Benchmarking neural lossless compression algorithms on multi-purpose astronomical image data

Anonymous Author(s)

Abstract

The site conditions that make astronomical observatories in space and on the ground so desirable—cold and dark—demand a physical remoteness that leads to limited data transmission capabilities. Such transmission limitations directly bottleneck the amount of data that can be acquired. Thus, improving data compression capabilities, which then allows for more data to be obtained, can directly benefit the scientific impact of observatories. Traditional methods for compressing astrophysical data are manually designed. Neural data compression, on the other hand, holds the promise of learning compression algorithms end-to-end from data while leveraging the spatial, temporal, and wavelength structures of astronomical images. This paper introduces AstroCompress (`https://huggingface.co/AnonAstroData`): a neural compression challenge for astrophysics data, featuring four new datasets (and one legacy dataset) with 16-bit unsigned integer imaging data in various modes: space-based, ground-based, multi-wavelength, and time-series imaging. We provide code for easily accessing the data and benchmark seven compression methods (three neural and four non-neural, including all practical state-of-the-art algorithms)). Our results indicate that neural compression techniques can enhance data collection at observatories, and provide guidance on the adoption of neural compression in scientific applications.

1 Introduction

Machine learning is having a growing impact on the natural sciences [Carleo et al., 2019, Wang et al., 2023]. One of the primary hurdles of data-driven scientific discovery is bandwidth bottlenecks in data collection and transmission, particularly if the data is collected autonomously in high-throughput scientific instruments like telescopes or biological sequencing devices. This induces the need to develop novel compression methods for scientific domains.

Data compression algorithms – or *codecs* – based on machine learning have recently surpassed traditional codecs on visual media, such as images and videos [Yang et al., 2023]. These applications often prioritize fast decoding speed at the expense of slower encoding [Yang and Mandt, 2023a]. By contrast, scientific applications generally involve high-frequency data collection, which need to be encoded and transmitted rapidly; the decoding cost is less of a concern, as the downstream data analysis can be done on powerful supercomputers over days or weeks. Moreover, scientific data exibit unique statistical patterns and signal-to-noise distributions. These differences present novel challenges and opportunities for neural compression.

This work explores the potential of neural compression in the astronomy domain. Astronomical imaging captures the locations, spatial extent, temporal changes, and colors of celestial objects and events in 2D arrays. Repeated integrations enhance depth (higher signal-to-noise) through post-processing co-addition and can be used to measure time variations in celestial events. Color information is obtained from co-spatial observations using different wavelength filters.

Preprint. Under review.

As detector sizes grow and costs per pixel decline, larger optical and infrared arrays are being deployed. The need to transmit large amounts of data efficiently is a worsening and already critical challenge for premier astronomical facilities, which often operate in remote locations to optimize observations. This remoteness complicates data transmission to distant computational and archival centers. For ground-based facilities, the inability to transmit raw data as quickly as it is obtained (e.g., via wired internet) means hard copies of the data must be (periodically) moved physically, degrading time-sensitive science [Bezerra et al., 2017]. Space-based facilities risk losing data if it cannot be transmitted with high-enough bandwidth. These constraints are practical and evident in current space missions. The Kepler mission [Borucki et al., 2010] produced $\sim$190 MB of raw data every 6 seconds across the sky, but had to transmit only the averages of many image exposures around certain preselected stars due to bandwidth and storage limits [Jenkins and Dunnuck, 2011]. The TESS satellite [Ricker et al., 2015] undergoes a very similar process. JWST enforces strict data rate limits during observations to stay within deep space downlink constraints, affecting data collection and storage [STSCI, 2024]. In simple terms, transmission bandwidth limitations fundamentally thus constrain data acquisition and influence the design and cost of new facilities for astronomy research. **Today, we essentially are forced to delete huge fractions of space-collected data forever.** Compressing raw imaging data before transmission is highly desirable, balancing better compression ratios against compute and hardware costs, especially for space-based facilities.

In sum, our main contributions are as follows:

- A large ($\sim$320 GB), novel dataset captures a broad range of real astrophysical imaging data, carefully separated into train and test sets, with easy access via HuggingFace `datasets`.
- An extensive comparison of classical and practical neural compression methods on this data, the first publication to our knowledge where neural compression has been systematically studied on astronomical imaging.
- Various qualitative analyses that further our understanding of the bit allocation in astronomical images and that inform future lossy compression codec designs.

2 Background and Related Work

Compression of astronomy images today is largely done by 20+ year old algorithms such as bzip2, Hcompress, or Rice (cf. Pence et al. 2009 and §4.1), though the consultative Committee for Space Data Systems (CCSDS[1]) now recommends JPEG-LS and JPEG-2000 (not yet JPEG-XL). Notably, all of these works rely on manually designing codecs using classical probability distributions and transforms.

In contrast to traditional hand-designed compression algorithms developed for general purposes, the promise of neural compression lies in learning compression codecs end-to-end from specialized datasets [Yang et al., 2023]. By understanding the data distribution, these algorithms can optimally assign code vectors to binary bitstrings, offering significant improvements in both lossy and lossless compression performance (within model limitations).

Neural compression can be categorized into lossy and lossless methods. Lossy compression typically focuses on optimizing a trade-off between bit-rate and reconstruction error (distortion), based on a (variational) autoencoder model and extensions [Theis et al., 2017, Ballé et al., 2017, 2018, Mentzer et al., 2020, Yang and Mandt, 2023b]. For lossless compression, the primary goal is to approximate the density of discrete data. Models used in this approach include normalizing flows [Hoogeboom et al., 2019], diffusion models [Kingma et al., 2021], VAEs [Townsend et al., 2019, Mentzer et al., 2019], and probabilistic circuits [Liu et al., 2022]. Although lossless compression is more straightforward as it does not involve trade-offs like distortion vs. realism [Blau and Michaeli, 2018] or constructing task-specific losses [Dubois et al., 2021, Matsubara et al., 2022], developing a practical codec with very high compression ratio remains a challenge.

A small but growing community has been testing neural compression methods for data from specialized scientific domains. Hayne et al. [2021] published a study on neural compression for image-like turbulence and climate data sets using a lossy neural compression model. Choi et al. [2021] studied similar neural compression models on plasma data. Huang and Hoefler [2023] compress climate data

[1] `https://public.ccsds.org/default.aspx`.

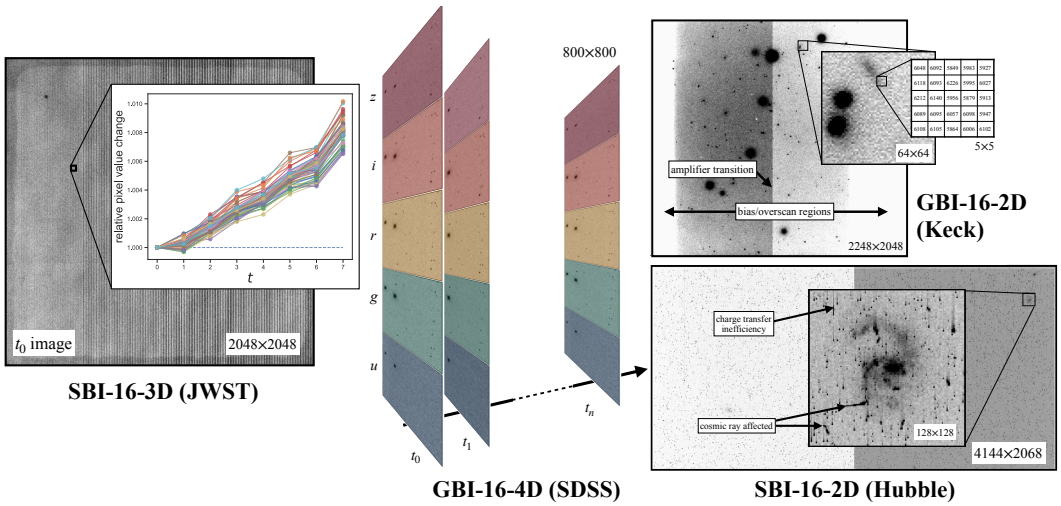

Figure 1: Depiction of salient features in the AstroCompress corpus using representative images from each dataset. Inset to the JWST t_0 (first) image are the value changes in time for a small sample of pixels. In SDSS there are 5 filtered images per observation epoch, up to a variable number n observations in the same portion of the sky. The inset of Hubble zooms in on a spiral galaxy, showing cosmic ray hits (black) and charge transfer inefficiency, causing vertical flux smearing. The actual pixel values in Keck are shown for a zoomed-in 5×5 pix^2 region.

by overfitting a neural network and using the network weights as compressed data representation. Wang et al. [2023] adopted a classical-neural hybrid approach in medical image compression. Overall, we are unaware of any efforts applying neural compression to astronomical images.

While the astronomical data in public archives (e.g., MAST[2]) exist in nearly limitless quantities, the assembly process for a machine learning suitable corpus requires significant domain knowledge. For example, previous attempts at ML-friendly corpus creation, such as Galaxy10 [Leung and Bovy, 2019] (see also Xue et al. 2023 and Khujaev et al. 2023), have generally rescaled data to 8-bit RGB images, destroying nearly all of the dynamic range and thus much of the information useful for novel science.

