chen et al., 2021; Ballé et al., 2017), primarily focus on rate-distortion optimization (Shannon et al., 1959), *i.e.*, constraining the entropy model to accurately estimate the probability distribution of latent space symbols while simultaneously maximizing the pixel-level likelihood between the reconstructed and original images. Nevertheless, conventional compression methods lack semantic representation constraints, limiting their ability to preserve task-relevant information while reducing redundancy. As illustrated in Fig. 1a, assessment on MS-COCO (Lin et al., 2014) tasks with images compressed at 0.15 bpp shows learning-based methods MLIC++ (Jiang et al., 2023) and ELIC (He et al., 2022) exhibited a notable superiority compared to the handcrafted WebP (Si & Shen, 2016) and VTM-17.0 (Bross et al., 2021), particularly in multimodal tasks like Video Question Answering (Zhang et al., 2023), highlighting their potential to capture richer semantics for intelligent multimedia analysis.

In response to these limitations, the paradigm of Video Coding for Machines (VCM) (Yang et al., 2024; Choi & Bajić, 2022; Duan et al., 2020) has emerged as a promising solution. VCM integrates image compression with feature representation to achieve both compactness and efficiency, aiming to meet the dual objectives of high-fidelity human vision and high-precision machine vision(Ge et al., 2024; Li et al., 2024; Liu et al., 2023b; Bai et al., 2022).

Despite advancements in VCM, most methods still treat tasks in isolation (Liu et al., 2021; Bai et al., 2022) or focus only on predefined tasks (Liu et al., 2023b; Li et al., 2024), overlooking the benefits of grouping supportive tasks or the complex relationships across task feature spaces. Preliminary studies (Shi et al., 2023; Fifty et al., 2021; Standley et al., 2020; Zamir et al., 2018) and our findings in Fig. 1b reveal statistically significant correlations among tasks, both positive and negative. These correlations highlight the synergies and conflicts in multi-task compression. Therefore, identifying task groups that leverage synergies and reduce conflicts is essential for improved multi-task learning and optimized compression performance. Furthermore, Fig. 1c demonstrates the existence of local and global causal contextual relationships across tasks with different semantic granularities. Exploiting conditional relationships in representations enables more accurate prediction of symbol distributions through conditional entropy, thereby improving compression efficiency. Conventional methods, however, process tasks independently, ignoring inter-task dependencies, which leads to redundancy.

Thus, an essential question arises: *How can we discern and exploit the interdependencies among tasks to achieve efficient multi-task representation compression?* Addressing this question requires models that can (1) discern mutually beneficial and conflicting tasks to group them effectively, and (2) disentangle intricate sub-task relationships.

To address this challenge, we propose a paradigm shift with *Taskonomy-Aware Multi-Task Compression* (TAMC), which integrates task grouping and causal discovery for compact multi-task representation compression. By leveraging causal relationships between tasks, TAMC enhances task performance and improves overall compression efficiency. Our approach consists of two key components. First, we cluster inter-coherent tasks into groups that share a universal representation, leveraging synergies among mutually beneficial tasks to improve accuracy and reduce encoding volume. This is the first work to systematically group tasks for compression, mitigating conflicts and preventing performance degradation. Second, we construct a directed acyclic graph (DAG) based on conditional entropy to capture causal relationships among task representations. This approach identifies task dependencies across different abstraction levels, uncovering inter-task relationships and mapping information flow through directed graphs. By traversing causal paths, parent task representations provide informative cross-task contexts for child tasks, as dedicated in Fig. 2, reducing uncertainty, improving compression efficiency, and enhancing scalability.

Our contributions: 1. We demonstrate that leveraging intricate inter-task relationships significantly improves rate-performance efficiency by clustering tasks for collective compression and establishing causal links between clusters. 2. We introduce a DAG-based causal discovery framework via conditional entropy, which captures semantic dependencies across abstraction levels to enhance

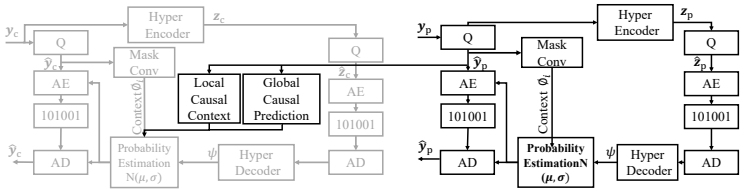


Figure 2: The proposed cross-task causal context compression framework. Gray components represent the original model, and black solid-lined components show the proposed extensions. The parent $\boldsymbol{y}_p$ provides an adjacent local and global causal context for the child $\boldsymbol{y}_c$.

system certainty and reduce information entropy, thereby improving compression compactness. 3. The proposed TAMC achieves superior performance across diverse downstream task benchmarks while remaining competitive in universal image reconstruction. Extensive experiments on key computer vision tasks using the Taskonomy dataset validate the effectiveness of our approach.

2 Related Work

Multi-Task Learning. Multi-task learning (MTL) improves performance by introducing inductive biases and emphasizing relevant features (Zhang & Yang, 2021). However, task competition for model capacity and ineffective shared representations often hinder MTL. Grouping compatible tasks is crucial for reducing conflicts and boosting performance (Lu et al., 2020; Yu et al., 2020; Chen et al., 2020; Kendall et al., 2018), yet current approaches often rely on human intuition (Zhang & Yang, 2021). Recent studies (Fifty et al., 2021; Wu et al., 2020; Standley et al., 2020) highlight the need for systematic task grouping to advance the field.

End-to-End Image Compression for Human and Machine Tasks. Gray components in Fig. 2 provide a high-level overview of E2E-learned image compression (Ballé et al., 2017), an image $\boldsymbol{x}$ is first encoded into latent representations $\boldsymbol{y}$ using an analysis transform $g_a(\boldsymbol{x};\boldsymbol{\theta}_e)$, then quantized to discrete values $\hat{\boldsymbol{y}}$. With a learned probability model $p_{\hat{\boldsymbol{y}}}(\hat{\boldsymbol{y}})$, $\hat{\boldsymbol{y}}$ can be losslessly coded using arithmetic coding. On the decoder side, a synthesis transform $g_s(\hat{\boldsymbol{y}};\boldsymbol{\theta}_d)$ reconstructs the image $\hat{\boldsymbol{x}}$ from $\hat{\boldsymbol{y}}$:

$$\boldsymbol{y} = g_a(\boldsymbol{x};\boldsymbol{\theta}_e),\ \hat{\boldsymbol{y}} = Q(\boldsymbol{y}),\ \hat{\boldsymbol{x}} = g_s(\hat{\boldsymbol{y}};\boldsymbol{\theta}_d). \tag{1}$$

To improve compression efficiency by decorrelating the latent space and estimating symbol probabilities, Ballé et al. (2018) introduces a hyperprior model that reduces spatial redundancies among latent variables, adding a few extra bits to convey spatial structure. This hyperprior model enables a more accurate entropy model and better estimation of $p_{\hat{\boldsymbol{y}}}(\hat{\boldsymbol{y}})$. It can be divided into a hyper analysis transform $h_a(\boldsymbol{y};\boldsymbol{\theta}_{he})$ and a synthesis transform $h_s(\hat{\boldsymbol{z}};\boldsymbol{\theta}_{hd})$:

$$\boldsymbol{z} = h_a(\boldsymbol{y};\boldsymbol{\theta}_{he}),\ \hat{\boldsymbol{z}} = Q(\boldsymbol{z}),\ p_{\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}) = h_s(\hat{\boldsymbol{z}};\boldsymbol{\theta}_{hd}). \tag{2}$$

Minnen et al. (2018) proposed a more accurate entropy model which jointly utilizes an autoregressive context model g_{cm}. The predicted Gaussian parameters $N(\mu,\sigma)$ of the distribution $p_{\hat{\boldsymbol{y}}}(\hat{\boldsymbol{y}})$ are functions of the learned parameters of the hyper-decoder, context model, and entropy parameter networks ($\boldsymbol{\theta}_{hd}$, $\boldsymbol{\theta}_{cm}$, and $\boldsymbol{\theta}_{ep}$, respectively):

$$p_{\hat{\boldsymbol{y}}}(\hat{\boldsymbol{y}} \mid \hat{\boldsymbol{z}}, \boldsymbol{\theta}_{hd}, \boldsymbol{\theta}_{cm}, \boldsymbol{\theta}_{ep}) = \prod_i \left(\mathcal{N}(\mu_i, \sigma_i^2) * \mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right)\right)(\hat{\boldsymbol{y}}_i), \tag{3}$$

with $\mu_i, \sigma_i = g_{ep}(\psi, \phi_i; \boldsymbol{\theta}_{ep})$, $\psi = g_h(\hat{\boldsymbol{z}};\boldsymbol{\theta}_{hd})$, and $\phi_i = g_{cm}(\hat{\boldsymbol{y}}_{<i};\boldsymbol{\theta}_{cm})$, $\mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right)$ is a uniform noise to approximate quantization during training. The overall loss function is:

$$\mathcal{L} = \mathcal{R}(\hat{\boldsymbol{y}}) + \mathcal{R}(\hat{\boldsymbol{z}}) + \lambda \cdot \mathcal{D}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \underbrace{\mathbb{E}\left[-\log_2(p_{\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}))\right]}_{\text{rate(latents)}} + \underbrace{\mathbb{E}\left[-\log_2(p_{\hat{\boldsymbol{z}}}(\hat{\boldsymbol{z}}))\right]}_{\text{rate(hyper-latents)}} + \lambda \cdot \underbrace{\mathcal{D}(\boldsymbol{x}, \hat{\boldsymbol{x}})}_{\text{distrotion}}, \tag{4}$$

where λ controls the rate-distortion tradeoff. The first term is the rate that corresponds to the cross entropy between the natural (marginal) distribution and the learned entropy model. The second term is the rate to transmit hyperprior. The third term measures the reconstruction quality according to the given distortion metric d (*e.g.*, PSNR or MS-SSIM). Recent advancements include architectures such

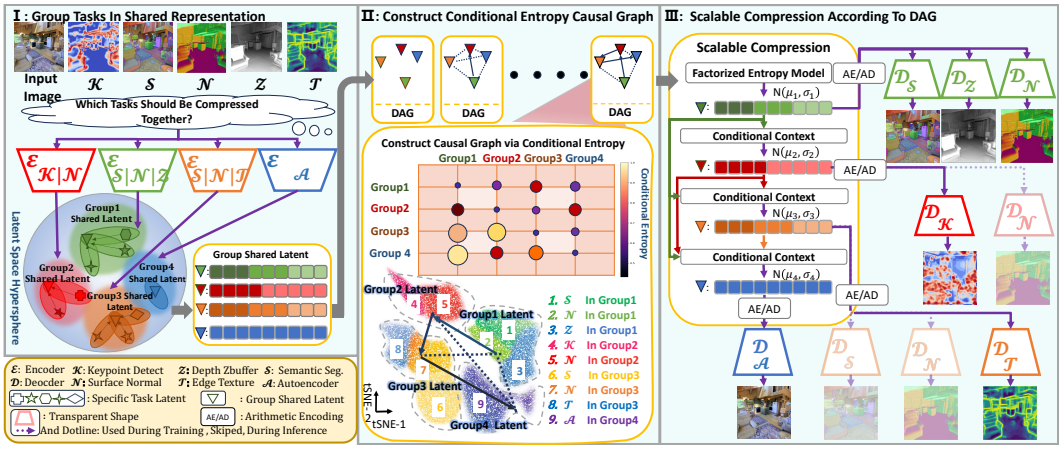


Figure 3: Overview of TAMC. **Given a series of tasks, how can we effectively cluster them in the latent semantic space and construct a causal graph to optimize bitrate-performance simultaneously?** To find a feasible solution, we follow these steps: (**I**): Group tasks based on inter-task coherence under bitrate constraints. (**II**): Construct a DAG via conditional entropy to capture causal relationships. (**III**): Compress grouped representations according to DAG in a scalable manner.

