10th International Workshop on
ON-BOARD PAYLOAD DATA COMPRESSION
12-14 October 2026
Barcelona - SPAIN

COMPRESSED LEARNING BASED SEMANTIC COMPRESSION UTILIZING CCSDS 123.0-B-2

Protim Bhattacharjee, Peter Jung

German Aerospace Center (DLR),
Rutherfordstr. 2, 12489 Berlin, Germany
Email: protim.bhattacharjee@dlr.de

ABSTRACT

Downlink is a central bottleneck in Earth observation. Semantic compression transmits only the features required by a downstream task rather than the full imagery. Learned feature coders, however, require per task trained entropy models and can be computationally expensive to run onboard. We present a study of the standardized CCSDS-123.0-B-2 codec as a task-agnostic, near-lossless entropy coder for neural features one of the few standards offering a tunable, bounded per-sample error up to the input bit depth. We treat a feature as a width-one multispectral cube and cast its compression as a multiple measurement vector (MMV) problem with a compressed learning linear sketch, considering both a random Gaussian and a learned sketch matrix. To enable training, we approximate the codec's quantization with a uniform straight-through quantizer and show that this surrogate tracks the real CCSDS bitrate across the absolute error limit sweep. We then investigate neural feature properties, in particular spectral axis smoothness, that benefit the predictor, and use a residual ratio penalty to learn smoother sketch measurements. Finally, on EuroSAT-MS we benchmark CCSDS based feature compression against a learned, task-specific coder, and discuss the trade-offs that make a standardized, task-agnostic codec a practical onboard choice.

INTRODUCTION

AI has enabled faster access to data analytics in Earth observation (EO) through imagery available from remote platforms. With the increase of the no. of remote platforms and the diversity and complexity of payload data generated, the major bottleneck is the downlink of the acquired data. Sentinel-2 payloads acquire, store, and downlink approximately 1.6 TB of data per orbit [1]. Most modern AI models have an encoder-decoder architecture, the output of the encoder, called the feature or the embedding, is required by the decoder to produce the analytics for a given task, e.g., classification, segmentation, and the like, and thus access to the imagery is not always necessary. This motivates us to study semantic compression, where one only downlinks the information necessary for the specific task. In the context of neural image compression (NIC) learnable feature compression has been studied [2]. Even though now lossless NIC algorithms are possible [3], the compute complexity of these models is still very high and one still needs to learn a task specific entropy encoder. Also, in space/EO missions' reliability of the compressed payload data from off-the-shelf compressors is essential and the standardization plays an important role here. To these ends, in this work we propose and explore the possibility to utilize the CCSDS 123.0-B-2 standard [4] as a near-lossless compressor of the features. This ensures that the features required for the downstream task are communicated in a near-lossless fashion and also allows for task-agnostic entropy encoding compared to other standard feature compression methods [2]. Thus, we utilize the standard to do semantic compression. In the rest of this article, we use CCSDS as the stand in for CCSDS 123.0-B-2 standard.

Features are not raw cube and generally are vectors in some high dimensional space. Feature maps few layers above the model architecture might be used, however, the literature generally considers the penultimate layer after the non-linearity as the feature vector. Thus, we treat the features as a multispectral data cube with an image width of 1. Keeping on-board compute constraints in mind, we study compressed learning [5] based linear sketch matrices as feature extractors. Gaussian random matrices and learned matrices are considered. For the so-called decoder, linear decoders are considered. Decoders can be compute heavy as they are expected to work on-ground, however, there is a trade-off between the information retained by the cheap on-board sketch and the decoder capacity needed to recover it. In order to convert the floating-point data to digital numbers (DN) integers that are accepted datatype of the input to the standard, a straight through estimator (STE) style quantizer is considered. A worst-case uniform quantizer with step size equal to the $2a_z + 1$, the maximum allowed absolute error in band z [4], is utilized. As the standard employs a residual predictor, it benefits from the smoothness along the spectral dimensions. As it is not evident that features from a feature extractor backbone

would have such smoothness, we explore the nature of feature smoothness and try to understand whether it plays a role in its compression.

In the next section, we describe the studied architecture and losses in detail, and provide the learning procedure and identify the components necessary for the CCSDS encoder. In the results section we present our findings and conclude the article with a discussion section.