3 AstroCompress Corpus

Our central contribution is the AstroCompress corpus, curated to capture a broad range of real astrophysical imaging data and presented to enable the exploration of neural compression. The corpus is released on HuggingFace and can be easily accessed using Python, with code examples in the Supplementary Material. The corpus consists of 5 distinct datasets, spanning a variety of observing conditions from space and from Earth, types of detector technology, and large dynamic ranges. The quantity of data is much larger and more varied than previous compression-focused corpora [Pata and Schindler, 2015, Maireles-González et al., 2023] to ensure ample training data for ML-based approaches. Besides the typical 2D imaging data, we also include higher-dimensional (3D and 4D) data cubes containing multiple images of the same spatial origin but along different wavelength and/or temporal dimensions. We briefly describe the five datasets comprising AstroCompress, presenting the some key features in Fig. 1, and defer details of their composition and acquisition to the Supplementary Material:

GBI-16-2D (Keck) is a diverse, 2D optical imaging dataset from the ground-based W. M. Keck Observatory meant to evaluate model generalizability. **SBI-16-2D (Hubble)** is a 2D dataset derived from the Hubble Space Telescope (HST) Advanced Camera for Surveys (ACS; Sirianni et al. 2005). **SBI-16-3D (JWST)** comes from the NIRCAM instrument onboard the James Webb Space Telescope (JWST), and contains time-series image cubes. **GBI-16-4D (SDSS)** is a ground-based dataset assembled from the Sloan Digital Sky Survey (SDSS; York et al. 2000). These are 4D data cubes across both wavelength and time, allowing evaluation of compression on tensors with large correlations.

[2] https://mast.stsci.edu/

GBI-16-2D-Legacy is a small ground-based dataset obtained across many different telescopes, and is reproduced from the public corpus released by Maireles-González et al. [2023]. Our experiments only made use of the subset of data from the Las Cumbres Observatory (LCO).

4 Experiments

We establish the compression performance of our selected neural and non-neural compression methods on our proposed datasets. We describe our experiment protocol in Sec. 4.2, and present our main results in Sec. 4.3. Our results show that, even with only minimal architectural adjustments, neural compression can match or even surpass the best classical codecs. Moreover, neural codecs designed for natural image data, such as L3C and PixelCNN, have difficulty exploiting cross-frame correlations in astronomy images, likely due to the image noise characteristics.

To better understand how the behavior of the algorithms depends on data characteristics, we examine bit-rate allocation both qualitatively and quantitatively in Sec. 4.4. Echoing earlier findings from Pence et al. [2009], we see a strong correlation between bit-rate and measures of noise such as SNR and exposure time, confirming that most of the bits are allocated to noisy pixels that can be well modeled by i.i.d. white noise distributions. Lastly, we explore the out-of-domain generalization performance of a neural compression method, IDF, in Sec. E.3.

4.1 Compression Methods

We consider four non-neural methods as baselines, including three standard codecs from the Joint Photographic Experts Group (JPEG) and one codec developed by the Jet Propulsion Laboratory (JPL). The `imagecodecs` library provides the necessary APIs for all methods. Specifically, we run **JPEG-XL**, **JPEG-LS**, **JPEG-2000**, and **RICE** codecs in lossless mode with default settings. Additionally, we run JPEG-XL under the maximum compression ratio mode as an extra reference.

We adopt three well-known neural lossless compression methods in the literature, representing key approaches in deep generative modeling for compression:

Integer Discrete Flows (IDF): a flow-based model extending the concept of normalizing flows [Rezende and Mohamed, 2015] for lossless compression [Hoogeboom et al., 2019]. Unlike conventional normalizing flow models that operate on continuous data, IDF employs discrete bijective mappings using invertible neural networks to connect discrete pixels with a discrete latent.

L3C: a VAE-based lossless image compression method utilizing a two-part coding scheme [Mentzer et al., 2019]. It involves training a hierarchical VAE with discrete latent representations to capture high-level information about the input image.

PixelCNN++: an autoregressive model using masked convolutions to model the distribution of each pixel given previous pixels in a raster scan order [Salimans et al., 2017]. PixelCNN++ naturally allows for lossless compression using autoregressive entropy coding [Mentzer et al., 2019].

4.2 Experiment setup

We experiment on two categories of data: single-frame images and correlated-frame images with additional temporal or wavelength information. We consider the LCO, Keck, and Hubble datasets as single-frame image datasets. For the JWST datasets, we select the first time step of each 3D image cube and form a single-frame dataset, called JWST-2D, and the residual images between time steps as a separate, correlated-frame dataset, called JWST-2D-Res. In principle, entire JWST 3D arrays could be compressed by encoding the initial frame and subsequent residual frames separately. We sub-select three benchmark datasets from SDSS as follows: (1) the first time step of the r filter band forms a single-frame dataset, called SDSS-2D; (2) the first time step of the g, r, and i filter bands constitute a correlated-frame dataset, called SDSS-3Dλ; and (3) the first three time steps of the r filter band create a correlated-frame dataset, called SDSS-3DT.

4.3 Compression Performance

The top subsection of Table 1 presents compression ratios on single-frame compression experiments. PixelCNN++ frequently achieves the best performance, except on the LCO dataset [Maireles-

Experiment	Neural Codec			Non-neural Codec				
	IDF	L3C	PixelCNN++	JPEG-XL (max)	JPEG-XL	JPEG-LS	JPEG-2000	RICE
LCO	2.83	1.67	1.41	**2.98**	2.78	2.81	2.80	2.65
Keck	2.04	1.89	**2.08**	2.01	1.97	1.97	1.96	1.84
Hubble	2.94	2.90	3.13	**3.26**	2.92	2.86	2.67	2.64
JWST-2D	**1.44**	1.38	**1.44**	1.38	1.33	1.35	1.37	1.24
SDSS-2D	2.91	2.36	3.35	**3.38**	3.14	3.16	3.20	2.96
JWST-2D-Res	3.14	2.91	2.80	**3.35**	2.37	3.24	1.69	3.08
SDSS-3Dλ	3.05	2.29	2.88	**3.49**	3.23	3.24	3.28	3.05
SDSS-3DT	3.03	2.59	3.02	**3.48**	3.23	3.24	3.29	3.05

Table 1: Compression ratios for all methods across experiments, with bold text indicating the best performance and underlined text indicating the second best. The top and bottom subsections of the table contain single-frame and correlated-frame compression results, respectively.

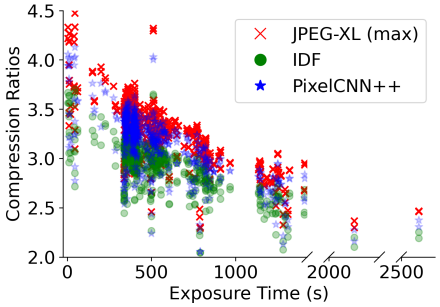

Figure 2: Hubble exposure times plotted against compression ratios using various algorithms. Longer exposure times tend to induce more incompressible noise and, hence, reduce compression ratios.

González et al., 2022] where it overfits on the small amount of data—whereas IDF thrives. PixelCNN++ often closely matches JPEG-XL (max) with less than a 4% difference and occasionally outperforms JPEG-XL with $\sim$ 4-10% improvements. IDF also demonstrates competitive performance on many datasets, despite being non-autoregressive. However, JPEG-XL (max) consistently exhibits high compression ratios across datasets, indicating that there is still potential for improvement in future neural codecs. Notably, the highest compression ratio previously achieved for the LCO dataset was 2.79 by Maireles-González et al. [2023], which is surpassed by JPEG-XL (max) in our benchmark. Our results for non-neural codecs, including JPEG-LS, JPEG-2000, and RICE, are consistent with the previously published benchmarks on this dataset.

Interestingly, non-neural codecs show superior performance boosts on our correlated-frame data, as seen in the bottom subsection of Table 1. Neural codecs seem to struggle with capturing correlations across different frames. This indicates a need for further improvement in neural compression techniques to better extract cross-wavelength or cross-timestep information.

4.4 The Effect of Noise

Following Pence et al. [2009], our Figure 2 illustrates an inverse relationship between compression ratios and exposure time, which is one of the key variables in determining the signal-to-noise ratio (SNR) in an image. We demonstrate a negative correlation between these two variables, likely because an increase in exposure time results in an increased number of noise bits. We hypothesize that the plateau seen will reverse at higher exposure times as many CCD pixels reach their max physical value, reducing image entropy.

Figure 3 demonstrates that source (star/galaxy) pixels have higher bitrates, as expected—in a sense, these pixels are more "surprising," and thus a lower likelihood is assigned. Interestingly, the background pixel regions of the bitrate heatmap are significantly more noisy than the corresponding SNR background. This suggests that there may be potential for reduction in the bitrate of the higher bitrate background pixels, as we might expect most background pixels to exhaust a similar number of bits. The details of source pixel detection, background estimation and more are given in the appendix.