as residual networks (Theis et al., 2017; Mentzer et al., 2018; Rippel & Bourdev, 2017; Nakanishi et al., 2019; Cheng et al., 2020; He et al., 2022), self-attention (Chen et al., 2021; Cheng et al., 2020; Guo et al., 2021), ConvNext (Duan et al., 2023), invertible modules (Xu & Zhang, 2021; Xie et al., 2021; Ma et al., 2020), Generative Adversarial Networks (GANs)(Agustsson et al., 2019; Santurkar et al., 2018; Tschannen et al., 2018; Mentzer et al., 2020), and transformers(Lu et al., 2022; Xiang et al., 2022; Zhu et al., 2021; Zou et al., 2022; Liu et al., 2023a). These advancements, combined with improved entropy models (e.g., hierarchical (Ballé et al., 2018), auto-regressive (Minnen et al., 2018; He et al., 2021; Xiang et al., 2022)), multireference entropy (Qian et al., 2020), and innovations such as channel-wise and spatial-wise acceleration (Minnen & Singh, 2020; He et al., 2021; Jiang et al., 2023; He et al., 2022), codebooks and vector quantization (Zhu et al., 2022), and hierarchical VAEs (Duan et al., 2023), have significantly improved compression performance. Many existing techniques overlook their impact on downstream tasks like classification, detection, and segmentation, which require task-specific feature retention. Task-aware paradigms such as Video Coding for Machines (VCM)(Yang et al., 2024; Choi & Bajić, 2022) focus on machine vision-targeted compression(Li et al., 2024; Liu et al., 2023b), including *feature-assisted coding*(Liu et al., 2024b), *scalable coding*(Liu et al., 2021), and *intermediate feature compression*(Kim et al., 2023; Chen et al., 2021). While effective for individual tasks(Liu et al., 2021), these methods often neglect multi-task interactions. Partial solutions, such as "Coding for Human Perception" and "Coding for Machine" (Li et al., 2024; Liu et al., 2023b), fall short in capturing complex multi-task relationships, underscoring the need for better multi-task compression.

3 Approach

3.1 Architecture

An overview of TAMC is provided in Fig. 3. It comprises three key components: (1) a *inter-coherent task cluster* that groups mutually coherent tasks into a shared representation space; (2) a *conditional entropy graph*, constructed using causal discovery to reveal dependencies; and (3) a *scalable compressor*, which compresses multiple feature layers along graph paths.

Components (1) and (2) draw inspiration from the lookahead stage in traditional video encoding (Li, 2003; He & Mitra, 2002; Wang & Kwong, 2008; Ma et al., 2005), which performs preliminary analysis to optimize the encoding performance of Component (3) under bitrate constraints. In Component (1), 2× downsampled low-resolution images are used as inputs, combined with a low-complexity feature extraction and decoding backbone (Standley et al., 2020), to efficiently pre-analyze task grouping

performance under the constraint of bitrate consumption. The task grouping strategy leverages gradient coherence to cluster tasks based on mutual information, maximizing shared information while minimizing redundancy. In Component (2), a conditional entropy graph organizes the grouped tasks hierarchically, facilitating information transfer from parent tasks to child tasks. This hierarchical structure improves representation certainty and encoding efficiency by providing a more precise cross-task entropy estimation model.

3.2 GROUP INTER-COHERENT TASKS

The objective of task grouping is to cluster tasks that can share representations, thereby optimizing the performance of multiple tasks under a given bitrate budget b. As task grouping is conducted in the pre-analysis module, the bitrate budget b is loosely approximated $\propto$ the number of task groups.[1]. Formally, a set of n tasks $\mathcal{T} = \{\tau_1, \ldots, \tau_n\}$, which is partitioned into k subgroups $\mathcal{C} = \{c_1, \ldots, c_k\}$, where each task $\tau_i \in \mathcal{T}$ belongs to at least one subgroup, *i.e.*, $\forall \tau \in \mathcal{T}, \exists c_g \in \mathcal{C} \mid \tau \in c_g$[2]. $k \leq n$ and each subgroup $c_g \subseteq \mathcal{T}$ shares a feature extractor (encoder) $g_a(\boldsymbol{x}; \boldsymbol{\theta}_e)$, as defined in Eq. 1. To distinguish between the lookahead and formal encoding stages, the set of shared feature extractors is denoted as $\mathcal{E} = \{l_{a_1}, \ldots, l_{a_k}\}$. For a raw image $\boldsymbol{x}$, the encoder l_{a_g} in subgroup c_g generates a shared representation $\boldsymbol{y}_g = l_{a_g}(\boldsymbol{x}; \boldsymbol{\theta}_{e_g})$. Each task τ_i is associated with a unique decoder $l_{s_i}(\boldsymbol{y}; \boldsymbol{\theta}_{d_i})$, corresponding to the synthesis transform $g_s(\hat{\boldsymbol{y}}; \boldsymbol{\theta}_d)$ in Eq. 1. The decoders $D = \{l_{s_1}, l_{s_2}, \ldots, l_{s_n}\}$ map the latent feature $\boldsymbol{y}_{g(i)}$ into task space, where $g(i)$ denotes the subgroup index for τ_i. The predicted output $\hat{\boldsymbol{x}}_i = l_{s_i}(\boldsymbol{y}_{g(i)}; \boldsymbol{\theta}_{d_i})$ is then compared with the ground truth for evaluation.

Task performance is measured using task-specific loss functions $L_i(\tau_i)$, *e.g.* PSNR/MS-SSIM for pixel reconstruction and cross-entropy loss for segmentation. For a given grouping strategy $\mathcal{C}$, the overall performance of the task grouping is aggregated as $\sum_{i=1}^{n} L_i(\tau_i|\mathcal{C})$. The bitrate $\mathcal{R}$ required to transmit the latent feature $\boldsymbol{y}_g$ for each group c_g is approximated by $b_g = B(\boldsymbol{y}_g)$, where B is $\propto$ the amount of data transmitted, with each shared representation consuming a unit cost of 1. Our goal is to minimize the total loss subject to the total bitrate constraint b:

$$\mathcal{L} = \sum_{i=1}^{n} \lambda_i \, L_i(\tau_i|\mathcal{C}) + \sum_{j=1}^{k} B(\boldsymbol{y}_j), \quad \text{s.t.} \sum_{j=1}^{k} B(\boldsymbol{y}_j) \leq b. \tag{5}$$

To achieve this, we evaluate the impact of task gradients on one another to determine the subgroup formation. Tasks within the same group share gradient updates, encoding parameters, and feature representations, enabling mutual learning. Gradient consistency is used to guide task grouping. Consider a multi-task model with shared parameters $\Theta_g = \{\boldsymbol{\theta}_{e_g}\}$ for group c_g, while Θ_u represents task specific parameters for τ_u . We determine whether to add τ_u to group c_g by evaluating the gradient consistency between Θ_u and Θ_g. Specifically, given a batch of inputs $\boldsymbol{x}$, we optimize the following total loss to measure the effect of task gradients:

$$\mathcal{L}_{total} = \sum_{i \in c_g \cup u} L_i(\tau_i \mid \boldsymbol{x}, \Theta_g, \Theta_u). \tag{6}$$

At time step t, for input batch $\boldsymbol{x}^t$, the gradient descent update at step $t+1$ is computed as:

$$\Theta_{g|u}^{t+1} \leftarrow \Theta_g - \alpha \nabla_{\Theta_g^t} L_u(\tau_u \mid \boldsymbol{x}^t, \Theta_g^t, \Theta_u^t), \tag{7}$$

where $\Theta_{g|u}^{t+1}$ denotes the updated parameters after considering τ_u. Using these updated shared parameters, we calculate the forecast loss for other tasks while keeping task-specific parameters and inputs unchanged. Gradient coherence between tasks τ_u and τ_v is then measured as:

$$C_{u \to v}^t = 1 - \frac{L_v(\tau_v \mid \boldsymbol{x}^t, \Theta_{g|u}^{t+1}, \Theta_v^t)}{L_v(\tau_v \mid \boldsymbol{x}^t, \Theta_g^t, \Theta_v^t)}. \tag{8}$$

A positive $C_{u \to v}^t$ indicates that the update from τ_u reduces the loss for τ_v, while a negative value $C_{u \to v}^t$ value indicates conflicting parameter update directions. Tasks with high gradient coherence

[1]Note: In the E2E compression training phase, bitrate consumption is defined as $\mathbb{E}_{\boldsymbol{x} \sim p_{\boldsymbol{x}}} \left[-\log_2 p_{\hat{\boldsymbol{y}}}(\hat{\boldsymbol{y}}) \right]$. During deployment, the cumulative distribution function (CDF) of $P_{\hat{\boldsymbol{y}}}(\hat{\boldsymbol{y}})$ is used for arithmetic coding to determine the final bitrate.

[2]As shown in Fig. 3, tasks may belong to multiple groups. Transparent tasks are used during training to boost group performance but are discarded during inference.

are grouped to share encoders and representations. After T timesteps, the cumulative coherent measure is calculated as: $\hat{C}_{u\to v} = \frac{1}{T}\sum_{t=1}^{T} C^t_{u\to v}$ After obtaining the gradient coherence measures between all pairs of tasks, the number of possible grouping for n tasks is given by the Bell number: $\text{Bell}_{n+1} = \sum_{k=0}^{n} \binom{n}{k}\text{Bell}_k, \text{Bell}_0 = 1$, which grows rapidly with n. To address this computational complexity, we apply relaxed estimates for higher-order cluster coherent measures. Specifically, for a triplet of tasks $\{\tau_u, \tau_v, \tau_w\}$, the triplet cost is estimated as the average of pairwise coherence measures $\hat{C}^t_{u\to v}$ and $\hat{C}^t_{w\to v}$. Assuming each cluster has a unit bitrate cost, we adopt a branch-and-bound method, as in prior works (Standley et al., 2020; Zamir et al., 2018), to search for locally optimal grouping strategies under the given bitrate budget b.

3.3 Causal Graph-Based Compression Method

Constructing DAG via Conditional Entropy. Let $\mathcal{Y} = \{Y_1, Y_2, \ldots, Y_k\}$ be a set of task-shared random variables, where each Y_i corresponds to the shared representation $\boldsymbol{y}_i$ for task group c_i. We say that Y_i *causes* Y_j if there exists a function f and an exogenous random variable N, independent of Y_i, such that $Y_j = f(Y_i, N)$. This causal relationship is represented by the edge $Y_i \to Y_j$ in a directed acyclic graph (DAG) $G = (V, E)$, where V denotes the vertex (*i.e.* $\mathcal{Y}$ in our context) and the set of E denotes the set of edges. A structural causal model is defined as $\mathcal{M} = (\mathcal{Y}, \mathcal{N}, \mathcal{F}, \mathcal{P}_{\mathcal{N}})$, where $\mathcal{F} = \{f_1, f_2, \ldots, f_k\}$ are functions, $\mathcal{N}$ is the set of exogenous variables, and $\mathcal{P}_{\mathcal{N}}$ is a probability distribution over $\mathcal{N}$. The joint distribution over $\mathcal{Y}$ is induced by $\mathcal{P}_{\mathcal{N}}$ and $\mathcal{F}$, denoted as $\mathcal{P}_{\mathcal{Y}}$. In G, a node Y_i is a parent of node Y_j if $Y_j = f_j(Y_i, S, N_j)$ for some $S \subset \mathcal{Y}$.

