Feature Coding in the Era of Large Models: Dataset, Test Conditions, and Benchmark

Supplementary Material

1. Details of baseline methods

1.1. Packing details

1.1.1. Semantic segmentation

In semantic segmentation, an image is pre-processed into multiple 518×518 patches by crop and resize operations. These patches are fed into DINOv2, which produces features of shape 1370×1536. In the proposed dataset, each image is pre-processed into 2 patches, resulting in features with a shape of $2 \times 1370 \times 1536$. We reshape F_{Seg} from $2 \times 1370 \times 1536$ into 2740×1536 by vertically stacking them.

1.1.2. Depth estimation

In depth estimation, an image is first horizontally flipped. Both the original and flipped images are padded to dimensions that are multiples of 14 before being processed by DINOv2. For each image, DINOv2 generates $4 \times 1611 \times 1536$ features at the split point SP_{DM}. Thus, the shape of F_{Dpt} is $2 \times 4 \times 1611 \times 1536$. F_{Dpt} is packed in two steps. First, each $4 \times 1611 \times 1536$ feature is horizontally packed into a 1611×6144 feature by concatenating the features along the width. Second, the two 1611×6144 features (original first and then flipped images) are vertically stacked, forming a 3222×6144 feature.

1.1.3. Text-to-image synthesis

For text-to-image synthesis, the original feature F_{TTI} has a shape of $16 \times 128 \times 128$, where 16 represents the number of channels. We divide the 16 channels into 4 subgroups. Within each subgroup, the 128×128 features are horizontally concatenated to form a 128×512 feature. Finally, the four 128×512 features are vertically stacked to produce a final packed feature of size 512×512.

1.2. Training details

1.2.1. Training dataset construction

We construct our training dataset from public datasets. The training features are extracted in the same way as the test dataset. The source code will be made publicly available, and we encourage researchers to extract features from their own source data.

As mentioned in the main text, raw features consume a significant amount of storage. During training, we encountered storage challenges. For example, 20000 original F_{Dpt} features occupy 1474 GB of storage. As the demand for additional training data grows, storage requirements could increase substantially, often exceeding the resources available to researchers in academic institutions. To address this issue, we crop the features into smaller dimensions to facilitate feature coding training. The specific cropping configurations are detailed in Table 1.

It is worth emphasizing that while cropping is effective for feature coding training, this strategy is not applicable to large model training. For large models, cropping features to accelerate training is impractical, as downstream models typically require complete semantic information. For example, cropping a 256×256 patch from F_{Dpt} would make it challenging to perform accurate depth estimation. Therefore, designing efficient feature coding methods remains crucial for reducing the storage burden in large model deployments.

After feature extraction, the same truncation and quantization operations are applied to the training dataset. For the Hyperprior baseline, all features are normalized to the range [0, 1] before training.

1.2.2. Hyperparameter Setting

The hyperparameters, such as the number of training epochs, initial learning rate, and λ values, are provided in Table 1. The number of training epochs is determined based on the size of the training dataset. In our experiments, all training processes converge to a stable loss. Five different λ values are used to achieve various bitrates.

2. Additional experimental results

2.1. Complexity analysis

We present the encoding and decoding complexity analyses in Table 2. The VTM baseline and Hyperprior baseline are evaluated on an Intel® Xeon® E5-2690 v4 CPU and a single NVIDIA GeForce RTX 4090 GPU, respectively. It is important to note that the VTM baseline is a handcrafted codec and cannot be executed on a GPU.

For the VTM baseline, encoding time increases as QP decreases and feature resolution increases. The block-based VTM codec exhaustively examines multiple partitions and modes during encoding, making it significantly more time-consuming than decoding. VTM is one of the reference implementations of the VVC standard. In practical deployments, a more efficient VVC codec, such as VVEnc, can be used to accelerate encoding. For the Hyperprior baseline, the decoding complexity is slightly higher than the encoding complexity. Across different bitrates, the encoding and decoding complexities remain comparable.