Figure 4 furthermore demonstrates the extreme bit-rate consumption by background noise pixels. On this image, 98.5% of pixels fell under 3 SNR. Some source pixels were placed in the lowest SNR

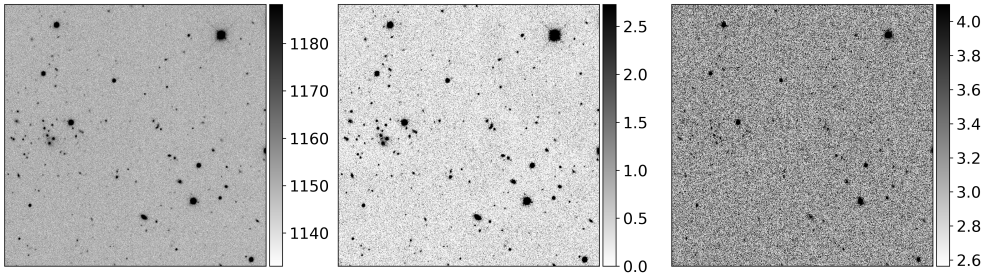

Figure 3: From left to right, an example SDSS-2D image: raw image, SNR heatmap, and PixelCNN++ bitrate heatmap. Colors are z-score normalized for visualization; colorbars indicate true values.

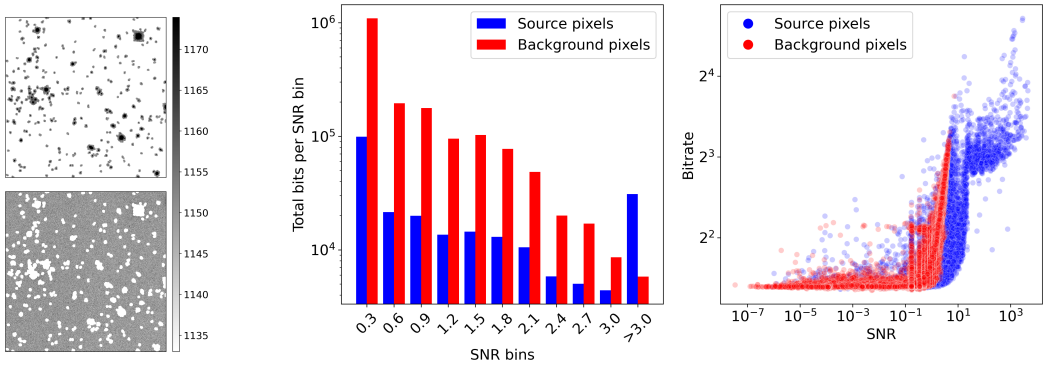

Figure 4: **Left**: SNR heatmap of an example image (Figure 3), showing source (top) vs. background (bottom) pixels. **Middle**: histogram of PixelCNN++ total bit allocation for various binned SNR values. **Right**: scatterplot showing the positive correlation between SNR and PixelCNN++ bitrate.

bin, but this is likely due to some overestimation of source radii in the source masking process. The scatterplot is interestingly almost a step function, with a jump from $\texttt{Bitrate} \approx 3$ to $\texttt{Bitrate} \approx 10$ at $\texttt{SNR} \approx 10^1$—the transition point at which stars and galaxies emerge over background noise.

5 Conclusion

AstroCompress aims to incentivize the development of astronomy-tuned neural codecs for eventual real-world deployment, by providing datasets that are representative of real use cases. While this work focuses on neural compression, many other machine learning frameworks are intimately related. Compression is "bijectively" linked with likelihood estimation by Shannon's source coding theorem [Shannon, 1948]. We suggest the use of our dataset in other machine learning for astronomy contexts, such as self-supervised learning for foundation models, semantic search and anomaly detection.

The exploration in AstroCompress will likely transfer to other kinds of high-resolution, high-bit depth imagery, such as radio astronomy, satellite and biological imaging. The Square Kilometer Array is expected to collect 62 exabytes per year [Farnes et al., 2018] and improvement in lossless compression could translate to a meaningful reduction in storage costs. Current efforts[3] in mapping the mouse hippocampus are estimated to contain 25 petabytes of imaging data. Extreme demand for data compression will soon extend to other data types as well, such as human genomics data[4], which is expected to generate 2–10 exabytes of data over the next decade. We hope that this work will kick off further exploration of neural compression models for science and the curation of new compression datasets across scientific fields.

[3] https://sites.research.google/neural-mapping/

[4] https://www.genome.gov/about-genomics/fact-sheets/Genomic-Data-Science

Acknowledgments and Disclosure of Funding

This research is based in part on observations made with the NASA/ESA Hubble Space Telescope obtained from the Space Telescope Science Institute, which is operated by the Association of Universities for Research in Astronomy, Inc., under NASA contract NAS 5–26555. HST data are released under the Creative Commons Attribution 4.0 International license. This work is also based in part on observations made with the NASA/ESA/CSA James Webb Space Telescope. The data were obtained from the Mikulski Archive for Space Telescopes at the Space Telescope Science Institute, which is operated by the Association of Universities for Research in Astronomy, Inc., under NASA contract NAS 5-03127 for JWST. This research has made use of the Keck Observatory Archive (KOA), which is operated by the W.M. Keck Observatory and the NASA Exoplanet Science Institute (NExScI), under contract with the National Aeronautics and Space Administration. Funding for SDSS-III was provided by the Alfred P. Sloan Foundation, the Participating Institutions, the National Science Foundation, and the U.S. Department of Energy Office of Science. The SDSS-III website is http://www.sdss3.org/. All SDSS data used herein are considered in the public domain. Data in GBI-16-2D-Legacy, curated and released to the public in FITS format by Maireles-González et al. [2023], makes use of observations the Isaac Newton Group of Telescopes, from Las Cumbres Observatory global telescope network and from The Joan Oró Telescope (TJO). The authors also acknowledge support from the National Science Foundation (NSF) under an NSF CAREER Award (2047418), award numbers 2003237 and 2007719, by the Department of Energy under grant DE-SC0022331, and gifts from Qualcomm. J.S.B. and R.S. are partially supported by the National Science Foundation Grant #2206744.

References

Giuseppe Carleo, Ignacio Cirac, Kyle Cranmer, Laurent Daudet, Maria Schuld, Naftali Tishby, Leslie Vogt-Maranto, and Lenka Zdeborová. Machine learning and the physical sciences. *Rev. Mod. Phys.*, 91:045002, Dec 2019. doi: 10.1103/RevModPhys.91.045002. URL https://link.aps.org/doi/10.1103/RevModPhys.91.045002.

Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, Anima Anandkumar, Karianne Bergen, Carla P. Gomes, Shirley Ho, Pushmeet Kohli, Joan Lasenby, Jure Leskovec, Tie-Yan Liu, Arjun Manrai, Debora Marks, Bharath Ramsundar, Le Song, Jimeng Sun, Jian Tang, Petar Veličković, Max Welling, Linfeng Zhang, Connor W. Coley, Yoshua Bengio, and Marinka Zitnik. Scientific discovery in the age of artificial intelligence. *Nature*, 620(7972):47–60, August 2023. doi: 10.1038/s41586-023-06221-2.

Yibo Yang, Stephan Mandt, and Lucas Theis. An introduction to neural data compression. *Foundations and Trends® in Computer Graphics and Vision*, 15(2):113–200, 2023. ISSN 1572-2740. doi: 10.1561/0600000107. URL http://dx.doi.org/10.1561/0600000107.

Yibo Yang and Stephan Mandt. Computationally-efficient neural image compression with shallow decoders. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 530–540, 2023a.

Jeronimo Bezerra, Vinicius Arcanjo, Julio Ibarra, Jeff Kantor, Ron Lambert, Matt Kollross, Albert Astudillo, Shahram Sobhani, Sandra Jaque, Donald Petravick, Heidi Morgan, and Luiz Lopez. International networking in support of extremely large astronomical data-centric operations. In *Astronomical Data Analysis Software and Systems (ADASS XXVII) conference*, Santiago, Chile, 10/2017 2017.

William J. Borucki, David Koch, Gibor Basri, Natalie Batalha, Timothy Brown, Douglas Caldwell, John Caldwell, Jørgen Christensen-Dalsgaard, William D. Cochran, Edna DeVore, Edward W. Dunham, Andrea K. Dupree, Thomas N. Gautier, John C. Geary, Ronald Gilliland, Alan Gould, Steve B. Howell, Jon M. Jenkins, Yoji Kondo, David W. Latham, Geoffrey W. Marcy, Søren Meibom, Hans Kjeldsen, Jack J. Lissauer, David G. Monet, David Morrison, Dimitar Sasselov, Jill Tarter, Alan Boss, Don Brownlee, Toby Owen, Derek Buzasi, David Charbonneau, Laurance Doyle, Jonathan Fortney, Eric B. Ford, Matthew J. Holman, Sara Seager, Jason H. Steffen, William F. Welsh, Jason Rowe, Howard Anderson, Lars Buchhave, David Ciardi, Lucianne

Walkowicz, William Sherry, Elliott Horch, Howard Isaacson, Mark E. Everett, Debra Fischer, Guillermo Torres, John Asher Johnson, Michael Endl, Phillip MacQueen, Stephen T. Bryson, Jessie Dotson, Michael Haas, Jeffrey Kolodziejczak, Jeffrey Van Cleve, Hema Chandrasekaran, Joseph D. Twicken, Elisa V. Quintana, Bruce D. Clarke, Christopher Allen, Jie Li, Haley Wu, Peter Tenenbaum, Ekaterina Verner, Frederick Bruhweiler, Jason Barnes, and Andrej Prsa. Kepler Planet-Detection Mission: Introduction and First Results. *Science*, 327(5968):977, February 2010. doi: 10.1126/science.1185402.