ARCHITECTURE AND LOSS DESIGN

The basic compressed learning framework of [5] is used to design and study linear sketches. The schematic is shown in Fig. 1. To ensure preservation of at least the spectral smoothness of a multi-/hyperspectral datacube, the sketching problem is modelled as a multi-measurement vector (MMV) problem. A raw datacube of dimensions $\boldsymbol{n_1 \times n_2 \times p}$, with $\boldsymbol{n_1}$ and $\boldsymbol{n_2}$ being the spatial coordinates and $\boldsymbol{p}$ the spectral coordinates, is reshaped into a data matrix, $\boldsymbol{X}$ of dimensions $\boldsymbol{n \times p}$, with $\boldsymbol{n = n_1 \times n_2}$. Therefore, a sketching matrix, $\boldsymbol{A_\phi} \in \mathbb{R}^{m \times n}$, $\boldsymbol{\phi}$ is a learnable parameter, that produces a matrix, $\boldsymbol{Y = AX}$ of size $\boldsymbol{m \times p}$, mixes spatial information but preserves the spectral smoothness. The matrix multiplication is done is floating-point that converts the raw datacube and eventually Y to floating-point. However, the standard takes integers as inputs. To this we use a straight-through-estimator (STE) based quantizer. It has two functions, one to convert the floating-point Y to integers and second to approximate the quantization process inside CCSDS. The scale and offset for the datacube is evaluated and $\widetilde{\boldsymbol{Y}} = \boldsymbol{scale}.\boldsymbol{Y} + \boldsymbol{offset}$. To this uniform noise, $\boldsymbol{Y_q = round}(\widetilde{\boldsymbol{Y}}) + \boldsymbol{U}[-\boldsymbol{a_z}, +\boldsymbol{a_z}]$, is added and the input to the CCSDS is $\boldsymbol{Y_{DN}} = \boldsymbol{clip}(\boldsymbol{Y_q}, \boldsymbol{0}, \boldsymbol{2^D - 1})$, where $\boldsymbol{D}$ is the bit depth of the raw datacube. The data matrix $\boldsymbol{Y_{DN}}$ is presented to the standard as a $\boldsymbol{p \times m \times 1}$ datacube, i.e., a datacube with width = 1. The output of CCSDS is converted back to floating-point, $\widehat{\boldsymbol{Y}} = (\boldsymbol{Y_{DN}} - \boldsymbol{offset})/\boldsymbol{scale}$. A linear layer then maps, $\widehat{\boldsymbol{Y}}$ to $\widehat{\boldsymbol{X}}$, which is then input to the inference network. Unlike [2], where a task-specific feature is compressed by a learned entropy coder, we compress a task-agnostic linear sketch of the datacube with the standardized CCSDS codec, whose near-lossless mode bounds the per-sample error of the transmitted sketch by $\boldsymbol{a_z}$, and we recover the datacube with a recovery network, $\boldsymbol{f_\theta}$. The recovery network considered here is a simple single layer fully-connected network. The output from the recovery network is then fed to the downstream inference network. The training takes place in three stages. In the first stage, the sketch matrix, $\boldsymbol{A_\phi}$, and the decoder, $\boldsymbol{f_\theta}$, are learned together without the STE-quantizer for reconstruction loss, $\boldsymbol{\mathcal{L}_{recon}}(\boldsymbol{X}, \widehat{\boldsymbol{X}}) = \boldsymbol{MSE}(\boldsymbol{X}, \widehat{\boldsymbol{X}})$, as the mean-squared-error (MSE) of the raw cube and the reconstructed cube, and a smoothness loss promoting the smoothness over the spectral dimension of $\boldsymbol{Y}$. The residual ratio (RR) loss for a given band z and its neighbour z + 1 can be written as $\boldsymbol{\mathcal{L}_{RR}} = {}^{\mathrm{Var}(\boldsymbol{\Delta Y})}\big/_{\mathrm{Var}(\boldsymbol{Y})}$, where $\mathrm{Var}(\boldsymbol{\Delta Y}) = \mathrm{Var}_{\boldsymbol{z}}(\boldsymbol{Y}[\boldsymbol{z}] - \boldsymbol{Y}[\boldsymbol{z+1}])$, is the variance of the band differences. It measures the variance of adjacent band differences, normalized by the variance of the sketched measurements. The total loss for this stage is $\boldsymbol{\lambda_{recon}\mathcal{L}_{recon} + \lambda_{RR}\mathcal{L}_{RR}}$, where $\boldsymbol{\lambda_{recon}}$ and $\boldsymbol{\lambda_{RR}}$ are weighting parameters. In the second stage, the STE-quantizer is introduced, and path of $\boldsymbol{Y}$ to $\boldsymbol{Y_{DN}}$ to $\widehat{\boldsymbol{Y}}$ is placed into the reconstruction path approximating the CCSDS quantization process. The total loss of this stage is the weighted sum of three losses, the rate loss due to the quantizer, the reconstruction loss, and the smoothness loss, $\boldsymbol{\lambda_{recon}\mathcal{L}_{recon} + \lambda_{RR}\mathcal{L}_{RR} + \lambda_{rate}\mathcal{L}_{rate}}$. The rate loss is defined as, $\boldsymbol{\mathcal{L}_{rate}}(\boldsymbol{Y_q}) = \log_2(\max(\boldsymbol{Y_q}) - \min(\boldsymbol{Y_q}))/(\boldsymbol{2a_z + 1})$.