Identifying the true causal graph without experiments or strong assumptions is generally infeasible. To efficiently compress the representations $\mathcal{Y}$, we approximate causal relationships by minimizing the conditional entropy, which quantifies the uncertainty of one variable given another. Specifically, given Y_i and Y_j with domains $\mathcal{Y}_i$ and $\mathcal{Y}_j$ respectively, the pair-wise conditional entropy $H(Y_i \mid Y_j)$ (also denoted as the conditional entropy oracle $\mathscr{H}$ in **Algorithm 1**) is defined as:

$$H(Y_i \mid Y_j) = -\sum_{\boldsymbol{y}_i \in \mathcal{Y}_i, \boldsymbol{y}_j \in \mathcal{Y}_j} p(\boldsymbol{y}_i, \boldsymbol{y}_j) \log p(\boldsymbol{y}_i \mid \boldsymbol{y}_j). \tag{9}$$

We define the parent Y_p of the children variable Y_c as the one that minimizes the conditional entropy:

$$Y_p = \arg \min_{Y_j \in \mathcal{Y} \setminus \{Y_c\}} H(Y_c \mid Y_j). \tag{10}$$

Algorithm 1 : DAG Construction via Conditional Entropy

Require: Latents set $\mathcal{Y} = \{Y_1, \ldots, Y_k\}$, conditional entropy oracle $\mathscr{H}$
Ensure: Causal DAG $G = (V, E)$
Initialize $V = \mathcal{Y}$ and $E = \emptyset$
for each latent $Y_c \in \mathcal{Y}$ **do**
  Initialize min_cond_entropy $\leftarrow \infty$, $Y_p \leftarrow$ None
  for each latent $Y_j \in \mathcal{Y} \setminus \{Y_c\}$ **do**
    Compute $H(Y_c \mid Y_j)$
    if $H(Y_c \mid Y_j) <$ min_cond_entropy and $H(Y_j < Y_c)$ **then**
      Update min_cond_entropy $\leftarrow H(Y_c \mid Y_j)$
      Update $Y_p \leftarrow Y_j$
    end if
  end for
  Add edge $Y_p \to Y_c$ to E
end for
return $G = (V, E)$

By applying this criterion iteratively to all representations in $\mathcal{Y}$, we construct the edges E of the causal graph G. The causal discovery process is detailed in **Algorithm** 1.

Scalable Compression Using the Causal DAG. After constructing G, inspired by the causal context entropy model Guo et al. (2021) and MLIC++ (Jiang et al., 2023), we perform compression by traversing the graph in topological order. The parent representation $\boldsymbol{y}_p$ serves as an additional cross-task context for the child representation $\boldsymbol{y}_c$, enhancing compression efficiency. Specifically, when the parent latent representation $\boldsymbol{y}_p$ is assumed to *cause* the child representation $\boldsymbol{y}_c$, the distribution of $\boldsymbol{y}_p$ is modelled using the prior framework in Eq. 3. As illustrated in Fig. 2 (right part), the prediction of $p_{\hat{\boldsymbol{y}}_p}(\hat{\boldsymbol{y}}_p)$ involves a mask convolution $\boldsymbol{\phi}_{p,i} = g_{cm}(\hat{\boldsymbol{y}}_p < i; \boldsymbol{\theta}_{cm})$, which generates the local context $\boldsymbol{\phi}_{p,i}$. This context is then combined with the hyperpriors $\boldsymbol{z}_p$ to estimate the Gaussian distribution parameters for $\hat{\boldsymbol{y}}_p$. When it comes to decoding $\hat{\boldsymbol{y}}_c$, we aggregate both the decoded latent $\hat{\boldsymbol{y}}_p$ and the first half latent in the current spatial location $\hat{\boldsymbol{y}}_c$, generating more informative contexts $\boldsymbol{\phi}_{c,i}$. As shown in Fig. 4a, the whole process can be extended from Eq.3 and formulated as:

$$p_{\hat{\boldsymbol{y}}_c}(\hat{\boldsymbol{y}}_c \mid \hat{\boldsymbol{z}}_c, \boldsymbol{\theta}_{hd}, \boldsymbol{\theta}_{cm+}, \boldsymbol{\theta}_{ep}) = \prod_i \left(\mathcal{N}(\mu_{c,i}, \sigma^2_{c,i}) * \mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right) \right)(\hat{\boldsymbol{y}}_{c,i}), \tag{11}$$

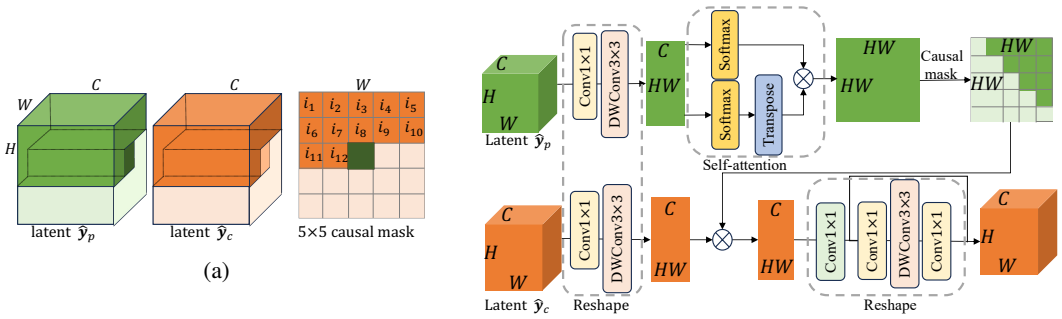


Figure 4: The structure of proposed cross-task causal context compression: (a) Cross-task local causal context mask convolution. (b) The parent $\hat{\boldsymbol{y}}_p$ provides a global context for the child $\hat{\boldsymbol{y}}_c$.

with $\mu_{c,i}, \sigma_{c,i} = g_{ep}(\boldsymbol{\psi}_c, \boldsymbol{\phi}_{c,i}; \boldsymbol{\theta}_{ep}), \boldsymbol{\psi}_c = g_h(\hat{\boldsymbol{z}}_c; \boldsymbol{\theta}_{hd})$, and $\boldsymbol{\phi}_{c,i} = g_{cm+}(\hat{\boldsymbol{y}}_c < i, \hat{\boldsymbol{y}}_p; \boldsymbol{\theta}_{cm+})$. The differences from Eq. 3 are highlighted in blue.

The above causal context model only extracts local correlations but ignores global correlations. *Causal global prediction model* is proposed to utilize the long-range correlations of the parent $\hat{\boldsymbol{y}}_p$ and child $\hat{\boldsymbol{y}}_c$. The overall process shown in Fig.4b and also formulated as:

$$\hat{\boldsymbol{y}}^{\text{attn}}_{c,i} = \underbrace{\text{softmax}_2\left(\hat{\boldsymbol{y}}^q_p < i\right)\text{softmax}_1\left(\hat{\boldsymbol{y}}^k_p < i\right)^\top}_{\text{non-negative}} \hat{\boldsymbol{y}}^v_c < i, \hat{\boldsymbol{y}}^{\text{conv}}_{c,i} = \text{conv}_{K\times K}(\hat{\boldsymbol{y}}^{\text{attn}}_{c,i}), \boldsymbol{\phi}_{gc,i} = \text{DepthRB}(\hat{\boldsymbol{y}}^{\text{conv}}_{c,i}), \tag{12}$$

where $\hat{\boldsymbol{y}}^q_p < i, \hat{\boldsymbol{y}}^k_p < i = \text{Embedding}(\hat{\boldsymbol{y}}_p < i), \hat{\boldsymbol{y}}^v_c < i = \text{Embedding}(\hat{\boldsymbol{y}}_c < i)$, and the embedding layer consists of a 1×1 convolutional layer and a 3×3 depth-wise convolutional layer. The 3×3 depth-wise convolutional layer is employed for learnable position embedding. Use $\boldsymbol{\theta}_{gc}$ denotes the trainable parameters in the Causal global prediction model, and then the Eq.3 can be extended as:

$$p_{\hat{\boldsymbol{y}}_c}(\hat{\boldsymbol{y}}_c \mid \hat{\boldsymbol{z}}_c, \boldsymbol{\theta}_{hd}, \boldsymbol{\theta}_{cm}, \boldsymbol{\theta}_{gc}, \boldsymbol{\theta}_{ep}) = \prod_i \left(\mathcal{N}(\mu_{c,i}, \sigma^2_{c,i}) * \mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right)\right)(\hat{\boldsymbol{y}}_{c,i}), \tag{13}$$

with $\mu_{c,i}, \sigma_{c,i} = g_{ep}(\boldsymbol{\psi}_c, \boldsymbol{\phi}_{c,i}; \boldsymbol{\phi}_{gc,i}; \boldsymbol{\theta}_{ep}), \boldsymbol{\psi}_c = g_h(\hat{\boldsymbol{z}}_c; \boldsymbol{\theta}_{hd})$, and $\boldsymbol{\phi}_{c,i} = g_{cm}(\hat{\boldsymbol{y}}_c < i, \hat{\boldsymbol{y}}_p; \boldsymbol{\theta}_{cm}), \boldsymbol{\phi}_{gc,i} = g_{gc}(\hat{\boldsymbol{y}}_c < i, \hat{\boldsymbol{y}}_p; \boldsymbol{\theta}_{gc})$. The differences from Eq. 11 are highlighted in green.

4 Experiments

4.1 Tasks and Datasets

To quantify the performance across diverse downstream tasks, we evaluate 5 compression benchmarks on the Taskonomy dataset (Zamir et al., 2018) across 6 tasks. Taskonomy is a large-scale computer vision dataset that includes over 4.5 million images from more than 500 buildings. Each image has 18 annotations covering 2D, 3D, and semantic tasks. The total size of the dataset is 11.16 TB. Due to limited computational and storage resources, we used the Tiny split for our experiments, which consists of 872,517 images in the training set and 16,000 images in the validation set and test set, respectively. We conducted experiments on 6 tasks selected from the 15 annotated tasks, *i.e.*, Semantic Segment, Keypoint 2D, Edge Texture, Surface Normal, Depth Zbuffer and Autoencoder. More details of tasks and loss measurements are placed in A.2.

4.2 Baselines

Our method is compared against several baselines, including JPEG (Pennebaker & Mitchell, 1992), WebP (Si & Shen, 2016), VTM-17.0 (Bross et al., 2021), as well as learning-based compression methods, *i.e.*, ELIC (He et al., 2022) and MLIC++ (Jiang et al., 2023).

To evaluate the performance of different compression methods across a variety of tasks, we used the official open-source code of compression methods to compress input images. Subsequently, we

assessed the performance of the reconstructed images on various tasks by analyzing the features extracted by the pre-trained downstream task models provided by the Taskonomy dataset.

For our method, we first employed Xception (Chollet, 2017) as the encoder backbone, and the task-specific decoder consists of four transposed convolutional layers and four convolutional layers. Initially, we trained the group-task shared encoder-decoder without compression loss for 60 epochs using the SGD optimizer (Ruder, 2016), with the learning rate decaying from 0.1 to 1e-4. Then, based on the conditional entropy of the shared representation, we constructed a directed acyclic graph (DAG) among the child nodes of the shared representation. Finally, we learned entropy models for different compression rates by following paths from parent nodes to child nodes. We set the task learning rate to 1e-4, and the learning rate of the hyperprior entropy model to 1e-4. We continued training for 50 epochs, adjusting the λ parameter in the distortion-rate trade-off $\lambda \times \mathcal{D} + \mathcal{R}$ with values from [0.04, 0.072, 0.14, 1, 1.932] to learn encoder-decoder models parameters and entropy model parameters for different bitrates.