Jon M. Jenkins and Jeb Dunnuck. The little photometer that could: technical challenges and science results from the Kepler Mission. In Howard A. MacEwen and James B. Breckinridge, editors, *UV/Optical/IR Space Telescopes and Instruments: Innovative Technologies and Concepts V*, volume 8146, page 814602. International Society for Optics and Photonics, SPIE, 2011. doi: 10.1117/12.897767. URL https://doi.org/10.1117/12.897767.

George R. Ricker, Joshua N. Winn, Roland Vanderspek, David W. Latham, Gáspár Á. Bakos, Jacob L. Bean, Zachory K. Berta-Thompson, Timothy M. Brown, Lars Buchhave, Nathaniel R. Butler, R. Paul Butler, William J. Chaplin, David Charbonneau, Jørgen Christensen-Dalsgaard, Mark Clampin, Drake Deming, John Doty, Nathan De Lee, Courtney Dressing, Edward W. Dunham, Michael Endl, Francois Fressin, Jian Ge, Thomas Henning, Matthew J. Holman, Andrew W. Howard, Shigeru Ida, Jon M. Jenkins, Garrett Jernigan, John Asher Johnson, Lisa Kaltenegger, Nobuyuki Kawai, Hans Kjeldsen, Gregory Laughlin, Alan M. Levine, Douglas Lin, Jack J. Lissauer, Phillip MacQueen, Geoffrey Marcy, Peter R. McCullough, Timothy D. Morton, Norio Narita, Martin Paegert, Enric Palle, Francesco Pepe, Joshua Pepper, Andreas Quirrenbach, Stephen A. Rinehart, Dimitar Sasselov, Bun'ei Sato, Sara Seager, Alessandro Sozzetti, Keivan G. Stassun, Peter Sullivan, Andrew Szentgyorgyi, Guillermo Torres, Stephane Udry, and Joel Villasenor. Transiting Exoplanet Survey Satellite (TESS). *Journal of Astronomical Telescopes, Instruments, and Systems*, 1:014003, January 2015. doi: 10.1117/1.JATIS.1.1.014003.

STSCI. JWST Data Volume and Data Excess, 2024. https://jwst-docs.stsci.edu/jwst-general-support/jwst-data-volume-and-data-excess [Accessed: May 30, 2024].

W. D. Pence, R. Seaman, and R. L. White. Lossless Astronomical Image Compression and the Effects of Noise. *PASP*, 121(878):414, April 2009. doi: 10.1086/599023.

Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. *International Conference on Learning Representations*, 2017.

Johannes Ballé, Valero Laparra, and Eero P Simoncelli. End-to-end optimized image compression. *International Conference on Learning Representations*, 2017.

Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. *International Conference on Learning Representations*, 2018.

Fabian Mentzer, George D Toderici, Michael Tschannen, and Eirikur Agustsson. High-fidelity generative image compression. *Advances in Neural Information Processing Systems*, 33:11913–11924, 2020.

Ruihan Yang and Stephan Mandt. Lossy image compression with conditional diffusion models. *Advances in Neural Information Processing Systems*, 36, 2023b.

Emiel Hoogeboom, Jorn Peters, Rianne Van Den Berg, and Max Welling. Integer discrete flows and lossless compression. *Advances in Neural Information Processing Systems*, 32, 2019.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

James Townsend, Tom Bird, and David Barber. Practical lossless compression with latent variables using bits back coding. *arXiv preprint arXiv:1901.04866*, 2019.

Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, and Luc Van Gool. Practical full resolution learned lossless image compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10629–10638, 2019.

Anji Liu, Stephan Mandt, and Guy Van den Broeck. Lossless compression with probabilistic circuits. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6228–6237, 2018.

Yann Dubois, Benjamin Bloem-Reddy, Karen Ullrich, and Chris J Maddison. Lossy compression for lossless prediction. *Advances in Neural Information Processing Systems*, 34:14014–14028, 2021.

Yoshitomo Matsubara, Ruihan Yang, Marco Levorato, and Stephan Mandt. Supervised compression for resource-constrained edge computing systems. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2685–2695, 2022.

Lucas Hayne, John Clyne, and Shaomeng Li. Using neural networks for two dimensional scientific data compression. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 2956–2965. IEEE, 2021.

Jong Choi, Michael Churchill, Qian Gong, Seung-Hoe Ku, Jaemoon Lee, Anand Rangarajan, Sanjay Ranka, Dave Pugmire, CS Chang, and Scott Klasky. Neural data compression for physics plasma simulation. In *Neural Compression: From Information Theory to Applications–Workshop@ ICLR 2021*, 2021.

Langwen Huang and Torsten Hoefler. Compressing multidimensional weather and climate data into neural networks. In *The Eleventh International Conference on Learning Representations*, 2023.

Kai Wang, Yuanchao Bai, Deming Zhai, Daxin Li, Junjun Jiang, and Xianming Liu. Learning lossless compression for high bit-depth medical imaging. In *2023 IEEE International conference on multimedia and expo (ICME)*, pages 2549–2554. IEEE, 2023.

Henry W. Leung and Jo Bovy. Deep learning of multi-element abundances from high-resolution spectroscopic data. *MNRAS*, 483(3):3255–3277, March 2019. doi: 10.1093/mnras/sty3217.

Zhiwei Xue, Yuhang Li, Yash Patel, and Jeffrey Regier. Diffusion Models for Probabilistic Deconvolution of Galaxy Images. *arXiv e-prints*, art. arXiv:2307.11122, July 2023. doi: 10.48550/arXiv.2307.11122.

Nodirkhuja Khujaev, Roman Tsoy, and Seungryul Baek. AstroYOLO: Learning Astronomy Multi-tasks in a Single Unified Real-Time Framework. In *Machine Learning and the Physical Sciences Workshop, NeurIPS*, 2023.

Petr Pata and Jaromir Schindler. Astronomical context coder for image compression. *Experimental Astronomy*, 39:495–512, 2015.

Òscar Maireles-González, Joan Bartrina-Rapesta, Miguel Hernández-Cabronero, and Joan Serra-Sagristà. Efficient lossless compression of integer astronomical data. *Publications of the Astronomical Society of the Pacific*, 135(1051):094502, 2023.

M. Sirianni, M. J. Jee, N. Benítez, J. P. Blakeslee, A. R. Martel, G. Meurer, M. Clampin, G. De Marchi, H. C. Ford, R. Gilliland, G. F. Hartig, G. D. Illingworth, J. Mack, and W. J. McCann. The Photometric Performance and Calibration of the Hubble Space Telescope Advanced Camera for Surveys. *PASP*, 117(836):1049–1112, October 2005. doi: 10.1086/444553.

Donald G. York et al. The Sloan Digital Sky Survey: Technical Summary. *AJ*, 120(3):1579–1587, September 2000. doi: 10.1086/301513.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications, 2017. URL https://arxiv.org/abs/1701.05517.

Òscar Maireles-González, Joan Bartrina-Rapesta, Miguel Hernández-Cabronero, and Joan Serra-Sagristà. Analysis of lossless compressors applied to integer and floating-point astronomical data. In *2022 Data Compression Conference (DCC)*, pages 389–398. IEEE, 2022.

Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

Jamie Farnes, Ben Mort, Fred Dulwich, Stef Salvini, and Wes Armour. Science Pipelines for the Square Kilometre Array. *Galaxies*, 6(4):120, November 2018. doi: 10.3390/galaxies6040120.

Md Abul Hayat, George Stein, Peter Harrington, Zarija Lukić, and Mustafa Mustafa. Self-supervised Representation Learning for Astronomical Images. *ApJ Letters*, 911(2):L33, April 2021. doi: 10.3847/2041-8213/abf2c7.

J. B. Oke, J. G. Cohen, M. Carr, J. Cromer, A. Dingizian, F. H. Harris, S. Labrecque, R. Lucinio, W. Schaal, H. Epps, and J. Miller. The Keck Low-Resolution Imaging Spectrometer. *PASP*, 107: 375, April 1995. doi: 10.1086/133562.

W. D. Pence, L. Chiappetti, C. G. Page, R. A. Shaw, and E. Stobie. Definition of the Flexible Image Transport System (FITS), version 3.0. *AAP*, 524:A42, December 2010. doi: 10.1051/0004-6361/201015362.