Task	Source Dataset	Number of Samples	Feature Shape	Epoch	Learning Rate	Lambda
Cls	ImageNet	5000	256×256	800	1e-4	0.001, 0.0017, 0.003, 0.0035, 0.01
Seg	VOC2012	5000	256×256	800	1e-4	0.0005, 0.001, 0.003, 0.007, 0.015
Dpt	NYUv2	20000	256×256	200	1e-4	0.001, 0.005, 0.02, 0.05, 0.12
CSR	Arc-Challenge, OpenBookQA	6000	64×4096	200	1e-4	0.01405, 0.0142, 0.015, 0.15, 10
TTI	Captions from COCO2017	50000	512×512	60	1e-4	0.005, 0.01, 0.02, 0.05, 0.2

Table 1. Training details of the Hyperprior baseline.

Task	**Image Classification**			**Semantic Segmentation**			**Depth Estimation**			**Common Sense Reasoning**			**Text-to-Image Synthesis**		
Metric	**QP**	**Enc.**	**Dec.**	**QP**	**Enc.**	**Dec.**	**QP**	**Enc.**	**Dec.**	**QP**	**Enc.**	**Dec.**	**QP**	**Enc.**	**Dec.**
VTM Baseline	22	267.20	0.23	22	2301.30	1.11	22	14169.36	5.15	22	336.27	0.24	22	107.98	0.16
	27	275.02	0.22	27	2039.34	0.94	27	10862.76	3.54	27	423.84	0.29	27	87.33	0.15
	32	124.40	0.15	32	1399.71	0.52	32	7776.30	2.33	32	362.98	0.24	32	62.11	0.15
	37	28.00	0.13	37	407.30	0.32	37	1590.27	1.14	37	162.05	0.17	37	32.38	0.24
	42	8.55	0.14	42	70.66	0.27	42	348.68	1.17	42	66.28	0.18	42	15.70	0.15
Metric	λ	**Enc.**	**Dec.**	λ	**Enc.**	**Dec.**	λ	**Enc.**	**Dec.**	λ	**Enc.**	**Dec.**	λ	**Enc.**	**Dec.**
Hyperprior Baseline	0.01	0.04	0.06	0.015	0.3	0.38	0.12	1.57	2.05	10	0.07	0.12	0.2	0.03	0.03
	0.0035	0.04	0.05	0.007	0.31	0.4	0.05	1.59	2.05	0.07	0.07	0.09	0.05	0.03	0.03
	0.003	0.04	0.05	0.003	0.31	0.39	0.02	1.55	1.88	0.015	0.06	0.08	0.02	0.03	0.03
	0.0017	0.05	0.06	0.001	0.29	0.35	0.005	1.54	1.84	0.0142	0.07	0.08	0.01	0.03	0.03
	0.001	0.05	0.06	0.0005	0.29	0.36	0.001	1.57	1.91	0.01405	0.06	0.08	0.005	0.03	0.03

Table 2. Encoding and decoding complexity evaluations. "Enc." and "Dec." denote encoding time and decoding time, respectively. The running time is measured in seconds.

2.2. Generalizability analysis

Here, we visualize the λ-BPFP curves and the rate-accuracy curves for the cross evaluation to provide a more intuitive presentation of the results. The λ-BPFP curves reveal that the same λ produces different rate-distortion trade-offs across various tasks, indicating that features extracted from different tasks do not share similar coding properties. From the rate-accuracy curves, we observe that coding models trained for one task struggle to achieve comparable performance when applied to another task. The generalizability of feature coding methods across multiple tasks has received limited attention in prior research. By presenting these results, we aim to highlight this important research direction and encourage further exploration.

2.3. Ablation on truncation and quantization

In this subsection, we examine the impact of truncation and quantization on feature coding. As shown in Table 4, truncation and quantization individually result in little task accuracy loss. When both are applied together, task accuracies experience slight decreases. However, applying the VTM baseline to quantized features leads to a significant reduction in accuracy.