4.3 Results

4.3.1 Performance of Compression for multiple tasks

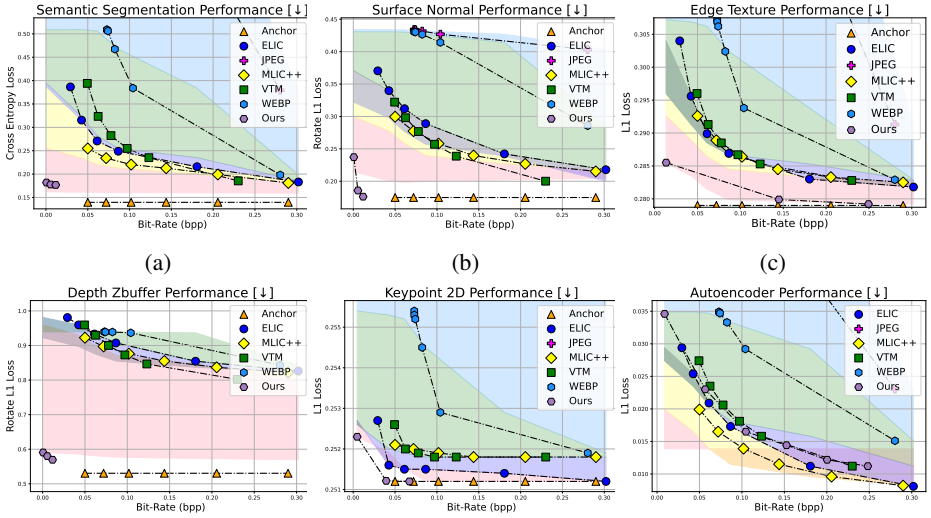


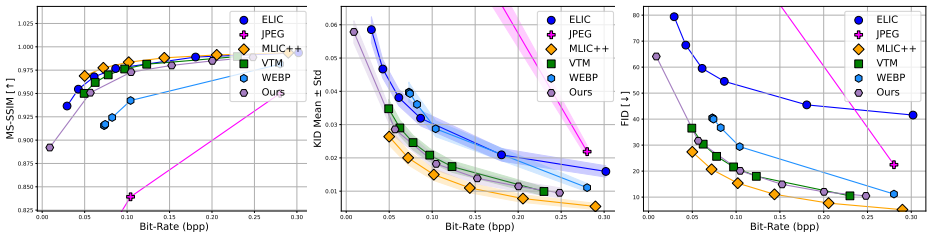
Figure 5: Comparison of performance-rate curves for 6 tasksusing baseline compression methods and our proposed TAMC. "Anchor" refers to the optimal performance of a supervision task obtained using uncompressed images as input. The area of the shade color visualizes bitrate-performance gains.

Our results, summarized in Fig. 5, demonstrate that TAMC achieves a new state-of-the-art in compression for multitask scenarios, significantly outperforming other baselines across 5 of 6 tasks. In multitask learning and compression, different tasks have distinct requirements for image information. Traditional compression methods (*i.e.*, JPEG, WebP, VTM) and end-to-end deep learning approaches (*i.e.*, ELIC, MLIC++) typically use a unified compression representation for all tasks, overlooking task relationships and dependencies. This often leads to conflicts, causing suboptimal performance for certain tasks. TAMC addresses this by first partitioning tasks into complementary groups, where tasks within the same group share representations. We then use causal discovery through conditional entropy to identify dependencies among groups. These shared representations are progressively compressed based on parent-child relationships, effectively leveraging the context priors from parent nodes to reduce the uncertainty of child nodes.

TAMC demonstrates particularly significant improvements in *depth zbuffer*, *semantic segmentation*, *surface normal*, and *edge texture*. For depth zbuffer, TAMC preserves more geometric details, especially the spatial structural information essential for depth buffer, through task grouping and shared representations. Our experiments validate this, which shows a substantial decrease in L1

loss, outperforming traditional methods and even existing deep-learning compression techniques. Semantic segmentation, normal surface prediction, and edge texture detection rely on local image details and geometric structures. TAMC effectively compresses while retaining features useful for these tasks through causal discovery and grouping mechanisms. Notably, TAMC outperforms traditional compression methods in low-bit-rate scenarios.

4.3.2 Performance of Image Compression



(a) (b) (c)

Figure 6: Comparison of performance-rate curves for image compression on the Taskonomy dataset using baseline methods and our proposed TAMC.

Method	Bitrate	Task Grouping1			Task Grouping2		Task Grouping3		
		Segment.	Depth	Normal	Normal	Keypoint	Segment.	Normal	Texture
JPEG	0.283	44.32%	29.86%	-7.05%	-14.10%	-44.58%	28.40%	-17.56%	-0.06%
WebP	0.281	3.13%	20.08%	-3.69%	2.79%	-49.00%	-0.91%	-5.43%	-0.59%
VTM-17.0	0.229	-0.24%	-5.28%	8.24%	11.16%	-46.65%	-1.82%	2.84%	-0.37%
ELIC	0.302	-1.50%	-0.03%	6.48%	9.69%	-39.11%	-2.47%	2.27%	-0.90%
MLIC++	0.289	-1.18%	-1.05%	6.65%	9.79%	-40.40%	-2.26%	2.47%	-0.93%
TAMC	0.224	-10.08%	3.33%	0.21%	12.43%	-45.02%	-4.84%	5.38%	-1.05%

Table 1: Performance loss reduction of grouped inter-coherent tasks relative to single-task training.

As shown in Figure 6, our proposed algorithm demonstrates competitive performance compared to the VTM baseline. For structure- and texture-sensitive metrics, such as MS-SSIM (Wang et al., 2003), our method (TAMC) is slightly inferior to VTM. For semantic-sensitive metrics, including KID (Bińkowski et al., 2018), FID (Heusel et al., 2017) in Fig. 6, and LPIPS (Zhang et al., 2018) in Fig. 14, TAMC is slightly superior to VTM, reflecting its strong capacity for semantic understanding and perceptual quality retention. However, the Peak Signal-to-Noise Ratio (PSNR) results in Fig. 14a are comparatively lower, indicating room for improvement in pixel-level fidelity. This could be attributed to our choice of Xception (Chollet, 2017) as the encoder backbone and a task-specific decoder consisting of four transposed convolutional layers and four convolutional layers, which aligns with the architectures pre-trained for machine tasks (Fifty et al., 2021) but does not yet integrate the full advantages of advanced backbone modules for image compression, *e.g.*, GDN (Ballé et al., 2018), residual networks (Cheng et al., 2020), and transformers (Zou et al., 2022; Zhu et al., 2021; Lu et al., 2022), suggesting potential for further exploration.

5 Analysis

5.1 Ablation of Group Inter-Coherent Tasks

In our experiments, given 5 downstream tasks: {Semantic Segmentation, Depth Zbuffer, Edge Texture, Surface Normal, Keypoints 2D}, our task grouping results based on gradient coherence between task pairs are as follows: **Group 1** {Semantic Segmentation, Depth Zbuffer, Surface Normal}, **Group 2** {Surface Normal, Keypoints 2D} and **Group 3** {Semantic Segmentation, Surface Normal, Edge Texture. As illustrated in Fig. 1, during inference, only the tasks marked in green within each group are decoded and referenced for downstream tasks, while the remaining tasks are solely used to enhance performance during training and discarded during inference.

For traditional and end-to-end compression methods, we assess the relative performance of task grouping in downstream tasks compared to performance without task grouping. For our progressive compression paradigm, we evaluate the relative performance of latent space disentanglement with task grouping versus without task grouping. In the ablation study of grouping inter-coherent tasks, the causal discovery module is not employed. Our goal is to evaluate the effectiveness and generalization of task grouping in the context of multi-task compression. From the experimental results, it is evident that for baseline methods, the performance of tasks such as Semantic Segmentation in Group 1, Keypoints 2D in Group 2, and Edge Texture in Group 3 experienced a significant degradation when compared to single-task training due to the influence of other tasks. Although the tasks of Depth Zuffer and Normal Surface in Group 1 did not exhibit any noticeable improvements and even showed slight performance declines, they contributed positively to the performance enhancements of other tasks within their respective groups. Additionally, the shared representation effectively reduced the bitrate budget, indicating its efficiency. For TAMC, where the task groupings in the latent space were applied in the reconstructed pixel space for downstream task inference, it can be observed that, apart from JPEG and WebP, other algorithms demonstrated trends similar to ours. This could be attributed to the fact that the low-dimensional pixel space in JPEG and WebP suffers from compression artifacts and natural pixel space biases, which introduce deviations from the predicted classification results.

5.2 Ablation of Causal Discovery Topology

To better understand how causal discovery graphs constructed via conditional entropy optimize both collaborative bitrate and task performance, we analyzed their effectiveness in determining the optimal trade-off between these factors. As visualized in Fig. 7, different graph construction methods are compared across various settings. In Setting A, where the graph follows principles of causal discovery via conditional entropy, the resulting tasks exhibit better coordination, balancing both bitrate efficiency and task performance. In contrast, Setting B and Setting C represent random graph constructions that do not adhere to causal discovery principles. The visualized results show that Setting B requires a higher bitrate to achieve similar performance while Setting C not only consumes more bitrate but also results in performance degradation. This comparison highlights the importance of proper graph construction in multi-task compression systems.

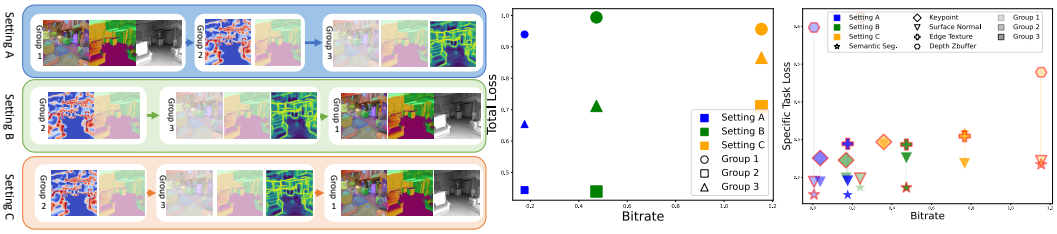


Figure 7: Impact of different DAG topologies on bitrate and multi-task performance (lower left indicates better solutions, where both bitrate and performance loss are minimized). **Left**: Example of graph construction methods. **Middle**: Task performance of graph node and bitrate consumption of the entire causal graph. Note: Different colors denote different graph topologies, and different shapes represent different nodes. **Right**: A more detailed breakdown of the middle, showing bitrate consumption and performance for different nodes in each causal graph. Note: Different colors represent different topologies, with varying transparency of the same color indicating different nodes within the same graph. Different shapes represent different tasks. Red-bordered shapes highlight the best-performing task in each topology (prioritizing task performance over bitrate).

6 Conclusion

In this work, we introduced a novel multi-task representation compression framework that leverages causal discovery via conditional entropy to optimize the trade-off between bitrate efficiency and task performance. By grouping mutually beneficial tasks and constructing a DAG to characterise their interdependencies, our method enables efficient compression of disentangled representations. Through extensive experiments on key computer vision tasks, we demonstrated the effectiveness of our approach in both bitrate reduction and task accuracy. Our findings highlight the importance of properly structured task groupings and causal relationships in multi-task compression, offering a promising direction for future work in video coding for machine learning and multi-task optimization.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Eirikur Agustsson, Michael Tschannen, Fabian Mentzer, Radu Timofte, and Luc Van Gool. Generative adversarial networks for extreme learned image compression. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 221–231, 2019.

Yuanchao Bai, Xu Yang, Xianming Liu, Junjun Jiang, Yaowei Wang, Xiangyang Ji, and Wen Gao. Towards end-to-end image compression and analysis with transformers. In *AAAI*, pp. 104–112, 2022.

Johannes Ballé, Valero Laparra, and Eero P Simoncelli. End-to-end optimized image compression. In *ICLR*, 2017.

Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *ICLR*, 2018.

Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018.

Benjamin Bross, Ye-Kui Wang, Yan Ye, Shan Liu, Jianle Chen, Gary J Sullivan, and Jens-Rainer Ohm. Overview of the versatile video coding (vvc) standard and its applications. *IEEE TCSVT*, 31 (10):3736–3764, 2021.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45, 2024.

Tong Chen, Haojie Liu, Zhan Ma, Qiu Shen, Xun Cao, and Yao Wang. End-to-End learnt image compression via non-local attention optimization and improved context modeling. *IEEE TIP*, 30: 3179–3191, 2021.

Zhao Chen, Jiquan Ngiam, Yanping Huang, Thang Luong, Henrik Kretzschmar, Yuning Chai, and Dragomir Anguelov. Just pick a sign: Optimizing deep multitask models with gradient sign dropout. *Advances in Neural Information Processing Systems*, 33:2039–2050, 2020.

Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Learned image compression with discretized gaussian mixture likelihoods and attention modules. In *CVPR*, pp. 7939–7948, 2020.

Hyomin Choi and Ivan V Bajić. Scalable image coding for humans and machines. *IEEE Transactions on Image Processing*, 31:2739–2754, 2022.

