The Journal of Open Source Software

deepCR: Cosmic Rejection with Deep Learning

Keming Zhang[1] **and Joshua S. Bloom**[1, 2]

1 Department of Astronomy, University of California, Berkeley **2** Lawrence Berkeley National Laboratory

DOI: 10.21105/joss.01651

Software

- Review
- Repository
- Archive

Submitted: 13 August 2019
Published: 02 September 2019

In partnership with

AMERICAN ASTRONOMICAL SOCIETY

This article and software are linked with research article DOI 10.3847/1538-4357/ab3fa6, published in the Astrophysical Journal.

Summary

Astronomical imaging and spectroscopy data are frequently corrupted by "cosmic rays" (CR) which are high energy charged particles that are instrumental, terrestrial, or cosmic in origin. When such particles pass through solid state detectors, such as charged coupled devices (CCDs), they create excess flux in the pixels hit which lead to artifacts in images. These artifacts must be identified and either masked or replaced, before further scientific analysis could be done on the image data. It is straightforward to identify these artifacts when multiple exposures of the same field are taken. In such cases, a median image could be calculated from aligned single exposures, effectively creating a CR-free image. Each one of the exposures is then compared with the median image to identify the cosmic rays. However, when CCD read-out times are non-negligible, or when sources of interest are transient or variable, cosmic ray rejection with multiple exposures can be sub-optimal or infeasible. These cases would require specialized algorithms to detect cosmic rays in single images.

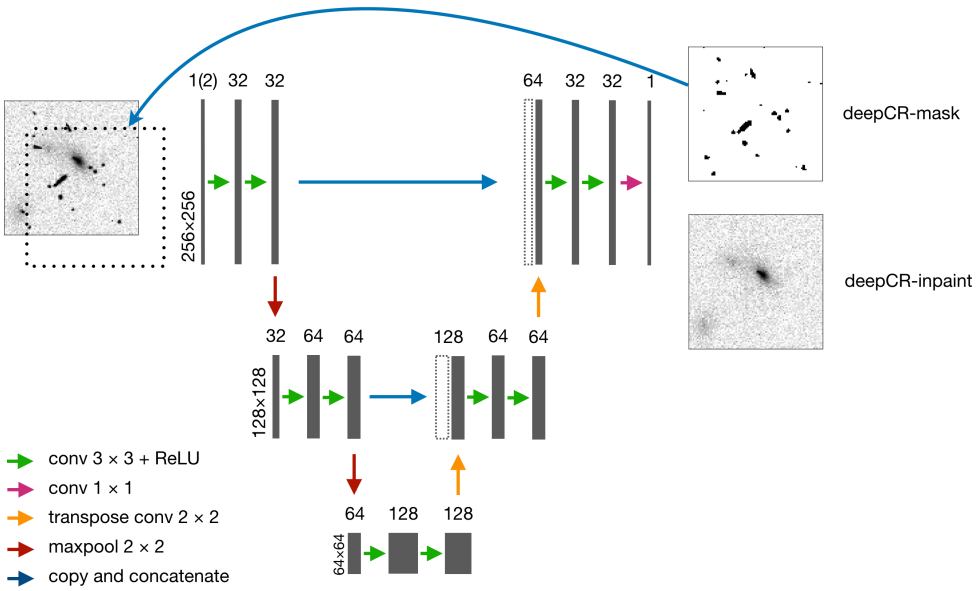

Figure 1: Neural network architecture of deepCR. Feature maps are represented by gray boxes while the number of channels and example feature map dimensions are indicated on the top of and to the left of each feature map, respectively. Different computational operations are marked in the legend to the lower left. Unfilled boxes to the right of blue arrows represent feature maps directly copied from the left, which are to be concatenated with the adjacent feature map. To apply the inpainting model, the predicted mask (dotted box at left) is concatenated with the original image as the input.

deepCR is a Python package for single frame cosmic ray rejection which is based on deep learning and written with the Pytorch framework (Paszke et al., 2017). Since deepCR is

based on deep learning, different models trained on data taken with different instrument configurations are required, when applied to different data. The current version of `deepCR` is prepackaged with model for Hubble Space Telescope ACS/WFC imaging data, and we expect models available to grow with contribution from the community. We plan to host a "model zoo" which enables `deepCR` to work across different instrument configurations.

The API of `deepCR` includes functionality for both applying models and training models. To apply an available model, `deepCR` takes in an input image and produces a cosmic ray mask and an "inpainted" image, with the artifact pixels replaced with `deepCR` predictions. To train a new model, users would feed in custom dataset to the training API, which is automated. `deepCR` works with both CPU, which is well-threaded at application time, and GPU. On GPU, training a new model takes as short as 20 minutes, while applying `deepCR` on a 10 Mpix image requires less than 0.2 second, orders of magnitude faster than current state of the art `LACosmic` (van Dokkum, 2001).