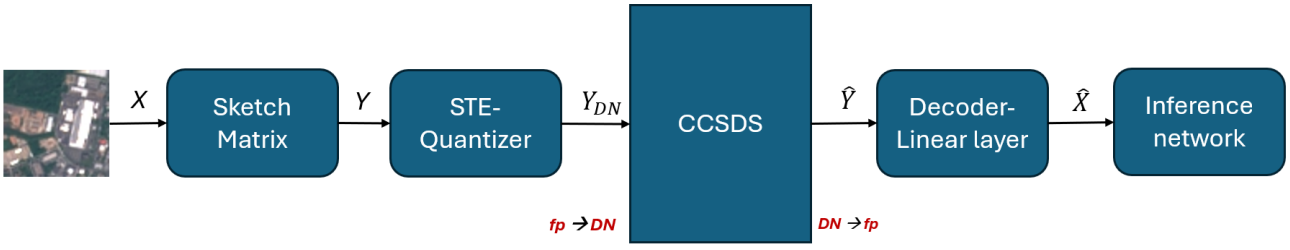


Fig. 1: Proposed sketching architecture with the CCSDS standard.

In the third stage the sketch matrix, STE-Quantizer is frozen and the inference network is learned on $\widehat{\boldsymbol{X}}$. Here the task loss, $\boldsymbol{\mathcal{L}_{Task}}(\boldsymbol{c}, \hat{\boldsymbol{c}})$, $\boldsymbol{c}$ and $\hat{\boldsymbol{c}}$ denoting the true class and the predicted class respectively, is the standard cross-entropy loss. Thus, the sketch matrix is not trained with respect to the task. This is because, we would like to use the CCSDS to protect the sketched feature which could possibly be task agnostic. For the total loss, $\boldsymbol{\mathcal{L} = \lambda_{recon}\mathcal{L}_{recon} + \lambda_{RR}\mathcal{L}_{RR} +}$

$\boldsymbol{\lambda_{rate}\mathcal{L}_{rate} + \lambda_{ce}\mathcal{L}_{ce}}$, in stage 1, $\boldsymbol{\lambda_{rate}}$ and $\boldsymbol{\lambda_{ce}}$are zero. For stage two, $\boldsymbol{\lambda_{ce} = 0}$, and for stage 3, $\boldsymbol{\lambda_{recon} = \lambda_{RR} = \lambda_{rate} = 0}$. The CCSDS component is only used during testing. During test no uniform is added, $\boldsymbol{Y_{DN} = clip(round(cale.Y + offset), 0, 2^D - 1)}$ is fed to the CCSDS and the CCSDS output is converted back to floating-point as described above.

RESULTS AND DISCUSSIONS

The task vs. rate study is based on a Sentinel-2 multispectral dataset EuroSAT-MS [6]. We note that this is not a raw datacube, it is L1C dataset. We use it to study the trends presented in this work. It is a landuse-landcover (LULC) classification dataset with 10 classes and 13 channels. Each cube is of size 64 x 64 x 13. We use 21600 images for training, 3240 for validation, and 1080 for test. We report results for the test set. Point of note, the networks are setup to study the trends and not to be maximally performant on the dataset. We utilize the CCSDS-123.0-B-2 implementation from [8]. We use the absolute error limit and narrow column-oriented mode of the compressor as we have considered width 1 cubes. The rate is defined in terms of bits per feature, the output of the feature extractor, in case of ResNet50, or sketch measurements, in case of proposed architecture, that is input to the compressor.