Without truncation, feature values are often limited to a small subset within the full range of [0, 1023]. In this scenario, during feature coding, the codec's internal quantization maps values with a small range to a single value, introducing significant distortion. By applying truncation, this range is expanded, which helps mitigate distortion caused by VTM's internal quantization. To demonstrate this effect, we test several truncation ranges and encode the truncated features using fixed QPs (27, 32, 37). As presented in Table 5, smaller truncation ranges generally yield higher accuracy but come at the cost of increased bitrates. Therefore, selecting an appropriate truncation range is crucial for balancing the target bitrates and accuracy.

2.4. Distribution analysis on decoded features

We visualize the distribution of reconstructed features in Fig. 2, comparing the two baseline methods at similar bitrates. For SP_{DM}, the Hyperprior baseline fails to reconstruct features within their original distribution range. The reconstructed features appear truncated during the feature coding process. We attribute this truncation to the Hyperprior's internal quantization (rounding operation), where the transformed latent variables are directly rounded to integers before entropy coding. In contrast, the VTM baseline quantizes original features into a broader distribution range of [0, 1023], effectively preventing the truncation phenomenon. A similar truncation phenomenon is observed for SP_H. For SP_{GS}, the features are encoded at a higher bitrate and the truncation phenomenon is not observed.

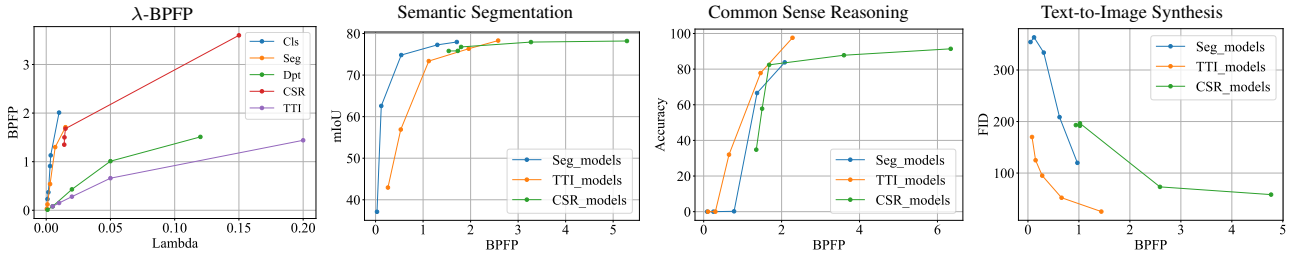

Figure 1. λ-BPFP curves and R-A curves of generalizability evaluation on Seg, CSR, and TTI tasks.

	Proposed Features						Existing Features					
Source Dataset	**Split Point**	**Max**	**Min**	**Mean**	**IV**	**GM**	**Split Point**	**Max**	**Min**	**Mean**	**IV**	**GM**
ImageNet	SP_{DS}	79.01	-502.04	0.0782	17.04	3.97	$C5$-ReLU	7.81	0	0.4594	9839.49	143.22
VOC2012	SP_{DS}	94.88	-485.23	0.0780	13.30	4.30	$C5$	8.00	-4.24	0.2473	7323.72	131.90
NYUv2	SP_{DM1}	3.25	-2.33	-0.0015	128.91	17.92	$p2$	10.83	-11.11	-0.1726	8452.88	159.45
	SP_{DM2}	4.99	-26.08	-0.0005	36.96	10.00	$p3$	8.35	-9.02	-0.1277	9418.53	168.38
	SP_{DM3}	23.51	-314.26	0.0042	9.61	2.76	$p4$	9.06	-9.93	-0.0629	8499.73	158.89
	SP_{DM4}	96.04	-493.17	0.0731	12.54	4.35	$p5$	9.61	-8.22	-0.0295	9402.06	169.13
Arc-Challenge	SP_{GS}	27.50	-26.38	0.0038	419.14	33.28	/	/	/	/	/	/
Captions from COCO2017	SP_{H}	3.67	-2.11	0.5664	21130.12	138.53	/	/	/	/	/	/

Table 3. Statistics of the proposed features and existing features. (Gradient magnitude values are scaled by 10^5).