C. Rockosi et al. The low-resolution imaging spectrograph red channel CCD upgrade: fully depleted, high-resistivity CCDs for Keck. In Ian S. McLean, Suzanne K. Ramsay, and Hideki Takami, editors, *Ground-based and Airborne Instrumentation for Astronomy III*, volume 7735 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, page 77350R, July 2010. doi: 10.1117/12.856818.

Masao Sako, Bruce Bassett, Andrew C. Becker, Peter J. Brown, Heather Campbell, Rachel Wolf, David Cinabro, Chris B. D'Andrea, Kyle S. Dawson, Fritz DeJongh, Darren L. Depoy, Ben Dilday, Mamoru Doi, Alexei V. Filippenko, John A. Fischer, Ryan J. Foley, Joshua A. Frieman, Lluis Galbany, Peter M. Garnavich, Ariel Goobar, Ravi R. Gupta, Gary J. Hill, Brian T. Hayden, Renée Hlozek, Jon A. Holtzman, Ulrich Hopp, Saurabh W. Jha, Richard Kessler, Wolfram Kollatschny, Giorgos Leloudas, John Marriner, Jennifer L. Marshall, Ramon Miquel, Tomoki Morokuma, Jennifer Mosher, Robert C. Nichol, Jakob Nordin, Matthew D. Olmstead, Linda Östman, Jose L. Prieto, Michael Richmond, Roger W. Romani, Jesper Sollerman, Max Stritzinger, Donald P. Schneider, Mathew Smith, J. Craig Wheeler, Naoki Yasuda, and Chen Zheng. The Data Release of the Sloan Digital Sky Survey-II Supernova Survey. *PASP*, 130(988):064002, June 2018. doi: 10.1088/1538-3873/aab4e0.

Chris Stoughton et al. Sloan Digital Sky Survey: Early Data Release. *Astronomical Journal*, 123(1): 485–548, January 2002. doi: 10.1086/324741.

Hannah Gulick, Jessica Lu, Steve Beckwith, Joshua Bloom, and Guy Nir. CubeSat for Rapid IR and Optical Surveys. In *44th COSPAR Scientific Assembly. Held 16-24 July*, volume 44, page 1985, July 2022.

S. M. Kahn, N. Kurita, K. Gilmore, M. Nordby, P. O'Connor, R. Schindler, J. Oliver, R. Van Berg, S. Olivier, V. Riot, P. Antilogus, T. Schalk, M. Huffer, G. Bowden, J. Singal, and M. Foss. Design and development of the 3.2 gigapixel camera for the Large Synoptic Survey Telescope. In Ian S. McLean, Suzanne K. Ramsay, and Hideki Takami, editors, *Ground-based and Airborne Instrumentation for Astronomy III*, volume 7735 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, page 77350J, July 2010. doi: 10.1117/12.857920.

Jonathan P. Gardner et al. The James Webb Space Telescope Mission. *PASP*, 135(1048):068001, June 2023. doi: 10.1088/1538-3873/acd1b5.

Rodger I. Thompson, Marcia Rieke, Glenn Schneider, Dean C. Hines, and Michael R. Corbin. Initial On-Orbit Performance of NICMOS. *APJL*, 492(2):L95–L97, January 1998. doi: 10.1086/311095.

Rob Seaman, William Pence, and Rick White. fpack: FITS Image Compression Program. Astrophysics Source Code Library, record ascl:1010.002, October 2010.

Alberto G Villafranca, Jordi Portell, and Enrique García-Berro. Prediction error coder: a fast lossless compression method for satellite noisy data. *Journal of Applied Remote Sensing*, 7(1): 074593–074593, 2013.

Brian Thomas, Tim Jenness, Frossie Economou, Perry Greenfield, Paul Hirst, David S Berry, Erik Bray, Norman Gray, Demitri Muna, James Turner, et al. Learning from fits: Limitations in use in modern astronomical research. *Astronomy and Computing*, 12:133–145, 2015.

Thulfiqar H Mandeel, Muhammad Imran Ahmad, Noor Aldeen A Khalid, and Mohd Nazrin Md Isa. A comparative study on lossless compression mode in webp, better portable graphics (bpg), and jpeg xl image compression algorithms. In *2021 8th International Conference on Computer and Communication Engineering (ICCCE)*, pages 17–22. IEEE, 2021.

Supplementary Material

A Dataset Supplementals

Telescope	Dataset ID	Location	Central Wavelength (Å)	Bandpass Filter	Resolution (arcsec / pix)	Image Sizes (pix×px)	Total # Arrays	Total Dataset Size (GB)	Approx. Total Pixels
LCO	GBI-16-2D-Legacy	multiple sites Earth	varies	B, V, rp, ip	0.58	3136×2112	9	0.1	5.5×10^7
Keck	GBI-16-2D	HI, USA, Earth	varies, 4500–8400	B, V, R, I	0.135	2248×2048; 3768×2520	137	1.5	7.4×10^8
Hubble	SBI-16-2D	LEO, Space	5926	F606W	0.05	4144×2068	4282	69	3.7×10^{10}
JWST	SBI-16-3D	L2, Space	19840	F200W	0.031	2048×2048 × #Groups	1273	90	7.0×10^9
SDSS	GBI-16-4D	NM, USA, Earth	3543, 4770, 6231, 7625, 9134	all of $[u, g, r, i, z]$	0.396	800×800×5 × #Timesteps	500	158	1.6×10^{10}

Table 2: Datasets summary. Total pixels computed using each 16-bit data point as one "pixel."

A.1 Naming conventions

Throughout this work, the data are referred to in three different ways depending on the context; we elucidate the reference conventions here.

A.1.1 Dataset ID

The adopted Dataset ID naming convention provides a shorthand description of the origin and form of the data:

(*origin*)(*data taking mode*)-(*number of bits per pixel*)-(*dimensionality*),

where *origin* is either space-based (SB) or ground-based (GB), and *data taking mode* refers to the primary objective of the exposure, only I (imaging[5]). -2D implies each instance is a 2-dimensional array. -3D may be thought of as a temporal sequence of images (ie., movie) taken in roughly the same portion of the sky. -4D may be thought of as a temporal sequence of images taken in the same portion of the sky at different wavelengths. Clearly, -3D and -4D may be decomposed into individual 2D images. All data in the corpus are `uint16` but future datasets may be added with different bit depths.

This is a standardized way of naming our datasets that include functional details, allowing one to quickly search for or identify the kind of data from the name alone.

A.1.2 Telescope or Survey Name

For brevity and ease of reading, we use the "telescope" column as nicknames to refer to each of our published datasets throughout our paper.

A.1.3 Experimental dataset names

For several reasons, some of our benchmark models were trained on a subset of a dataset. To this end, we added additional descriptors to the dataset nicknames as a way to describe these subselections. LCO, Keck, and Hubble were unchanged and used fully. JWST was used in two forms: a first frame model from JWST-2D (`jwst[t=0]` for every image cube), and JWST-2D-Res (we used all of the residual images to train a separate residual coding model, i.e., `jwst[t=i+1]` $-$ `jwst[t=i]` for all i, for every image cube). SDSS was split into three datasets: SDSS-2D (`sdss[t=0]`[λ=2]), SDSS-3Dλ (`sdss[t=0]`[λ=1,2,3]), SDSS-3DT (`sdss[t=0,1,2]`[λ=2]).

[5]Other data taking modes such as spectroscopy may be added in future work.

A.2 Accessibility

All datasets herein are released under the AstroCompress project as HuggingFace `datasets` and may be accessed as `numpy` nd-arrays in Python:

```
import numpy as np
from datasets import load_dataset

dataset = load_dataset(f"AstroCompress/{name}", config, split=split, \
                       streaming=True, trust_remote_code=True)
ds = dataset.with_format("np", columns=["image"], dtype=np.uint16)
ds[0]["image"].shape # -> (tuple with shape of the numpy array)
```

Here `name` is one of the five datasets described below, `config` is either "tiny" (small number of files for code testing purposes; default value) or "full" (full dataset), and `split` is one of "train" or "test". To use the datasets with `pytorch`:

```
import torch
from torch.utils.data import DataLoader

dataset = load_dataset(f"AstroCompress/{name}", config, split=split, \
                       streaming=True, trust_remote_code=True)
dst = dataset.with_format("torch", columns=["image"], dtype=torch.uint16, \
                          device=device)
dataloader = DataLoader(dst, batch_size=batch_size, num_workers=num_workers)
next(iter(dataloader)) # -> yields the first batch of images
```

where `device` is the pytorch device to use (e.g., "gpu", "mps:0").

A.3 Dataset collection and scientific details

A.3.1 Related dataset works

With dozens of parameters defining each observation (sky region, exposure time, filter, grating, pupil, etc.) and myriad data structures, existing astronomical archives such as MAST are too unwieldy for ML practitioners. In addition to the previously mentioned Galaxy10, a past dataset curation Hayat et al. [2021] assembled a 266 GB corpus of processed `float32` 64×64 pix^2 5-filter image cutouts around bright galaxies. AstroCompress, in contrast, is focused on `uint16` images that represent a much wider diversity of real-world raw data, including large regions of low SNR and major imaging artifacts.