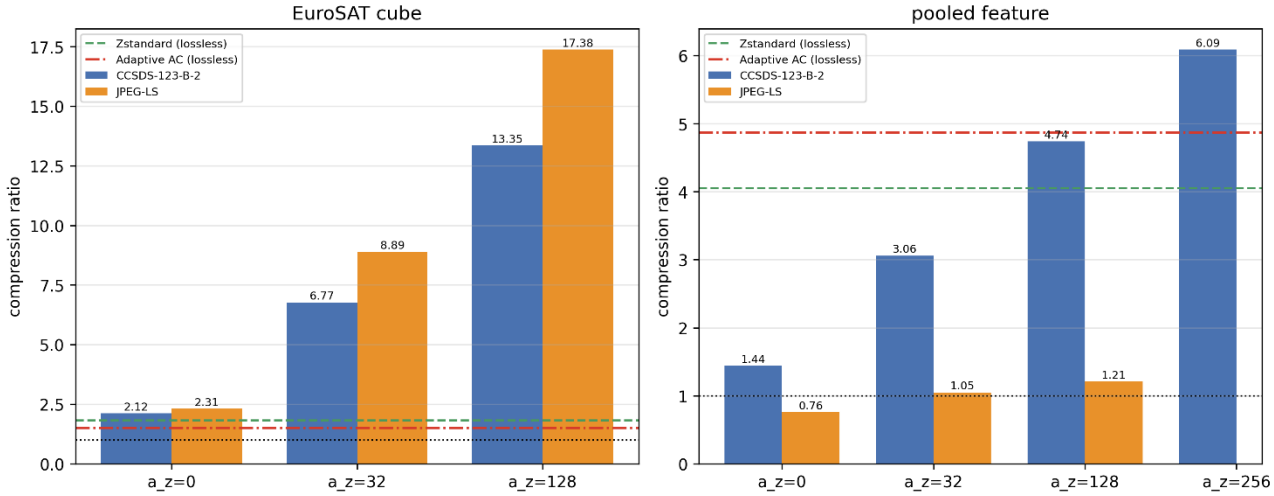


Fig. 2: Compression ratio of CCSDS-123-B-2, JPEG-LS, Zstandard, and adaptive arithmetic coding on the EuroSAT datacubes (left) and pooled ResNet50 features (right), across the near-lossless error bound a_z. Predictive coders win on the structured raw cube; only CCSDS compresses the unstructured pooled features while offering a tunable per-sample error bound. All the methods on the pooled features have the same lossless accuracy of 0.95 (also see Fig. 6).

For our classification model we chose a pretrained ResNet50 model from [7]. To motivate the use of CCSDS as the standard compressor, we compare it against JPEG-LS, Zstandard, and adaptive arithmetic coding (AAC) on two inputs: the EuroSAT datacube and the pooled ResNet50 features by sweeping the near-lossless bound a_z, matched to JPEG-LS's NEAR) where supported (Fig. 2). On the raw cube, the predictive coders CCSDS and JPEG-LS dominate the general-purpose ones, with JPEG-LS slightly ahead, while on the unstructured ResNet50 features the order reverses. JPEG-LS even expands the data, while CCSDS compresses at every operating point, surpassing Zstandard at a_z = 128 and all classical coders at a_z = 256. JPEG-LS limits its bound to NEAR ≤ 255, whereas CCSDS admits arbitrary error limits and reaches operating points JPEG-LS cannot. Among coders that compress features effectively, only CCSDS offers a tuneable near-lossless mode with a deterministic per-sample guarantee, making it the appropriate choice for transmitting sketched measurements. Additionally, all the coders with pooled features have the same lossless, $a_z = 0$, accuracy. CCSDS also maintains task performance across a_z (Fig. 6).