Configuration	**Cls**	**Seg**	**Dpt**	**CSR**
Truncation Only	100	81.60	0.4992	100
Quantization Only	99	81.37	0.4982	100
Truncation + Quantization	100	79.93	0.4941	100
Quantization + VTM QP22	0	5.35	1.6085	6

Table 4. Ablations on truncation and quantization operations.

Image Classification				**Semantic Segmentation**			
[-5, 5]		**[-30, 30]**		**[-5, 5]**		**[-30, 30]**	
BPFE	**Acc.**	**BPFE**	**Acc.**	**BPFE**	**mIoU**	**BPFE**	**mIoU**
2.90	100	0.38	91	2.84	79.83	0.37	76.38
2.05	100	0.06	24	1.98	79.40	0.08	63.46
1.15	100	0.02	14	1.10	78.68	0.01	39.32
Depth Estimation				**Common Sense Reasoning**			
[-5, 5], [-5, 5]		**[-20, 20], [-30, 30]**		**[-2, 2]**		**[-10, 10]**	
BPFE	**RMSE**	**BPFE**	**RMSE**	**BPFE**	**Acc.**	**BPFE**	**Acc.**
1.42	0.6490	0.58	0.8569	3.16	98	0.84	98
0.73	0.9604	0.17	1.1360	2.30	99	0.15	82
0.33	1.2453	0.04	1.3073	1.44	99	0.04	20

Table 5. Ablation results on truncation regions. SP_{DM1} and SP_{DM2} features are truncated into [-1, 1] and [-2, 2] in all cases, respectively.

3. Additional dataset analysis

In addition to the visualizations provided in the main text, we present further statistical analyses of the proposed dataset.

3.1. Distribution analysis

We compare the statistical characteristics of the proposed features with those of existing features by randomly sampling 10 examples from each dataset and reporting their maximum, minimum, and mean values in Table 3. Overall, the proposed features exhibit a broader distribution range compared to existing features. In particular, for multi-scale features, the distribution range expands in deeper network layers, whereas existing features exhibit smaller variations. Moreover, although the proposed features tend to be more asymmetric, their mean values remain closer to zero.

3.2. Redundancy analysis

To evaluate spatial redundancy, we compute intensity variance (IV) and gradient magnitude (GM), where gradients are derived using the Sobel operator. Before these computations, both pixel and feature values are quantized to 10-bit integers. The intensity variance of the proposed features spans a wide range, from 9.61 to 22918.26, whereas existing features have a narrower range from 7323.72 to 9839.49. A similar trend is observed in gradient magnitude. These broader variations underscore the higher diversity of the proposed features. Furthermore, among the proposed features, textual features reveal distinct distribution characteristics compared to visual features. For instance, SP_{GS} and SP_{H} exhibit larger intensity variance and gradient magnitude. This observation highlights the necessity and impor-