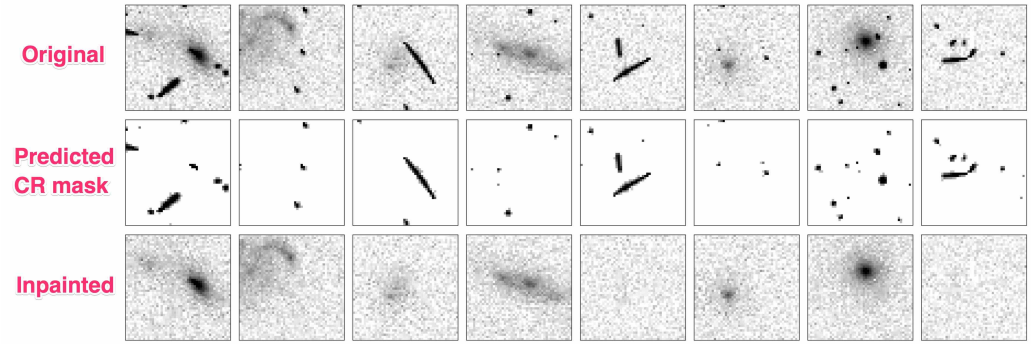

Figure 2: Examples of cosmic ray contaminated image cutouts (first row), deepCR cosmic ray mask predictions (middle row), and images with artifact pixels replaced with deepCR predictions (last row).

In the paper accompanying `deepCR` (Zhang & Bloom, 2019), the authors showed that on Hubble Space Telescope (HST) ACS/WFC data, `deepCR` is more robust, and at least as fast as the current state-of-the-art single frame cosmic ray rejection package, `LACosmic`. The API of `deepCR` serve as a drop in replacement for `LACosmic`, so that users may experiment with different packages easily. At reasonable false detection rates, `deepCR` achieved near perfect cosmic ray detection in extragalactic and globular cluster fields, and above 90% in more difficult dense stellar fields in nearby resolved galaxies. Since HST imaging is among the hardest cosmic ray rejection to be solved, `deepCR` would work well across many different instrument set-ups, including ground based imaging and spectroscopy. The combination of speed and accuracy of `deepCR` allows astronomers to potentially save large amounts of precious observational and computational resources.

Acknowledgements

This work was supported by a Gordon and Betty Moore Foundation Data-Driven Discovery grant, and has made use of the following software:

astropy (Astropy Collaboration et al., 2018); astrodrizzle (Hack et al., 2012); numpy (van der Walt, Colbert, & Varoquaux, 2011); scipy (Jones, Oliphant, Peterson, & others, 2001); matplotlib (Hunter, 2007); astroscrappy (C. McCully et al., 2018); pytorch (Paszke et al., 2017); Jupyter (Kluyver et al., 2016); Scikit-image (Walt et al., 2014)

References

Astropy Collaboration, Price-Whelan, A. M., Sipőcz, B. M., Günther, H. M., Lim, P. L., Crawford, S. M., Conseil, S., et al. (2018). The astropy project: Building an open-science project and status of the v2.0 core package. *The Astronomical Journal*, *156*(3), 123. doi:10.3847/1538-3881/aabc4f

Hack, W. J., Dencheva, N., Fruchter, A. S., Armstrong, A., Avila, R., Baggett, S., Bray, E., et al. (2012). AstroDrizzle: More than a New MultiDrizzle. In *American astronomical society meeting abstracts #220*, American astronomical society meeting abstracts (Vol. 220, p. 135.15).

Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. *Computing in Science and Engineering*, *9*, 90–95. doi:10.1109/MCSE.2007.55

Jones, E., Oliphant, T., Peterson, P., & others. (2001). SciPy: Open source scientific tools for Python. http://www.scipy.org/.

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., et al. (2016). Jupyter notebooks – a publishing format for reproducible computational workflows. (F. Loizides & B. Schmidt, Eds.). IOS Press.

McCully, C., Crawford, S., Kovacs, G., Tollerud, E., Betts, E., Bradley, L., Craig, M., et al. (2018, November). Astropy/astroscrappy: V1.0.5 zenodo release. doi:10.5281/zenodo.1482019

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., et al. (2017). Automatic differentiation in pytorch. In *NIPS-workshop*.

van der Walt, S., Colbert, S. C., & Varoquaux, G. (2011). The numpy array: A structure for efficient numerical computation. *Computing in Science Engineering*, *13*(2), 22–30. doi:10.1109/MCSE.2011.37

van Dokkum, P. G. (2001). Cosmic-Ray Rejection by Laplacian Edge Detection, *113*, 1420–1427. doi:10.1086/323894

Walt, S. van der, Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., et al. (2014). Scikit-image: Image processing in Python. *PeerJ*, *2*, e453. doi:10.7717/peerj.453

Zhang, K., & Bloom, J. S. (2019). deepCR: Cosmic Ray Rejection with Deep Learning. *arXiv e-prints*.