Investigating further the structure of the input to the CCSDS, we study the spectral axis feature smoothness of the input to the standard. As we operate the standard in a narrow column code there are no spatial axis smoothness to exploit. To study the utility of optimizing a feature for smoothness we check whether a spectral axis smoothness actually helps the predictor architecture. We generate a spectrally smooth random matrix by a cumulative sum over the spectral axis and destroy this smoothness by performing a convex combination with parameter α of another random matrix. We estimate the smoothness of the features along the spectral axis by the residual ratio. A lower value of *RR*, specifically less than 1 is considered good.

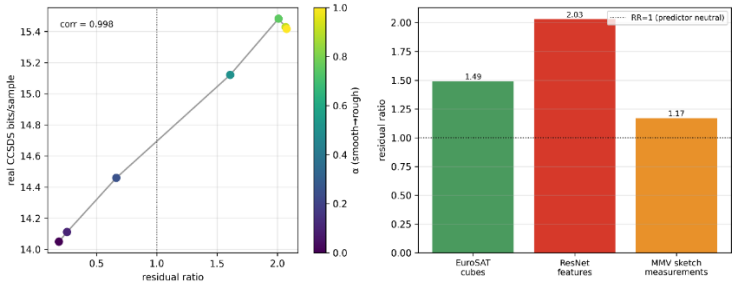


Fig. 3: Spectral axis smoothness. *Left panel*: synthetic experiment. A residual ratio less than 1 leads to smaller no. of bits per sample. *Right Panel*: Residual ratio for the EuroSAT cubes, the ResNet50 features, and the sketch measurements, i.e., Y. Unlike the ResNet50 features that lack spectral smoothness, the sketch measurements preserve or even lower the residual ratio of the datacube.

From Fig. 3, we see that a smaller residual ratio leads to better compression. The designed pattern of smoothness via the sketching measurements can also be seen on the right panel of Fig. 3. The sketch measurements are not only able to preserve the residual ratio of the EuroSAT datacube but also reduce it. The next component that needs to be verified is the STE-Quantizer design. We record the bit rate of the surrogate rate during training and their corresponding true rate from the CCSDS compressor at test and compare them for different a_z. Figure 4 plots the real versus surrogate bitrate. Even though the STE-Quantizer is a simplified open loop, no-residual, and employs a uniform quantizer, it still follows the general shape of the actual CCSDS bitstream. As the surrogate does not use features of the CCSDS quantizer, operating on residuals and having data dependent error distribution, it constantly overestimates the rate and this was the exact STE-Quantizer design requirement to provide worst-case bitrate performance.

Next, we see the evolution of residual ratio across stages in epochs. In Fig. 5 we see that in stage 1, the residual ratio decreases initially and then plateaus. In stage 2, the residual ratio initially increases and then starts to decrease towards another plateau. Finally, we compare [2] and the CCSDS standard with respect to the ResNet50 features. In [2] the ResNet50 backbone and classification head are split and connected by a factorized scale hyperprior entropy encoder and an arithmetic coder [9]. For proper comparison we also feed the same feature generated by the ResNet50 backbone to the CCSDS with different values of a_z. The result is shown in Fig. 6. We see that both compressors are able to maintain task performance, accuracy, across different bit rates. The network trained specifically for the task, Singh et al. [2], has an order of magnitude advantage over the lossless compression, a_z = 0, of the CCSDS standard.

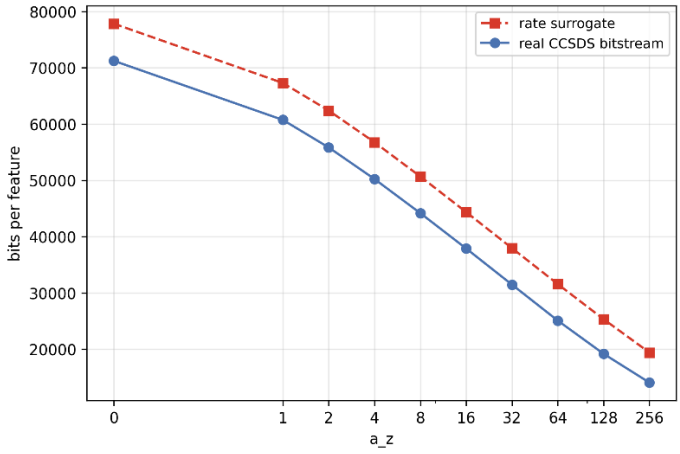


Fig. 4: Surrogate versus real CCSDS bitrate comparison across a_z.