François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

Lingyu Duan, Jiaying Liu, Wenhan Yang, Tiejun Huang, and Wen Gao. Video coding for machines: A paradigm of collaborative compression and intelligent analytics. *IEEE TIP*, 29:8680–8695, 2020.

Zhihao Duan, Ming Lu, Zhan Ma, and Fengqing Zhu. Lossy image compression with quantized hierarchical vaes. In *WACV*, pp. 198–207, 2023.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12873–12883, 2021.

Chris Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. Efficiently identifying task groupings for multi-task learning. *Advances in Neural Information Processing Systems*, 34: 27503–27516, 2021.

Xingtong Ge, Jixiang Luo, Xinjie Zhang, Tongda Xu, Guo Lu, Dailan He, Jing Geng, Yan Wang, Jun Zhang, and Hongwei Qin. Task-aware encoder control for deep video compression. In *CVPR*, pp. 26036–26045, 2024.

Zongyu Guo, Zhizheng Zhang, Runsen Feng, and Zhibo Chen. Causal contextual prediction for learned image compression. *IEEE TCSVT*, 32(4):2329–2341, 2021.

Dailan He, Yaoyan Zheng, Baocheng Sun, Yan Wang, and Hongwei Qin. Checkerboard context model for efficient learned image compression. In *CVPR*, pp. 14771–14780, 2021.

Dailan He, Ziming Yang, Weikun Peng, Rui Ma, Hongwei Qin, and Yan Wang. ELIC: Efficient learned image compression with unevenly grouped space-channel contextual adaptive coding. In *CVPR*, pp. 5718–5727, 2022.

Zhihai He and Sanjit K Mitra. Optimum bit allocation and accurate rate control for video coding via/spl rho/-domain source modeling. *IEEE transactions on Circuits and Systems for Video Technology*, 12(10):840–849, 2002.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Forrest N Iandola. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

Wei Jiang, Jiayu Yang, Yongqi Zhai, Peirong Ning, Feng Gao, and Ronggang Wang. MLIC: Multi-reference entropy model for learned image compression. In *ACM MM*, pp. 7618–7627, 2023.

Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7482–7491, 2018.

Yeongwoong Kim, Hyewon Jeong, Janghyun Yu, Younhee Kim, Jooyoung Lee, Se Yoon Jeong, and Hui Yong Kim. End-to-end learnable multi-scale feature compression for vcm. *IEEE TCSVT*, 2023.

Md Tahmid Rahman Laskar, M Saiful Bari, Mizanur Rahman, Md Amran Hossen Bhuiyan, Shafiq Joty, and Jimmy Xiangji Huang. A systematic study and comprehensive evaluation of chatgpt on benchmark datasets. *arXiv preprint arXiv:2305.18486*, 2023.

Han Li, Shaohui Li, Shuangrui Ding, Wenrui Dai, Maida Cao, Chenglin Li, Junni Zou, and Hongkai Xiong. Image compression for machine and human vision with spatial-frequency adaptation. *arXiv preprint arXiv:2407.09853*, 2024.

Zhengguo Li. Adaptive basic unit layer rate control for jvt. In *JVT 7th Meeting, Pattaya, Mar2003*, 2003.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, pp. 740–755. Springer, 2014.

Jinming Liu, Heming Sun, and Jiro Katto. Learned image compression with mixed transformer-cnn architectures. In *CVPR*, pp. 14388–14397, 2023a.

Kang Liu, Dong Liu, Li Li, Ning Yan, and Houqiang Li. Semantics-to-signal scalable image compression with learned revertible representations. *IJCV*, 129(9):2605–2621, 2021.

Lei Liu, Zhihao Hu, Zhenghao Chen, and Dong Xu. Icmh-net: Neural image compression towards both machine vision and human vision. In *ACM MM*, pp. 8047–8056, 2023b.

Yixin Liu, Kai Zhang, Yuan Li, Zhiling Yan, Chujie Gao, Ruoxi Chen, Zhengqing Yuan, Yue Huang, Hanchi Sun, Jianfeng Gao, et al. Sora: A review on background, technology, limitations, and opportunities of large vision models. *arXiv preprint arXiv:2402.17177*, 2024a.

Yuxi Liu, Wenhan Yang, Huihui Bai, Yunchao Wei, and Yao Zhao. Region-adaptive transform with segmentation prior for image compression. *arXiv preprint arXiv:2403.00628*, 2024b.

Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 12-in-1: Multi-task vision and language representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10437–10446, 2020.

Ming Lu, Peiyao Guo, Huiqing Shi, Chuntong Cao, and Zhan Ma. Transformer-based image compression. In *DCC*, pp. 469–469. IEEE, 2022.

Haichuan Ma, Dong Liu, Ning Yan, Houqiang Li, and Feng Wu. End-to-end optimized versatile image compression with wavelet-like transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Siwei Ma, Wen Gao, and Yan Lu. Rate-distortion analysis for h. 264/avc video coding and its application to rate control. *IEEE transactions on circuits and systems for video technology*, 15(12): 1533–1544, 2005.

Qi Mao, Tinghan Yang, Yinuo Zhang, Zijian Wang, Meng Wang, Shiqi Wang, Libiao Jin, and Siwei Ma. Extreme image compression using fine-tuned vqgans. In *2024 Data Compression Conference (DCC)*, pp. 203–212. IEEE, 2024.

Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool. Conditional probability models for deep image compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4394–4402, 2018.

Fabian Mentzer, George Toderici, Michael Tschannen, and Eirikur Agustsson. High-fidelity generative image compression. In *Advances In Neural Information Processing Systems*, 2020.

David Minnen and Saurabh Singh. Channel-wise autoregressive entropy models for learned image compression. In *ICIP*, pp. 3339–3343. IEEE, 2020.

David Minnen, Johannes Ballé, and George D Toderici. Joint autoregressive and hierarchical priors for learned image compression. *NeurIPS*, 31, 2018.

Ken M Nakanishi, Shin-ichi Maeda, Takeru Miyato, and Daisuke Okanohara. Neural multi-scale image compression. In *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part VI 14*, pp. 718–732. Springer, 2019.

William B Pennebaker and Joan L Mitchell. *JPEG: Still image data compression standard*. Springer Science & Business Media, 1992.

Yichen Qian, Zhiyu Tan, Xiuyu Sun, Ming Lin, Dongyang Li, Zhenhong Sun, Hao Li, and Rong Jin. Learning accurate entropy model with global reference for image compression. *arXiv preprint arXiv:2010.08321*, 2020.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Oren Rippel and Lubomir Bourdev. Real-time adaptive image compression. In *International Conference on Machine Learning*, pp. 2922–2930. PMLR, 2017.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

Shibani Santurkar, David Budden, and Nir Shavit. Generative compression. In *Proc. of Picture Coding Symposium*, 2018.

Claude E Shannon et al. Coding theorems for a discrete source with a fidelity criterion. *IRE Nat. Conv. Rec*, 4(142-163):1, 1959.

Shi, Qimai Li, Wenlong Zhang, Jiaxin Chen, and Xiao-Ming Wu. Recon: Reducing conflicting gradients from the root for multi-task learning. In *The International Conference on Learning Representations*, 2023.

Zhanjun Si and Ke Shen. Research on the webp image format. In *Advanced graphic communications, packaging technology and materials*, pp. 271–277. Springer, 2016.

Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Which tasks should be learned together in multi-task learning? In *International conference on machine learning*, pp. 9120–9132. PMLR, 2020.

Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. *arXiv preprint arXiv:1703.00395*, 2017.

Michael Tschannen, Eirikur Agustsson, and Mario Lucic. Deep generative models for distribution-preserving lossy compression. In *Advances In Neural Information Processing Systems*, 2018.

Hanli Wang and Sam Kwong. Rate-distortion optimization of rate control for h. 264 with adaptive initial quantization parameter determination. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(1):140–144, 2008.

Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pp. 1398–1402. Ieee, 2003.

Sen Wu, Hongyang R Zhang, and Christopher Ré. Understanding and improving information transfer in multi-task learning. *arXiv preprint arXiv:2005.00944*, 2020.

Jinxi Xiang, Kuan Tian, and Jun Zhang. MIMT: Masked image modeling transformer for video compression. In *ICLR*, 2022.

Yueqi Xie, Ka Leong Cheng, and Qifeng Chen. Enhanced invertible encoding for learned image compression. In *ACM MM*, pp. 162–170, 2021.

Youmin Xu and Jian Zhang. Invertible resampling-based layered image compression. In *2021 Data Compression Conference (DCC)*, pp. 380–380. IEEE, 2021.

Wenhan Yang, Haofeng Huang, Yueyu Hu, Ling-Yu Duan, and Jiaying Liu. Video coding for machines: compact visual representation compression for intelligent collaborative analytics. *IEEE TPAMI*, 2024.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33: 5824–5836, 2020.

Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *CVPR*, pp. 3712–3722, 2018.

Ce Zhang, Taixi Lu, Md Mohaiminul Islam, Ziyang Wang, Shoubin Yu, Mohit Bansal, and Gedas Bertasius. A simple llm framework for long-range video question-answering. *arXiv preprint arXiv:2312.17235*, 2023.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Yu Zhang and Qiang Yang. A survey on multi-task learning. *IEEE transactions on knowledge and data engineering*, 34(12):5586–5609, 2021.

Zibin Zheng, Kaiwen Ning, Yanlin Wang, Jingwen Zhang, Dewu Zheng, Mingxi Ye, and Jiachi Chen. A survey of large language models for code: Evolution, benchmarking, and future trends. *arXiv preprint arXiv:2311.10372*, 2023.

Xiaosu Zhu, Jingkuan Song, Lianli Gao, Feng Zheng, and Heng Tao Shen. Unified multivariate gaussian mixture for efficient neural image compression. In *CVPR*, pp. 17612–17621, 2022.

Yinhao Zhu, Yang Yang, and Taco Cohen. Transformer-based transform coding. In *ICLR*, 2021.

Renjie Zou, Chunfeng Song, and Zhaoxiang Zhang. The devil is in the details: Window-based attention for image compression. In *CVPR*, pp. 17492–17501, 2022.

A APPENDIX

A.1 DISCUSSION

While graph models can effectively capture complex relationships between multiple representation spaces, it is well-known that the computational complexity of adding or removing nodes in graph structures is high. This becomes especially challenging in open-set scenarios, where tasks or data points are continuously evolving. Developing efficient methods for dynamically adding and removing nodes while maintaining the integrity of the graph remains an important research question.

Another critical point is the representation compatibility between different tasks. Ideally, the representation form and model architecture should be customized to fit the specific requirements of each task and application scenario. Until a truly unified model emerges, representations and architectures might not always be fully compatible across tasks. In our current work, we used a shared model architecture and representations across tasks, which, while ideal for controlled experimentation, may not reflect the diversity seen in real-world applications. However, our experiments successfully validated the feasibility and effectiveness of using a causal graph model based on conditional entropy for multi-task compression under these controlled conditions.

This work offers a foundational approach to multi-task compression using causal discovery, but the model's flexibility and scalability in more diverse and dynamic environments warrant further investigation.

A.2 MORE DETAILS OF EXPERIMENTAL SETTINGS

We applied and measured 6 tasks in Taskonomy[3], which is listed below:

- **Semantic Segment:** The annotations include 18 unique labels, with 16 object classes, a "background" class, and an "uncertain" class. For this task, we evaluate compression performance at different compression rates using cross-entropy loss.
- **Keypoint 2D:** This task involves 2D keypoint heatmaps. We assess compression performance at different compression rates using the L1 loss.
- **Edge Texture:** This task involves detecting 2D edge textures. Similar to Keypoints2D, we evaluate performance using L1 loss at different compression rates.
- **Surface Normal:** This task includes surface normal images, centered at 127. To evaluate performance under different compression rates, we use the `rotate_loss`, which is commonly applied in image processing or volume rendering tasks. The loss computes the L1 difference between the output and target and compares the result across 9 different orientations to find the minimal loss. This ensures that the model's depth predictions remain consistent under rotational and translational transformations, which is crucial when dealing with real-world noise and variations.
- **Depth Zbuffer:** This task involves Z-buffer depth images, measured in units of 1/512m with a maximum range of 128m. Similar to the Normal task, we use `rotate_loss` at different compression rates, first calculating the L1 difference between the output and target, then comparing across 9 orientations to find the minimal loss.
- **Autoencoder:** This task reconstructs RGB images at a resolution of 512×512. We evaluate compression performance at different compression rates using the L1 loss. Additionally, we assess fidelity using PSNR/SSIM (Wang et al., 2003) and perceptual quality using LPIPS (Zhang et al., 2018) / KID (Bińkowski et al., 2018) / FID (Heusel et al., 2017).