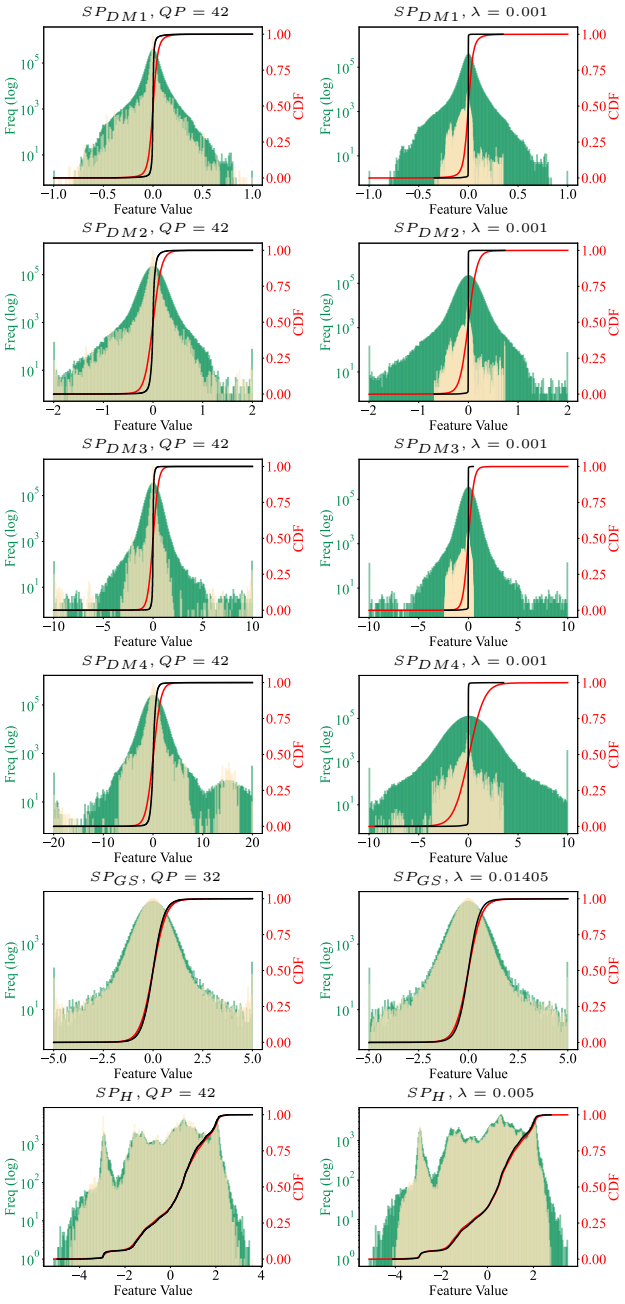

Figure 2. Frequency and CDF visualization of the original truncated and reconstructed features. Green and red are for the original truncated features. Yellow and black are for the reconstructed features. Left: the VTM baseline. Right: the Hyperprior baseline.

tance of incorporating textual features in the dataset.

3.3. Visualization

We visualize the distributions, feature blocks, and DCT blocks of F_{Cls} and F_{Seg} extracted from SP_{DS} in Fig. 3 and Fig. 4. These distributions exhibit high similarities

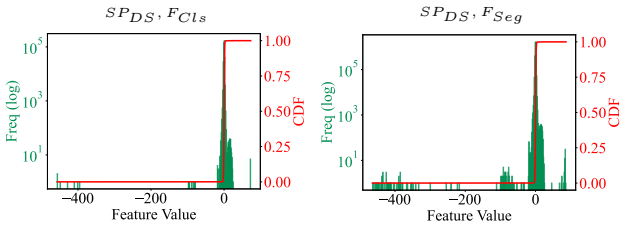

Figure 3. Visualization of the frequency distribution and CDF for the F_{Cls} and F_{Seg} features.

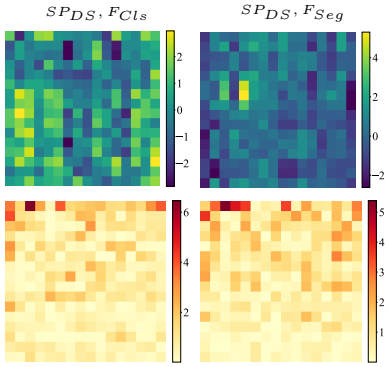

Figure 4. Visualization of the original feature blocks and their corresponding DCT blocks for the F_{Cls} and F_{Seg} features.

to the features extracted from SP_{DM4}, as they are derived from the same layer. To avoid redundancy in the main paper, these visualizations have been moved here.