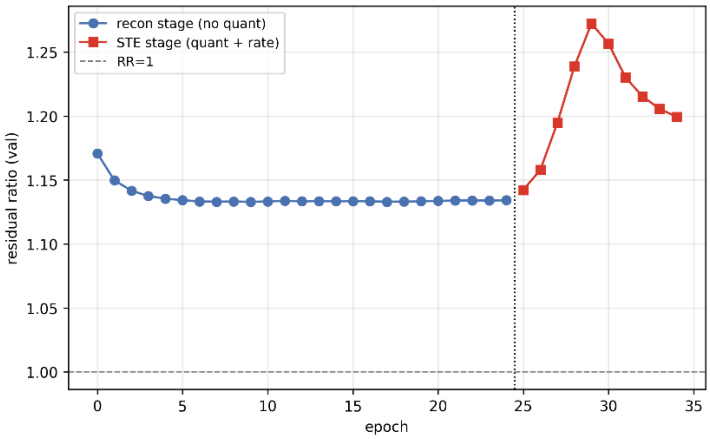


Fig. 5: Evolution of residual ratios across training stages.

For the sketching architecture discussed, two different sketching matrices were used, first a Gaussian random matrix and secondly a learned matrix that is implemented as a linear layer. A linear decoder layer is used to estimate the cube from the quantized sketch measurements. The value of a_z during training of stage 2 was fixed at 4. The choice of different weighting parameters was $\lambda_{recon} = 0.1, \lambda_{RR} = 0.1, \lambda_{rate} = 0.001, and\ \lambda_{ce} = 1$. For this experiment the sketch matrices with $m = 512$ are tested on 200 images, while $m = 256$ and $m = 128$ were tested on 500 and 1000 images respectively. This stratification is only for better usage of compute resources. Though none of the sketches reach the compression performance of [2] a no. of conclusions can be drawn. The learned sketch performs better than the random Gaussian sketch. With increasing a_z, a cliff effect is seen for both the Gaussian and learned sketch measurements that is also visible in the performance of the Singh et al. architecture at lower λ. At m = 128, the no. of elements in the sketch measurements is almost equal to the no. of elements in the 2048-d ResNet50 feature. At this m, the bitrate matches the pooled CCSDS but we see a loss of accuracy on the test set. The validation accuracy of the learned sketch for this m is 0.92, but as we have reported the test accuracy of the other models, we state we test accuracy for the plot. Further, a complete fine-tuning of the end-to-end network, including the classifier, was performed but the performance did not exceed learned sketch of $m = 256$.