A.3.2 Dataset collection overview

All of the data that we have pulled is public data from various astronomical archives. The specific licenses under which the data have been released are noted in the acknowledgment section. Here we detail the selection and curation process for each dataset. Table 2 provides a broad overview of the corpus.

At a high level: for each dataset, we generally come up with a set of logical filters and pull a list of *observations* via API and/or direct download from the archives where the data are disseminated. An observation refers to one telescope's "visit" to a certain place in the sky, integrated over a short period of time (typically less than 20 min). Each observation can result in multiple versions of that observation in the archives, as many archives store various processed versions of observations along with the associated calibration data used. For a given observation, we download the one file in raw form (`uint16`; processed data is usually `float32`). Finally, we select a subset of the downloaded data that contains no pairwise overlapping of the image footprint on the sky. This ensures that compression models trained on the training set do not overfit on regions of the sky present in the test data. It also allows users to safely create their own validation sets from subsets of our training sets.

A.3.3 GBI-16-2D (Keck)

This is a 2-D optical imaging dataset from the ground-based W. M. Keck Observatory, obtained in a variety of observing conditions, filters, and across exposure times from 30 seconds to >10 min. The data are all selected from the Low Resolution Imaging Spectrometer (LRIS; Oke et al. [1995]) and are from scientific observations (as opposed to calibration exposures) obtained by one of the co-authors of this paper (J.S.B.) from 2005 to 2010. Since LRIS has a dichroic optical element, which splits the incoming beam at a designated wavelength, the data are obtained either with the blue or red side camera. Each camera has its own filter wheel allowing for imaging in a variety of different bandpasses. The raw data of a given observation are stored in FITS format [Pence et al., 2010]. The LRIS CCDs are readout using 2 amplifiers which are stored in distinct logical and memory sections (called Header Data Units; HDUs) within the FITS files. The blue side camera has two CCDs and the redside camera had 1 CCD before June 2009 and 2 thereafter [Rockosi et al., 2010]. In addition to storing the light-exposed regions, LRIS FITS files usually include small regions of virtually read pixels, called overscan regions, which aid image processing.

Collection

We used the Keck Observatory Archive (KOA) to identify raw LRIS science images obtained under the principal investigator program by J.S.B. These data were then downloaded and checked for potential footprint overlaps using the positional information of the telescope pointing the FITS header. The data assembly code for the HuggingFace GBI-16-2D dataset handles the variety of FITS formats and emits 2D images of size 2248×2048 or 3768×2520 pix^2. An example image from this dataset, with two amplifier reads, is shown in Fig. 1.

A.3.4 SBI-16-2D (Hubble)

This dataset is based on data from the Wide Field Channel (WFC) instrument of the Advanced Camera for Surveys (ACS) onboard the Hubble Space Telescope (HST). Unlike the GBI-16-2D dataset, all observations in SBI-16-2D are obtained in space and with the same bandpass filter (F606W), providing a more uniform point-spread function across the dataset. Our goal with assembling SBI-16-2D is two-fold. First, we aim to provide a large ($>$ 50 GB), raw 2-D optical imaging dataset from space. Second, space-based CCD imaging, unlike ground based imaging, suffers significantly from charge particle collisions with the detectors (called "cosmic-ray [CR] hits"). Such random hits add spurious counts to the affected pixels, corrupting the scientific utility of the observations. WFC CCDs also demonstrate large amounts of charge transfer inefficiency leading to correlated streaks in the vertical direction (see Fig. 1).

Collection

We first compile a list of observations that fit our criteria, including instrument, filter choice, and integration time ranges. This data was then downloaded from the Mikulski Archive for Space Telescopes (MAST): `https://mast.stsci.edu/search/ui/#/hst`. For reproducibility, we provide a script that performs this in our HuggingFace repository located at `SBI-16-2D/utils/pull_hubble_csv.py`. Explanations for each filtering process can be found in the code comments therein. Each FITS file contains two images each of size 4144×2068; these images are stored in the 1st and 4th headers in the FITS files. After removing overlapping regions in the sky, the final dataset amounts to 4282 images of size 4144×2068.

A.3.5 SBI-16-3D (JWST)

This dataset comes from the NIRCAM instrument on the James Webb Space Telescope (JWST). NIRCAM conducts up-the-ramp imaging of various objects in several infrared wavelengths. During up-the-ramp sampling, the aperture is exposed to a region in space, and light continuously accumulates charge on the array. The array repeatedly measures this cumulative charge for multiple frames, thus creating a 3-D time-series image tensor. Every N frames is averaged to create a "group", each of which is stored in our arrays in the T dimension (where N is determined by the "read pattern"). We strongly encourage interested readers to learn more about the JWST readout patterns on the official documentation page [6]. It is worth noting that the time gap between subsequent observations varies,

[6] `https://jwst-docs.stsci.edu/jwst-near-infrared-camera/nircam-instrumentation/nircam-detector-overview/nircam-detector-readout-patterns`

depending on readout pattern, but is roughly in the realm of twenty seconds to two minutes. For this reason, we expect that most physical conditions present in the field of view should remain static across time steps.

Because the instrument repeatedly measures the cumulative optimal charge across time, the pixel value at a given spatial location cube increases with T until reaching a saturation value $(2^{16}-1)$. This directly allows for residual coding, i.e., compressing an initial 2D frame and the temporal differences of 2D frames subsequently.

Collection

Similar to Hubble, we pulled the JWST observations list from the JWST section of MAST. A script version of this can be found at `SBI-16-3D/utils/pull_jwst_csv.py`. Explanations for each filtering process can be found in the code comments there. We provide 1273 images of size 2048 by 2048 by T, where T represents time, under the F200W wavelength filter. T typically ranges as $5 \lessapprox T \lessapprox 10$.

A.3.6 GBI-16-4D (SDSS)

This dataset is assembled from "Stripe 82" of the Sloan Digital Sky Survey (SDSS; York et al. [2000]). Stripe 82, a $\sim$250 sq. deg. equatorial region was repeatedly imaged in 5 optical bandpasses with 30 total CCDs over the course of many months for several years, with supernova discovery as the main science objective [Sako et al., 2018]. Images are obtained in drift scan mode by fixing the telescope in declination and letting the sky naturally move across the field of view as the CCDs are readout at the sidereal rate. For a given nightly "run" across Stripe 82, the SDSS image reduction pipeline [Stoughton et al., 2002] creates (for each of the 6 camera columns, "camcols") "field" images spanning a similar declination (ie., north-south) and right ascension (RA; ie., east-west). The images served by the legacy SDSS archive[7] are 2048×1489 `uint16` pixels, calibrated with a world coordinate system (WCS) that maps pixel location to sky position.

In total, we release 500 GBI-16-4D cubes as part of this dataset, with an uncompressed size of $158GB$. At a fixed wavelength, the images in time mimic the raw `uint16` time sequence produced by imaging satellites, like Kepler, TESS, and CURIOS [Gulick et al., 2022]. Exploiting the correlations in time (and wavelength) should improve the effective compression compared to the single-slice capabilities.

We note that the gap in time between subsequent timestep images may be many nights, resulting in very different background sky conditions. While the background sky levels across nights are relatively uncorrelated, the pixels containing astrophysical sources exhibit significant correlation.

Collection

We queried the SDSS Stripe 82 database to assemble a table of 160k unique Stripe 82 field images deemed to be of top quality (`quality` flag = 3) and then randomly sampled fields and determined the center of the field sky positions. For each such field, we queried the SDSS Stripe 82 database for other images (regardless of `quality`) that overlap the field center and downloaded those images. Since field images from different runs are not aligned in RA, we also downloaded the two fields immediately adjacent and created a stitched-together mosaic of those three images. We then used the WCS in each mosaic image to cut out the same position across all available runs across all available filters.

The assembled data are 4-D cube representations of the same 800×800 pix^2 portion of the sky observed from t=1 up to $\sim$90 times in f=5 filters (u, g, r, i, z). Given the excellent but inherently noisy process of WCS fitting, the $t \times f$ image slices in a given cube are spatially aligned to $<$1 pix. As a result, for a fixed pixel location in a given cube there are high correlations in the pixel value across time and wavelength. While the effective integration time is identical across all image slices, there are slices are of higher signal-to-noise than others in the same cube and all have varying background levels. In the few cases where an image slices does not fully overlap the central region of the anchor field, we fill the missing region with values of zero.

[7] These images are from the reduction pipeline 40/41 ("rerun") and were part of data release 7 from SDSS. Newer reductions of the same raw data were released with different calibrations in `float32` format.

A.3.7 GBI-16-2D-Legacy

This small ground-based dataset obtained on multiple CCDs across many different telescopes is reproduced from the public corpus released by Maireles-González et al. [2023] and assembled by us in HuggingFace `dataset` format. Our experiments only made use of the subset of data from the Las Cumbres Observatory (**LCO**).