[3] `https://github.com/StanfordVL/taskonomy/tree/master/data` by (Zamir et al., 2018)

For the compression baselines, we used the open source codes, *i.e.*, JPEG[4], WebP [5], VTM-17.0[6], ELIC [7], and MLIC++ [8]. For downstream tasks, we uniformly used the Taskonomy pre-trained [9] Xception encoder and task-specific decoder. TAMC directly performs coherent task grouping and causal graph construction in the latent space, so its encoder and decoder follow the Xception structure of the pre-trained model. For the image compression task, better model architecture designs are already available to optimize performance.

A.3 Supplementary Ablation Studies

In this section, we extend our analysis to E2E compression involving multiple supervised tasks as auxiliary tasks. We also examine single-task groups, where each task is treated as an independent group rather than grouping tasks together. Additionally, we explore whether the conherence from Task A to Task B can be used to predict the conherence from Task B to Task A. Furthermore, we investigate the impact of prohibiting the same task from appearing in multiple clusters and assess whether this restriction leads to better or worse performance. Finally, we compare our approach with VQ-GAN-based compression method to evaluate its overall effectiveness.

A.3.1 Additional Results of E2E Compression with Multiple Task Auxiliary Loss

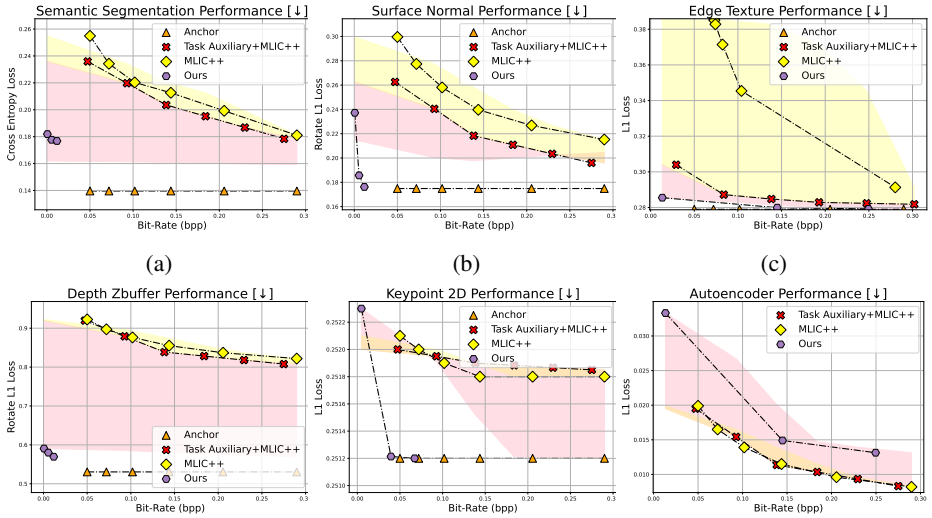


Figure 9: Performance-rate curves for 6 tasks: Semantic Segmentation, Depth Z-buffer, Surface Normal, Keypoint 2D, Edge Texture, and Autoencoder, on the Taskonomy dataset. The comparison includes baseline compression methods (MLIC++, Task Auxiliary+MLIC++) and our proposed TAMC. Results highlight the efficiency of our method in achieving superior task performance at various bit rates, demonstrating the necessity of task grouping and scalable encoding.

To evaluate the impact of task grouping and the DAG on performance, we conducted an ablation study where both components were removed. In this alternative setup, multiple auxiliary tasks were integrated directly into the compression framework, as illustrated in Fig. 8, optimizing the following combined loss function:

$$\mathcal{L} = \mathcal{L}_{\text{compression}} + \sum_i w_i . \mathcal{L}_{\text{task}_i} \tag{14}$$

[4] ftp://ftp.ijg.org/pub/jpeg/
[5] https://github.com/webmproject/libwebp
[6] https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM
[7] https://github.com/VincentChandelier/ELiC-ReImplemetation
[8] https://github.com/JiangWeibeta/MLIC
[9] https://drive.google.com/drive/folders/1XQVpv6Yyz5CRGNxetO0LTXuTvMS_w5R5?usp=sharing

Here, w_i represents the weight assigned to each task, we set equal weights for tasks in our ablation studies. We tested this approach on 6 tasks: Semantic Segmentation, Surface Normal, Edge Texture, Depth Z-buffer, Keypoint 2D, and Autoencoder. Fig. 9 presents the performance-rate curves for each task, comparing our method to baseline compression methods (e.g., MLIC++) and an auxiliary task-based variant (Task Auxiliary+MLIC++). The results demonstrate the advantages of our approach in both compression efficiency and task performance, as well as the complex interplay between task collaboration and conflict, which highlights the significance of task grouping and scalable encoding.

Our method consistently outperforms baseline compression methods, especially for tasks such as **Semantic Segmentation**, **Surface Normal**, **Edge Texture**, and **Depth Z-buffer**. Even at lower bit rates, our framework achieves notable improvements over MLIC++ and Task Auxiliary+MLIC++, underscoring its robustness and adaptability in multi-task settings. These results confirm the efficacy of task grouping and the integration of a causal DAG in preserving task performance under constrained compression conditions.

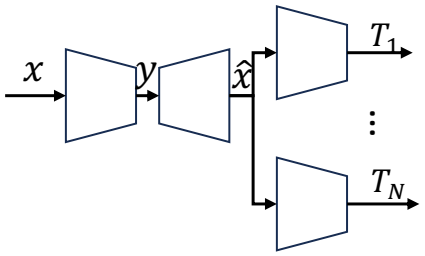


Figure 8: End-to-end compression with Multiple Task Auxiliary Loss.

The Task Auxiliary+MLIC++ variant, which replaces task grouping and DAG with auxiliary tasks, provides useful insights into task-level interactions. For tasks like **Edge Texture**, **Semantic Segmentation**, **Surface Normal**, and **Depth Z-buffer**, the auxiliary task approach yields substantial improvements compared to end-to-end compression methods, suggesting enhanced task collaboration and feature sharing. However, for **Autoencoder**, the auxiliary task approach performs similarly to end-to-end methods, indicating limited benefits for tasks with strong self-supervised structures.

In contrast, the **Keypoint 2D** task experiences performance degradation with the auxiliary task approach, likely due to task interference. This highlights the potential conflicts between task objectives, emphasizing the importance of careful task grouping to mitigate such issues.

The observed interplay of task collaboration and conflict further validates the need for task grouping. By grouping tasks based on gradient coherence, our framework minimizes inter-task interference and promotes effective task collaboration, explaining its superior performance relative to the auxiliary task-based approach. Moreover, these results show that uncoordinated task interactions can negatively impact specific tasks, such as **Keypoint 2D**.

Fig. 10 also reveals variations in bit-rate efficiency across tasks. For instance, **Semantic Segmentation** and **Surface Normal** maintain strong performance even at lower bit rates, while tasks like **Edge Texture** require higher bit rates due to the need for detailed feature representation. These findings highlight the importance of scalable encoding to accommodate the varying bit-rate needs of different tasks. By enabling task grouping and scalable encoding, our method addresses these challenges while optimizing compression efficiency.

A.3.2 Performance Comparison of Single Task vs. Grouped Tasks

To further examine the effectiveness of task groups, we trained a model based on single-task compression, where each task is treated as an independent group rather than grouping tasks together. Additionally, we set up a Grouping 1 compression model that jointly optimizes the task grouping of Semantic Segmentation, Depth Z-buffer, and Surface Normal, and a Group 2 compression model that jointly optimizes the task grouping of Surface Normal and Keypoint 2D. The results are shown in Tab.2, and further visualized in the performance curves in Fig. 10. As observed, Semantic Segmentation and Depth Z-buffer benefit from a more compact and high-precision representation when grouped in Group 1. Similarly, Keypoint 2D shows improved representation with higher compactness and precision in Group 2. Surface Normal's performance remains comparable between task grouping and the single-task approach.

One additional benefit of task grouping is the shared encoder and shared representations across multiple tasks. For example, in Group 1, a single encoder feature extraction is used for inference, consuming 0.0015 bpp for Semantic Segmentation, Depth Z-buffer, and Surface Normal. In contrast,

Method	Semantic Seg.		Depth Z-buffer		Surface Normal		Keypoint 2D	
	Bitrate	Test Loss	Bitrate	Test Loss	Bitrate	Test Loss	Bitrate	Test Loss
Single Task	0.0014	0.0852	0.0008	0.2925	0.0006	0.1315	0.0017	0.2439
	0.0055	0.0680	0.0051	0.2648	0.0052	0.0963	0.0645	0.1115
	0.0069	0.0674	0.0063	0.2643	0.0069	0.0938	0.0940	0.0954
Group 1	0.0015	0.0704	0.0015	0.2615	0.0015	0.1378	-	-
	0.0096	0.0598	0.0096	0.2419	0.0096	0.1079	-	-
	0.0139	0.0574	0.0139	0.2385	0.0139	0.1045	-	-
Group 2	-	-	-	-	0.0018	0.1528	0.0018	0.2412
	-	-	-	-	0.0623	0.1103	0.0623	0.0944
	-	-	-	-	0.0843	0.1080	0.0843	0.0936

Table 2: Performance-Bitrate comparing task grouping with single task. Group 1 represents the task grouping of Semantic Segmentation, Depth Z-buffer, and Surface Normal. Group 2 represents the task grouping of Surface Normal and Keypoint 2D.

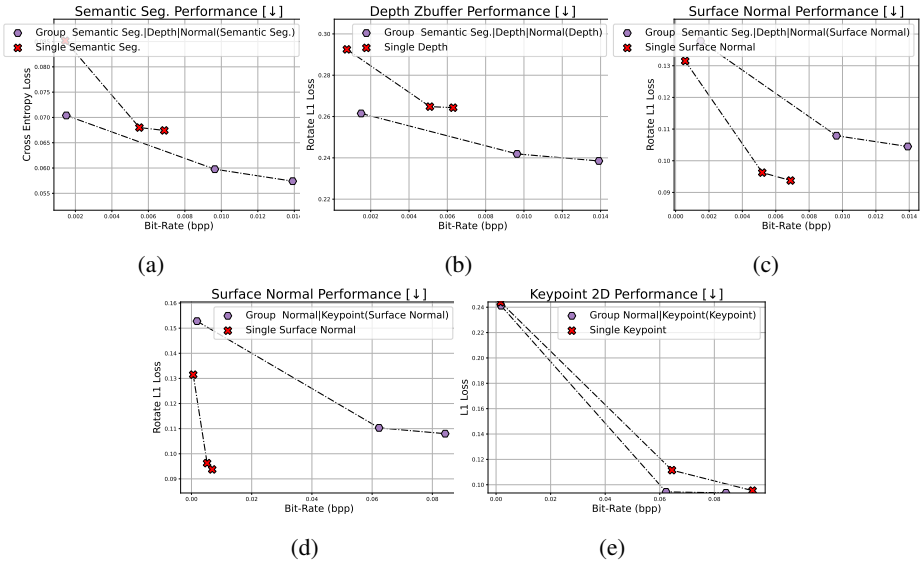


Figure 10: Performance-Bitrate curves comparing task grouping with single task. The purple curves represent the bitrate-performance performance of task grouping, while the red curves represent the performance of the single-task model.