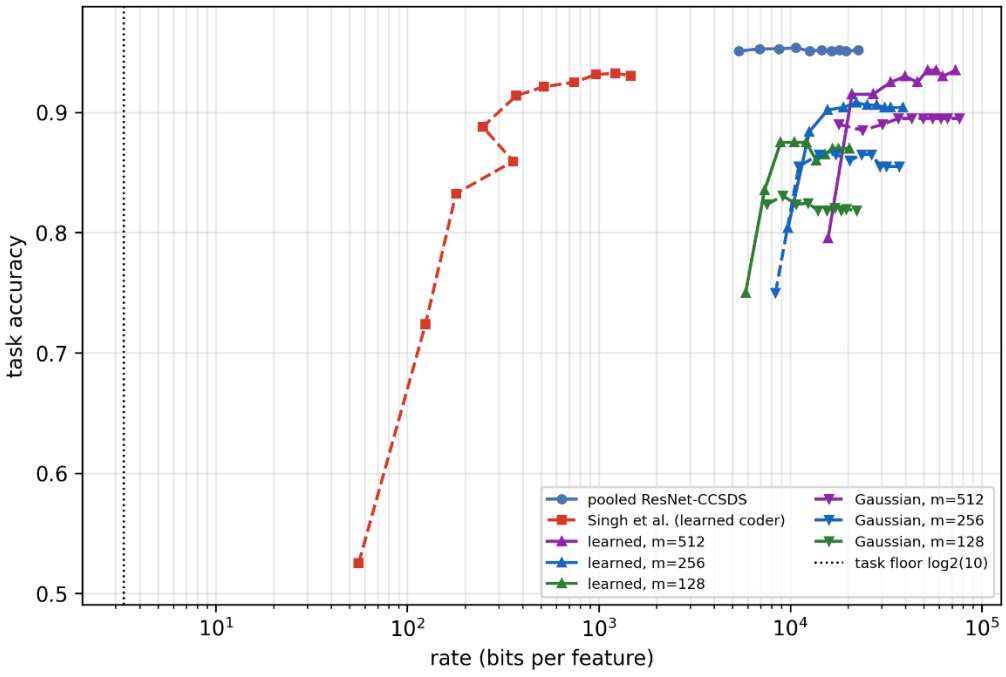


Fig. 6: Accuracy vs. Rate trade-off of different models for ResNet50 features on EuroSAT-MS dataset. In the Singh et al. the λ in the R-D trade-off goes from larger to smaller values as we move from right to left. For the CCSDS networks, we increase a_z as we move from right to left.

CONCLUSION

In this work we present a first study of neural feature compression using the CCSDS-123.0-B-2 standard as a task-agnostic entropy coder, one of the few standards that admits arbitrary user-defined error limits, up to the bit depth of the input. We cast feature compression as a multiple-measurement-vector (MMV) reconstruction problem and study sketching matrices intended to preserve the spectral-axis smoothness that the CCSDS predictor exploits. We establish that smoothness along the spectral axis is an important condition for effective compression by the standard, and we verify through a controlled experiment that the residual ratio predicts the real CCSDS bitrate. While the learned and Gaussian sketches do reduce the residual ratio of the measurements relative to the raw ResNet50 features, they do not render them smooth enough to be competitive with end-to-end feature-compression architectures that employ task-specific, learned entropy coders. In particular, the residual ratio of the sketched measurements never falls below one, which we identify as a likely cause of their limited rate–accuracy performance. Improving this will require a better weighting of the stage-wise losses, specifically retaining smoothness through the quantization stage and, better sketch strategies that perform structured sampling of the dataset distribution [10].

References

[1] ESA Sentinel-2 operations. *https://www.esa.int/Enabling_Support/Operations/Sentinel-2_operations*.
[2] S. Singh, S. Abu-El-Haija, N. Johnston, J. Ballé, A. Shrivastava and G. Toderici, "End-to-End Learning of Compressible Features," *2020 IEEE Int. Conf. on Image Proc. (ICIP)*, 2020.
[3] F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte and L. Van Gool, "Practical Full Resolution Learned Lossless Image Compression," *2019 IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019.
[4] M. Hernández-Cabronero *et al*., "The CCSDS 123.0-B-2 "Low-Complexity Lossless and Near-Lossless Multispectral and Hyperspectral Image Compression" Standard: A comprehensive review," in *IEEE Geosci. Remote Sens. Mag.*, vol. 9, no. 4, pp. 102-119, Dec. 2021.
[5] E. Zisselman, A. Adler and M. Elad. "Compressed Learning for Image Classification: A Deep Neural Network Approach.", *In Handbook of Numerical Analysis* (Vol. 19, pp. 3–17). Elsevier. (2018).
[6] P. Helber, B. Bischke, A. Dengel and D. Borth, "EuroSAT: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification," in *IEEE J. Sel. Topics in Appl. Earth Obs. and Remote Sens*., vol. 12, no. 7, pp. 2217-2226, July 2019.
[7] Y. Wang, C. M. Albrecht and X. X. Zhu, "Multilabel-Guided Soft Contrastive Learning for Efficient Earth Observation Pretraining," in *IEEE Trans. Geosci. Remote Sens.*, vol. 62, pp. 1-16, Oct. 2024.
[8] D. Vorhaug, S. Boyle and M. Orlandić, "High-Level CCSDS 123.0-B-2 Hyperspectral Image Compressor Verification Model," *2024 14th Workshop on Hyperspectral Imag. Sig. Proc.: Evolution in Remote Sens. (WHISPERS)*, 2024,
[9] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, N. Johnston, "Variantional image compression with a scale hyperpior" in *Int. Conf. Learning Representations (ICLR)*, 2018.
[10] J. Sauder, M. Genzel and P. Jung, "Gradient-Based Learning of Discrete Structured Measurement Operators for Signal Recovery," in *IEEE Journal Sel. Areas Inf. Theory*, vol. 3, no. 3, pp. 481-492, Sept. 2022.