Collection

We retrieved all of these files from a download page of Maireles-González et al. [2023] in a `.raw` format and used a script (`GBI-16-2D-Legacy/raw_to_fits.py` in HuggingFace) to convert the images to FITS format. Unlike in the other datasets, we did not check for nor remove potential overlapping images.

A.4 Dataset assembly

Rejection of overlapping images After the initial download of these raw images from various astronomy databases, a significant portion of the necessary data curation is to ensure that the imagery does not overlap on the sky. It is particularly essential to ensure that the spatial footprint of all data in the train set does not overlap with that of the test set. Because we anticipate that future use of this data will split it into validation sets as well, we conservatively ensure that all of our images have pairwise zero overlap.

This was done in two stages, the implementations for which can be found in the `utils/`

Stage 1. Before downloading raw imagery, we first download a metadata file that contains a list of observations and the right ascension (RA) and declination (DEC) of each observation. These are akin to standard spherical coordinates θ and ϕ. The corresponding pixel for the given RA and DEC varies depending on the data source; it can be the image center pixel, the RA and DEC of the target celestial object of interest, or some other pixel within the image entirely. In order to filter out images in this overlapping set, we ran a hierarchical agglomerative clustering (HAC) algorithm via `sklearn.cluster.AgglomerativeClustering` on a matrix of precomputed pairwise angular distance matrix via `astropy.coordinates.angular_separation`. We made a small modification to the astropy source code in order to allow numpy vectorization. The threshold for clustering was selected as `2*FOV`, where `FOV` was the field of view of any given telescope. Within each cluster, a subset of well-separated images was downloaded.

Stage 2. After downloading the data, we were able to use the World Coordinate System (WCS) of each image to map any given pixel in an image to an RA and DEC, using `astropy.wcs`. The Python library `spherical-geometry` is used to compute spherical polygons from the four corners of the image in RA/DEC, and computes overlaps between these polygon objects. In some cases, such as for Hubble, there are two images contained within a single FITS file, so we need to compute spherical polygons for both and then use the union of those two polygons for overlap calculations. Using these algorithms, we further filter out overlapping images within each cluster.

Code to download raw data from source and filter it can be found in the `utils` folder of each HuggingFace dataset. We hope these files, in combination with the observation list assembly code will facilitate efforts for future experts to expand on our dataset from the astronomy data sources themselves.

Disclaimers: Because the Keck dataset did not have WCS information, we did not run stage 2, and instead used a more stringent clustering threshold in stage 1 and took only one image from each cluster. No filtering was done on the very small Legacy dataset; it also contained no WCS, RA or DEC data.

B Further Motivation

We present below a brief amount of quantitative evidence on the explosion of astronomy data scale and the need for advances in its processing.

Astronomy data volumes are growing at an apparently superexponential rate (see Figure 1 of [Maireles-González et al., 2023]). A growth rate of about 10x every 10 years has become nearly 1000x every

10 years. The ability to transmit and store this data will become a massive burden preventing this growth rate from continuing. We hope for advanced compression to alleviate some of this problem.

The current state-of-the-art supercomputers administered by the United States Department of Energy were only recently made ready to handle exascale computing in 2023[8]—specifically Frontier and Aurora, the current two most powerful systems in the world according to `top500.org`. While this "exascale" refers to exaflops rather than exabytes, the need for compute scale is built specifically to address data scale.

In addition, costs per pixel for observatories are declining, allowing the deployment of larger and larger detectors and thus increased scale of data flow. For instance, the focal plane of the Rubin Observatory, now in the commissioning phase, is the largest optical camera ever built, with 20–30 TB of raw imaging data per night containing detectors totaling 3.2 billion pixels [Kahn et al., 2010]. The main imaging camera of the James Webb Space Telescope (JWST) features ten $2048 \times 2048\,\mathrm{pix}^2$ detectors [Gardner et al., 2023], compared to the single $256 \times 256\,\mathrm{pix}^2$ array of the Hubble Space Telescope [Thompson et al., 1998]. Each pixel value includes flux from astronomical sources (which we refer to as "sources" in the rest of the paper), sky background, A/D-introduced noise, and "dark current" arising from the non-zero detector temperature.

In sum, we repeat the breadth of data sources that will soon reach exabyte scale: radio astronomy, satellite imagery[9], genomes, brain mapping and beyond. We note the massive economic potential seen in some of these fields. Lossless methods can be applied without fear, but very carefully crafted lossy designs may soon bring forth orders of magnitude more performant pipelines.

C Baseline Algorithms and Implementation

C.1 The current state of astronomy image compression

Compression is widely used in astronomy to transmit raw data from satellites, to store smaller files in archives, and to speed the movement of files across networks. The consultative Committee for Space Data Systems (CCSDS[10]) periodically reports on recommendations for methods—all variants of JPEG-LS and JPEG-2000 (see below), though not yet JPEG-XL—to compress images and data cubes as they are transmitted in packets from satellites to ground stations. Specialized commercially available space-hardened hardware modules that implement these standards are in use both by NASA and the European Space Agency (ESA). Once on the ground, images are usually held and transmitted by major archives with bzip2, Hcompress, or Rice compression (cf. Pence et al. 2009 and §4.1). Compressed image storage (and manipulation codes; Seaman et al. 2010) are standardized in the file formats, like FITS [Pence et al., 2010] and HDF5, commonly in use by astronomers. Many have studied and proposed refinements of these methods both for lossless [Villafranca et al., 2013, Pata and Schindler, 2015, Thomas et al., 2015, Maireles-González et al., 2023, Mandeel et al., 2021] and lossy [Maireles-González et al., 2022] compression. All of these existing works rely on manually designing codecs using classical probability distributions and transforms.

C.2 Code details

From any dataset parent folder on HuggingFace, run `python utils/eval_baselines.py 2d` to get 2D evaluations of all algorithms. For JWST and SDSS, this `2d` argument may be changed to several other options that can be found in the command arguments `help` docstring. These options allow for JWST residual ("diffs", as stated in the code) compression, compressing entire 3D tensors for JWST and SDSS, as well as some unique SDSS experiments in which we compressed the top 8 bits and bottom 8 bits separately (`2d_top` and `2d_bottom`). We also attempted with poor results to compress 2D SDSS arrays composed of a single spatial pixel, but all wavelengths and timesteps (`tw`). The exact functionalities are documented in the script.

The script also saves the compression ratio, read time, and write time for every single image into a `.csv` file. We have already performed this and uploaded the CSV files for each dataset to the

[8] `https://science.osti.gov/-/media/bes/besac/pdf/202212/7-Helland--BESAC-Panel.pdf`

[9] `https://www.earthdata.nasa.gov/s3fs-public/2022-02/ESDS_Highlights_2019.pdf`

[10] `https://public.ccsds.org/default.aspx`.

HuggingFace parent repository. Additionally, running statistics on mean values are printed to the console.

Implementations for all the non-neural baseline algorithms were adapted from the Python library `imagecodecs=2024.1.1`. All codecs in this library call the native C codec, specifically: JPEG-XL via `libjxl`, JPEG-LS via `charls`, JPEG-2K via `OpenJPEG`, and RICE forked from `cfitsio=3.49`.

All the non-neural baseline algorithms natively support 16-bit inputs. JPEG-XL and JPEG-2K support multi-channel compression directly. To evaluate multichannel arrays using JPEG-LS and RICE, we applied `.reshape((H, -1))` to convert them to single-channel. This was performed only on SDSS data, which resulted in an image height of $H = 800$. For reporting compression ratio, we did not consider the bit-rate cost of transmitting the data shape needed for decoding, as this overhead is negligible for even the smallest image in our dataset.

D Details on neural compression methods

In this section, we provide an in-depth look at our experiments and modifications to existing neural compression methods to work with our astronomical image data. Our implementation can be found at https://anonymous.4open.science/r/AnonAstroCompress/.

D.1 Data format

As mentioned in the main text, most neural image compression methods are designed to handle 3-channel 8-bit RGB images, so we made minor modifications to the neural compression methods to handle the 16-bit data of AstroCompress.

At a high level, we experimented with two approaches: (1) adding support for 16-bit input directly; (2) treating the 16-bit input as the concatenation of two 8-bit inputs – the most significant byte (MSB) and least significant byte (LSB). We used the better of the two when reporting results, and generally found approach (2) to perform similarly or better than approach (1).

- To implement approach (1), we make the following modifications to support 16-bit input directly: For **IDF**, we change the input scaling coefficient from 2^8 to 2^{16}, so that it models the set $\mathbb{Z}/65536$ (instead of $\mathbb{Z}/256$). For **L3C** and **PixelCNN++**, which both use a discretized logistic mixture likelihood model [Salimans et al., 2017], we increase the number of bins from $2^8 - 1$ to $2^{16} - 1$.
- To implement approach (2), we generally convert 16-bit input into 8-bit input while doubling the number of channels. For a 2D (single-frame) image of shape $1 \times H \times W$, this corresponds to treating it as an 8-bit tensor of shape $2 \times H \times W$ where the first channel contains the least significant byte (LSB) and the second channel contains the most significant byte (MSB). For a 3D (multi-frame) image of shape $3 \times H \times W$, this corresponds to treating it as an 8-bit tensor of shape $6 \times H \times W$, where the first three channels contain the LSBs of the original input and the remaining channels contain the MSBs of the original input. The neural compression models are modified accordingly to support the increased number of channels:
  - For **IDF**, we simply doubled the number of input/output channels while keeping the rest of the architecture the same;
  - For **L3C** and **PixelCNN++**, we model 2-channel 8-bit images by using only the "R" and "G" parts of the original RGB autoregressive logistic mixture likelihood model, and we model 6-channel 8-bit images by performing the same autoregressive modeling as the RGB case for the first 3 (LSB) channels, and similarly for the remaining 3 (MSB) channels (so the LSB and MSB channels are modeled separately).