Method		Semantic Seg.	Depth Z-buffer	Surface Normal	Edge Texture	Keypoint 2D	Total Test Loss
Setting 1	Group 1	**0.532**	**0.2527**	**0.1064**	—	—	0.5288
	Group 2	—	—	—	**0.0232**	**0.0933**	
Setting 2	Group 1	**0.0532**	**0.2527**	**0.1064**	—	—	
	Group 2	—	—	0.1096	**0.0271**	**0.0750**	0.5176

Table 3: Number of Task Grouping=2. Tasks are not allowed to appear in multiple groups in Setting 1. Tasks can appear in multiple groups in Setting 2.

treating each task independently requires three separate feature extraction inferences, with a total bitrate of 0.0014 bpp + 0.0008 bpp + 0.0006 bpp = 0.0028 bpp to serve the three tasks.

A.3.3 Impact of Task Exclusivity Across Groups on Performance

From a performance ceiling perspective, allowing the same task to appear in different clusters maximizes the potential for task collaboration. As shown in Tab. 3 and 4, we conducted experiments under two settings: Setting 1, where tasks are not allowed to appear in multiple clusters, and Setting

Method		Semantic Seg.	Depth Z-buffer	Surface Normal	Edge Texture	Keypoint 2D	Total Test Loss
Setting 1	Group 1	**0.0528**	**0.2636**	—	—	—	0.5304
	Group 2	—	—	—	**0.0232**	**0.0933**	
	Group 3	—	—	**0.0975**	—	—	
Setting 2	Group 1	0.0532	**0.2527**	0.1064	—	—	0.4862
	Group 2	**0.0500**	—	**0.1025**	**0.0242**	—	
	Group 3	—	—	0.1110	—	**0.0568**	

Table 4: Number of Task Groupings=3.

2, where tasks can appear in multiple clusters. Setting 2 achieves a superior performance ceiling, demonstrating the advantages of task interdependence. Notably, since each task is only inferred once, Setting 2 does not introduce additional inference complexity. However, it does result in a significant increase in GPU memory consumption during training. This highlights the trade-off between performance and resource utilization when task exclusivity is relaxed.

A.3.4 Impact of Task Order on Task Grouping and Addressing Variability

Regarding whether the order of tasks affects the cost calculation and how to address potential variability in the method using coherence scores: In Sec. 3.2, the number of possible groupings for n tasks is given by the Bell numbers. To quickly estimate the similarity between tasks, after calculating $\hat{C}^t_{u\to v}$, $\hat{C}^t_{w\to v}$, we estimate the higher-order costs for $\{\tau_u, \tau_v, \tau_w\}$ which significantly reduces the computational complexity.

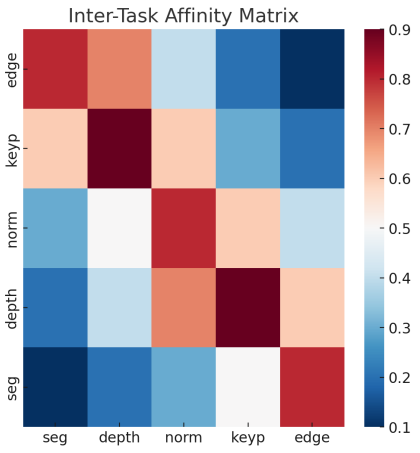


Figure 11: Inter-task coherence on Taskonomy. Red color signify higher inter-task affinities.

Although the order of tasks may influence the dynamics of gradient updates and the final learning outcomes in multi-task learning, particularly in non-convex optimization problems, we mitigate the impact of task order with the following operations:

1. The coherence score is a measure based on the impact of gradient updates between tasks on the loss function. It is a relative measure that reduces the impact of the absolute order of task execution. 2. By calculating the coherence scores throughout the entire training process and taking the average, we can mitigate the impact caused by specific stages of training, thus reducing the potential variability brought about by changes in the order of tasks. 3. To validate the reasonableness of this operation, in Fig.11, we experimentally demonstrate that although the coherence score between task pairs is not strictly symmetric ($\hat{C}^t_{u\to v} \neq \hat{C}^t_{v\to u}$), it exhibits a strong symmetry trend in practice. This allows us to approximate the values while maintaining accuracy and efficiency.

A.3.5 Comparison with VQ-GAN Compression

While VQGAN-based compression methods (Esser et al., 2021; Mao et al., 2024) achieve perceptual compression at low bitrates through discretized codebooks, our optimization goal diverges significantly. Unlike VQGAN (Mao et al., 2024), which prioritizes image reconstruction and perceptual quality, our approach focuses on compact, multi-task semantic compression. Specifically, we optimize for efficient semantic representation sharing across tasks, reducing redundancy in encoding. This contrasts with VQGAN's generative approach. Comparative experiments in Fig. 12 reveal key differences in performance and efficiency.

Although VQGAN (Mao et al., 2024) demonstrates significant improvements in compression rates over VAE backbone models, especially for perceptual tasks, it still introduces bias compared to the optimal supervision anchor, even with sufficient bitrate. This is due to its generative nature, which

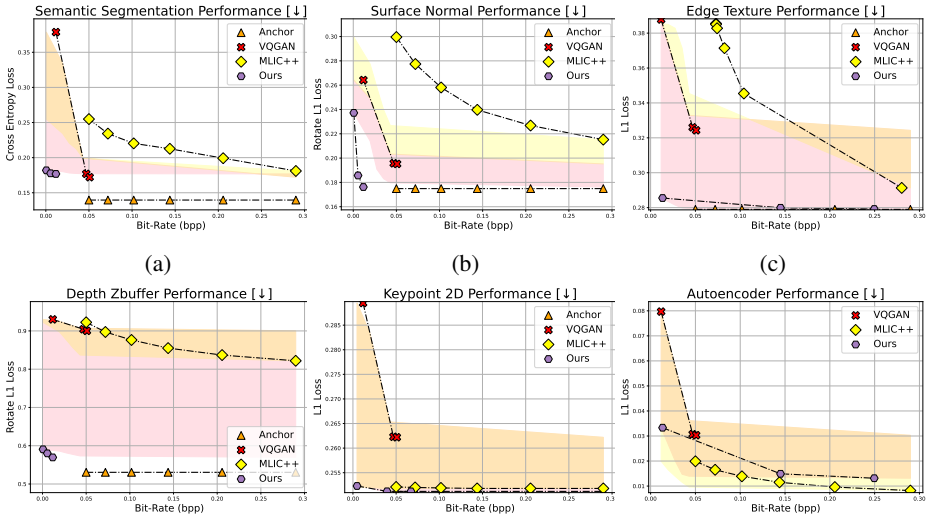


Figure 12: Comparison with VQGAN (Mao et al., 2024) in multi-task compression: Our method outperforms VQGAN in compression efficiency at low bitrates and closely matches the supervision anchor at higher bitrates, particularly in tasks like Keypoint 2D, Semantic Segmentation, and Depth Zbuffer. VQGAN, however, shows performance degradation in fine-grained tasks, highlighting the advantages of our task-aware semantic compression approach.

cannot fully eliminate reconstruction errors, leading to discrepancies from the ground truth. Moreover, VQGAN struggles with tasks requiring precise, sparse local feature detection (*e.g.*, Keypoint 2D), a limitation common in generative models that fail to capture fine-scale features. In contrast, our method is specifically designed to encode and preserve task-specific features, achieving superior performance in these tasks.

In multi-task settings (*e.g.*, Semantic Segmentation, Surface Normal, Edge Texture, Depth Zbuffer), our approach demonstrates superior compression efficiency at low bitrates and near-optimal performance at higher bitrates, closely matching the supervision anchor. VQGAN, while effective for perceptual compression, struggles to leverage task-specific semantic information, leading to inferior performance, particularly at low bitrates.

A.4 Complexity Analysis

A.4.1 Complexity Analysis of Compression for Multiple Tasks: ATC vs. CTA Paradigms

MLIC++ Encoder/Decoder Module	g_a	g_s
KParams	**12033.6**	**4396.3**
MMACs	**194556.9**	**296377.9**

Table 5: Parameters and Forward Macs of Encoder/Decoder of MLIC++ on 512 $\times$ 512 images.

Task Xception Encoder/Decoder Module	l_a	l_s
KParams	**16467.2**	**525.1**
MMACs	**25708.0**	**4968.1**

Table 6: Parameters and Forward Macs of Task Encoder/Decoder of Xception on 512 $\times$ 512 images.

In the task of compression for multiple downstream tasks, we investigate two compression paradigms: Analysis and Then Compression (ATC) and Compression and Then Analysis (CTA). ATC pipeline

Context Module	$\boldsymbol{\theta}_{hd}$	$\boldsymbol{\theta}_{cm}$	$\boldsymbol{\theta}_{cm+}$	$\boldsymbol{\theta}_{gc}$
KParams	**5810.1**	**755.2**	**1487.7**	**2264.4**
MMACs	**8925.1**	**1148.2**	**2264.4**	**7100.5**

Table 7: Parameters and Forward Macs of entropy context modules on 512 × 512 images.

consists of two main phases: the downstream task analysis phase, where the input image is used for specific tasks such as keypoint detection, segmentation, and depth estimation, and the feature compression phase, which includes the encoding and decoding steps. The CTA pipeline also consists of two main phases: the image compression phase, which includes encoding and decoding, and the downstream task analysis phase, where the compressed image is used for tasks such as keypoint detection, segmentation, and depth estimation. To better understand the computational costs involved in each module, we summarize the parameters and forward MACs (Multiply-Accumulate Operations) of the different components in the ATC and CTA pipelines. These values are presented in the tables 5, 6, 7. Below, we present the complexity analysis for both approaches.

Our proposed method belongs to ATC paradigm, and the total computational complexity of this pipeline results from both the downstream task analysis and the feature compression phases. The number of downstream tasks N and the number of task groups K directly affect the computational cost of the task analysis phase. The overall complexity is expressed as:

$$\begin{aligned} \text{Total Complexity of ATC} &= N \cdot l_s + K \cdot (l_a + \boldsymbol{\theta}_{hd} + \boldsymbol{\theta}_{cm+} + \boldsymbol{\theta}_{gc}) \\ &= N \cdot 4968.1 + K \cdot 40088.0, \end{aligned} \tag{15}$$

where $\boldsymbol{\theta}_{hd}$, $\boldsymbol{\theta}_{cm+}$, and $\boldsymbol{\theta}_{gc}$ represent the complexities of the context modules. N is the number of downstream tasks, K is the number of task groups, l_a and l_s represent the complexities of the Task Xception Encoder/Decoder for each individual task.

This formulation indicates that the task analysis phase (which involves both l_a and l_s) and the context module complexities contribute to the total computational cost.

The total computational complexity in the CTA pipeline arises from both the compression and the downstream task analysis phases. The number of tasks N directly affects the computational cost of the task analysis phase. The overall complexity is expressed as:

$$\begin{aligned} \text{Total Complexity of CTA} &= g_a + g_s + \boldsymbol{\theta}_{hd} + \boldsymbol{\theta}_{cm} + \boldsymbol{\theta}_{gc} + N \cdot (l_a + l_s) \\ &= 500008.6 + N \cdot 30676.1, \end{aligned} \tag{16}$$

where g_a and g_s are the complexities of the MLIC++ Encoder/Decoder. $\boldsymbol{\theta}_{hd}$, $\boldsymbol{\theta}_{cm}$, and $\boldsymbol{\theta}_{gc}$ represent the complexities of the context modules. N is the number of downstream tasks. l_a and l_s represent the complexities of the Task Xception Encoder/Decoder for each individual task.

This formulation emphasizes the contribution of the encoding/decoding processes (represented by g_a and g_s) as well as the task-specific encoding/decoding complexities l_a and l_s in the CTA pipeline.

A.4.2 Complexity Analysis of Our Lookahead Module

Our Lookahead module consists of two steps: task grouping and DAG construction. In the task grouping step, we first compute the coherence score between tasks using joint training on $N \times N/2$ task pairs. Then, we use pairwise coherence scores to estimate the coherence scores of higher-order task groupings. Finally, based on all task groups, task coherence scores and budget b, we select k multitask networks to maximize the overall task performance. This is an NP-hard problem. A brute force approach would take $O(|\mathcal{T}| \cdot |\boldsymbol{C}_0|^{\frac{b}{\min_{n \in \boldsymbol{C}_0} c_n}})$, which is exponential in the maximum number of groups that fit within the budget. Here, $|\mathcal{T}|$ is the number of tasks, $|\boldsymbol{C}_0|$ is the number of candidate networks, b is the budget, and $\min_{n \in \boldsymbol{C}_0} c_n$ is the smallest cost among the networks. This can be solved using the branch-and-bound-like algorithm provided in prior work (Standley et al., 2020; Zamir et al., 2018) as detailed in Sec.A.5.