D.2 Architecture and training details

IDF We adopted the implementation from Hoogeboom et al. [2019] at https://github.com/jornpeters/integer_discrete_flows. The network and training hyper-parameters are also set to be consistent with the default configurations from [Hoogeboom et al., 2019], which we find to yield the best results. We train on patches of 32×32 with a batch size of 256. We use a learning rate of 1×10^{-3} and an exponential decay scheduler with rate 0.999.

L3C We adopted the implementation from Mentzer et al. [2019] at https://github.com/fab-jul/L3C-PyTorch. We largely followed the default model configuration provided by [Mentzer et al., 2019] to train the model on 2D image data across all datasets. For 3D experiments, we increased the base convolution filter size and adjusted the latent channel size to 96 and 8, respectively. We trained on 32×32 patches, with a learning rate of 1×10^{-4} and an exponential decay scheduler with rate 0.9.

PixelCNN++ We adopted the implementation from https://github.com/pclucas14/pixel-cnn-pp and adopted the same model architecture as in the default configuration (5 resnet blocks, 160 filters, 12 logistic mixture components, and a learning rate of 2×10^{-4}). We also explored different ways of formatting/modeling 8-bit data (converted from 16-bit input), such as training two separate models for the LSB and MSB, or concatenating the LSB and MSB across the width dimension instead of channel dimension, but did not observe significant improvements compared to the basic approach of stacking the LSB and MSB along the channel dimension (as described in Section D.1).

E Additional Experiment Details

E.1 Handling 16-bit data

Handling 16-bit data Most neural lossless compression methods are designed for RGB image compression, operating on 8-bit (unsigned) integers. To accommodate 16-bit data of AstroCompress, we minimally modify the neural compression methods as follows: for L3C and PixelCNN++, which both use a discretized logistic mixture likelihood model, we increase the number of bins from $2^8 - 1$ to $2^{16} - 1$; for IDF, we simply change the input normalization constant from 2^8 to 2^{16}. Alternatively, we consider treating each 16-bit pixel as two sub-pixels: the most significant byte (MSB) and the least significant byte (LSB), converting a 1-channel 16-bit image into a 2-channel 8-bit image; we then double the number of input channels of each model accordingly. We refer to Supplementary Material for more details.

E.2 Training Splits and Data Preprocessing

For all experiments, we use a fixed split of training and testing images (details in Supplement). The training set is further divided into 85% for training and 15% for validation. For each method, we train and evaluate two model variants as described above, either handling 16-bit input directly or treating it as 8-bit input with double the number of channels; we report the best compression performance between the two variants. We train on random 32×32 spatial patches and apply random horizontal flipping. For evaluation, we divide each image evenly into 32×32 patches, apply reflective padding beforehand if needed. We evaluate the model's compression performance by compressing all patches of an image and combining the results to determine overall performance on that image. This process is then repeated for all test images, and the average compression performance is reported. Compression ratio is calculated as the uncompressed bit depth / negative log-likelihood assigned by the model, aligning closely with actual on-disk entropy coding performance using arithmetic coding.

E.3 Generalization performance

We take IDF as our representative model for investigating generalization performance, where we train and evaluate and all pairs of the single-frame compression tasks. Table 3 shows that the trained model from the Keck experiment generalizes decently across other experiments, even surpassing the model *trained and evaluated* on SDSS-2D. We hypothesize that the aforementioned diversity of wavelength filters in the Keck dataset may explain its unexpected generalizability.

	LCO	Keck	Hubble	JWST-2D	SDSS-2D
LCO	**2.83**	1.01	1.09	0.84	2.31
Keck	<u>2.70</u>	**2.05**	<u>2.20</u>	1.19	**3.02**
Hubble	0.67	0.94	**2.94**	<u>1.22</u>	0.69
JWST-2D	1.46	<u>1.45</u>	1.47	**1.44**	1.50
SDSS-2D	2.27	1.24	1.75	1.02	<u>2.91</u>

Table 3: IDF generalized performance across single-frame datasets. Rows indicate train set; columns indicate test set. Bold indicates best in test set; underline indicates second-best.

E.4 Computational metrics

To offer practical considerations, we present runtime metrics that would be valuable in assessing the feasibility of these methods for real-world applications. Table 4 shows various runtime for inference from neural methods and coding time for classical methods. We ran classical compression codecs on a single thread with a `Intel(R) Xeon(R) Silver 4112 CPU @ 2.60GHz` processor and neural compression network on a single `NVIDIA RTX 6000 ADA`. JPEG-XL with max effort and JPEG-2000 seem to scale quadratically with the number of input pixels, while the other classical algorithms scale linearly with the number of input pixels. Note that in order to work with limited GPU memory, our neural methods operate on 32×32 patches, which allows for linear scaling of encoding complexity but also likely limits the achievable compression ratio. By comparison, our non-neural baselines always receive full images as input.

	SDSS-2D (800x800)	Hubble (4144x2068)
IDF	0.42 ± 0.01	6.03 ± 0.24
L3C	5.18 ± 1.04	73.04 ± 2.36
PixelCNN++	1.48 ± 1.05	20.49 ± 0.18
JPEG-XL max	3.14 ± 0.14	87.76 ± 13.30
JPEG-XL default	0.06 ± 0.002	0.91 ± 0.07
JPEG-LS	0.02 ± 0.0002	0.316 ± 0.04
JPEG-2000	0.09 ± 0.003	1.76 ± 0.11
RICE	0.008 ± 0.0002	0.12 ± 0.02

Table 4: Compression runtime (in seconds/image) on the SDSS-2D and Hubble datasets. For neural methods, the runtime refers to the model likelihood inference time on the full image, without encoding. For classical methods, the runtime refers to the time taken to encode images.

E.5 Details on background noise estimation and source detection for post-experiment analyses

Figures 3 and 4 use data from a single, representative SDSS-2D frame. We used `photutils`[11] estimate the sky's background noise to get the SNR at each pixel, and then to compute a mask of all sources. The 2D background was estimated by dividing the 800x800 image into 50x50 patches and excluding pixels above 3σ of the median value. The medians and standard deviations of the remaining pixels were interpolated to get the final background and noise images. SNR was calculated as `SNR = (original_image - background) / noise`. Source pixels were detected by applying a kernel around any pixels above 3σ, creating a smoothed mask for signal-generating objects.

F Limitations of our work

Limitations of our dataset include a lack of spectroscopic, radio astronomy or floating-point data. In addition, we believe that the significant time delay in between our SDSS-3DT image exposures may not be representative of other optical telescopes that may do back-to-back exposures of the same region. Such a dataset may bring forth significant compression gains, the way we have shown for back-to-back infrared imaging via JWST frame differencing.

Additionally, the three neural compression methods we examined, although established in the literature, likely do not represent the state-of-the-art performance in neural lossless compression. We chose our three algorithms because they are relatively practical, whereas more recent neural compression methods may suffer from long runtimes.

[11] https://photutils.readthedocs.io/en/stable/background.html

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The paper claims regarding a new dataset and the corresponding evaluations are accurately reflected in the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The main goal of the paper is to spur research on neural compression for astronomy data. Limitations of the presented lossless compression methods are clearly stated; readers are, in fact, encouraged to improve on the author's compression results.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [No]

   Justification: this paper is an empirical investigation and does not have mathematical proofs.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Our experimental results rely on already published, publicly available compression models. In cases where the architecture was modified to be compatible with new data formats, we were explicit about the alterations. In addition, we will release the models along with the data sets.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Releasing our astrophysics dataset is among the main goals of this publication. Details are provided in the paper. The code is mainly adopted from published compression models and will be released along with the data.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Our experiments section contains all the details to make our results reproducible. Train/test splits come along with the data publication.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Data compression is a deterministic approach. We report compression ratios obtained on the full test data sets, which is a deterministic and reproducible operation.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We tried to be as detailed as possible about the computing resources used to train the model.

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: As our paper describes an astronomy data compression study, we are in agreement with the Code of Ethics.

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: See Broader Impacts paragraph.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We do not anticipate risks associated with the release of astrophysical data.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All data sources have been credited by citing the appropriate publications.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The new assets include the release of the four preprocessed astronomy datasets released with this paper. The data will be hosted on HuggingFace, and the license type will be provided.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The study contains neither crowdsourcing nor human subjects.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: There are no human subjects involved; IRB approval does not apply.