The overall time complexity of the DAG construction algorithm 1 is dominated by the nested loops that compute conditional and independent entropies for each latent group. The algorithm iterates over all K groups, leading to an outer loop complexity of $O(K)$. For each group, the algorithm

computes the conditional entropy between pairs of latent variables. This operation takes $O(n)$ time, where n is the size of the dataset (number of samples). Additionally, the algorithm computes the independent entropy for each group, which takes $O(m)$ time, where m represents the complexity of entropy calculation for a single group Thus, the overall time complexity is:

$$O(K^2 \cdot n \cdot m),$$

where K is the number of groups, n is the time complexity for calculating conditional entropy, m is the time complexity for calculating independent entropy for a group.

The process of computing task coherence scores, grouping tasks, and constructing the corresponding Directed Acyclic Graph (DAG) is computationally intensive. These steps share a conceptual similarity with the lookahead stage in traditional video encoding (Li, 2003; He & Mitra, 2002; Wang & Kwong, 2008; Ma et al., 2005). In video encoding, the lookahead stage performs a preliminary analysis of the video content to optimize the encoding process, ensuring that the final compression achieves the best trade-off between quality and bitrate. Specifically, it evaluates potential encoding decisions ahead of time to minimize redundancies and improve efficiency, all while adhering to bitrate constraints.

Similarly, in multi-task compression, the task grouping and causal relationship modeling steps aim to optimize the encoding of multiple tasks by leveraging the inherent interdependencies among them. However, due to the high computational complexity of calculating coherence scores between tasks and determining the optimal task groupings, an efficient pre-analysis phase is essential. Our contribution lies in exploring the effectiveness of task grouping and cross-task causal relationship modeling, demonstrating that these techniques can significantly enhance the performance of multi-task compression.

To make this process feasible and scalable, further exploration can involve using downsampled low-resolution images as inputs for the pre-analysis phase. This strategy, when combined with a low-complexity feature extraction backbone (Iandola, 2016), provides an efficient means of assessing task grouping performance under bitrate consumption constraints.

A.5 Implementation details of task grouping

As mentioned in Sec. 3.2, we adopt a branch-and-bound method, as in prior works (Standley et al., 2020; Zamir et al., 2018), to search for locally optimal grouping strategies under the given bitrate budget b. We here provide pseudo algorithm for clarity and the implementation is provided in Github[10]. Consider the situation in which we have an initial candidate set $\boldsymbol{C}_0 = \{n_1, n_2, ..., n_m\}$ of fully-trained networks that each solve some subset of our task set $\mathcal{T}$. Our goal is to choose a subset of $\boldsymbol{C}_0$ that solve all the tasks under budget b and the lowest overall loss. More formally, we want to find a solution $\boldsymbol{S}_b = \arg\min_{\boldsymbol{S} \subseteq \boldsymbol{C}_0 : \text{cost}(\boldsymbol{S}) \leq b} \mathcal{L}(\boldsymbol{S})$. It can be shown that solving this problem is NP-hard in general (reduction from Set-Cover). A brute force approach would take $O(|\mathcal{T}| \cdot |\boldsymbol{C}_0|^{\frac{b}{\min_{n \in \boldsymbol{C}_0} c_n}})$, which is exponential in the maximum number of groups that fit in our budget. This would be computationally challenging even for small problems.

However, many techniques exist that can optimally solve *most* instances of problems like these in reasonable amounts of time. All of these techniques produce solutions that perform equally well. We chose to use a branch-and-bound-like algorithm for finding this optimal solution (shown in Algorithm 2), but in principle the same solution could be achieved by other optimization methods, such as encoding the problem as a binary integer program (BIP) and solving it in a way similar to Taskonomy(Zamir et al., 2018). Algorithm 2 chooses the best subset of groups in our collection, subject to the inference time budget constraint. The algorithm recursively explores the space of solutions and prunes branches that cannot lead to optimal solutions. The recursion terminates when the budget is exhausted, at which point $\boldsymbol{C}_r$ becomes empty and the loop body does not execute. The sorting step on line 3 requires a heuristic upon which to sort. We found that ranking models based on how much they improve the current solution, $\boldsymbol{S}$, works well. It should be noted that this algorithm always produces an optimal solution, regardless of which sorting heuristic is used. However, better sorting heuristics reduce the running time because subsequent iterations will more readily detect and prune portions of the search space that cannot contain an optimal solution.

[10] https://github.com/tstandley/taskgrouping/

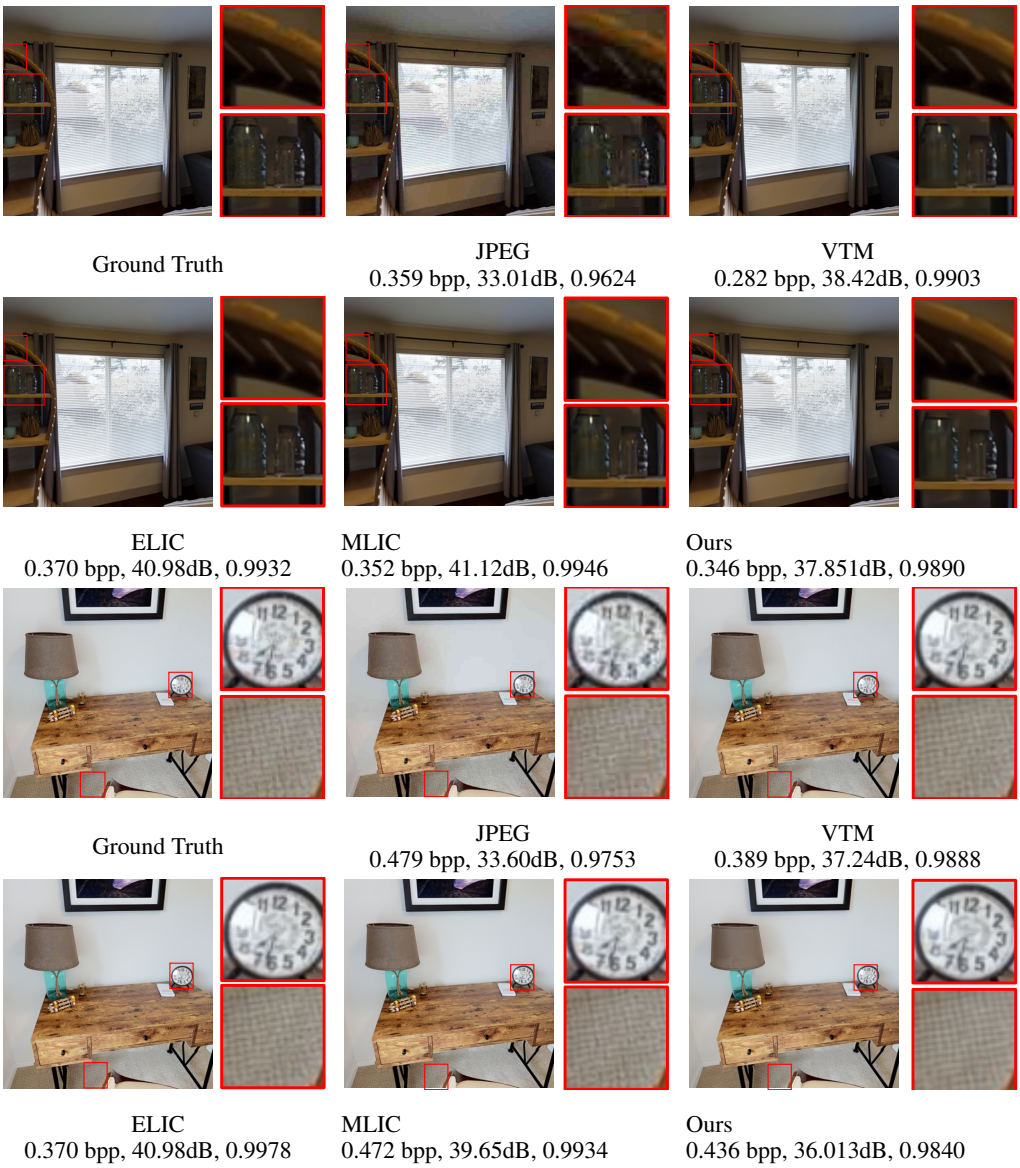


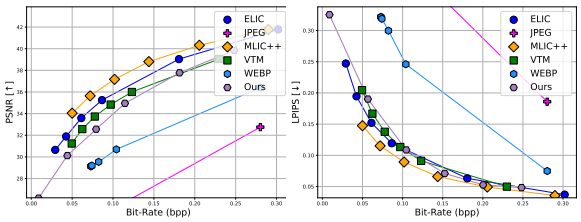
Figure 13: Subjective comparisons.

A.6 Subjective Results

In Fig. 13, we present examples showing competitive qualitative results from our method compared to VTM and end-to-end compression methods.

A.7 More Experimental Results

we present the detailed experimental results comparing the performance of our proposed TAMC with several baseline image compression methods on the Taskonomy dataset. Figure 14 provides a detailed comparison of PSNR-Bitrate and LPIPS-Bitrate performance.



(a) (b)

Figure 14: Comparison of PSNR-Bitrate performance and LPIPS (Zhang et al., 2018)-Bitrate performance for image compression on the Taskonomy dataset, using baseline methods and our proposed TAMC.

Algorithm 2 Get Task Grouping Strategy

Input: $\boldsymbol{C}_r$, a running set of candidate groups, each with an associated cost $c \in \mathbb{R}$ and a performance score for each task the network solves. Initially, $\boldsymbol{C}_r = \boldsymbol{C}_0$
Input: $\boldsymbol{S}_r \subseteq \boldsymbol{C}_0$, a running solution, initially
Input: $b_r \in \mathbb{R}$, the remaining time budget, initially b

function GETBESTNETWORKS($\boldsymbol{C}_r, \boldsymbol{S}_r, b_r$)
  $\boldsymbol{C}_r \leftarrow$ FILTER($\boldsymbol{C}_r, \boldsymbol{S}_r, b_r$)
  $\boldsymbol{C}_r \leftarrow$ SORT($\boldsymbol{C}_r$) ▷ Most promising groups first
  $\boldsymbol{Best} \leftarrow \boldsymbol{S}_r$
  for $n \in \boldsymbol{C}_r$ **do**
    $\boldsymbol{C}_r \leftarrow \boldsymbol{C}_r \setminus n$ ▷ $\setminus$ is set subtraction.
    $\boldsymbol{S}_i \leftarrow \boldsymbol{S}_r \cup \{n\}$
    $b_i \leftarrow b_r - c_n$
    $\boldsymbol{Child} \leftarrow$ GETBESTNETWORKS($\boldsymbol{C}_r, \boldsymbol{S}_i, b_i$)
    $\boldsymbol{Best} \leftarrow$ BETTER($\boldsymbol{Best}, \boldsymbol{Child}$)
  end for
  return $\boldsymbol{Best}$
end function

function FILTER($\boldsymbol{C}_r, \boldsymbol{S}_r, b_r$)
  Remove groups from $\boldsymbol{C}_r$ with $c_n > b_r$.
  Remove groups from $\boldsymbol{C}_r$ that cannot improve $\boldsymbol{S}_r$'s performance on any task.
  return $\boldsymbol{C}_r$
end function

function BETTER($\boldsymbol{S}_1, \boldsymbol{S}_2$)
  if $C(\boldsymbol{S}_1) < C(\boldsymbol{S}_2)$ **then**
    return $\boldsymbol{S}_1$
  else
    return $\boldsymbol{S}_2$
  end if
end function