Convolutional Autoencoders for Denoising Solar Images

Jimmy Lynch

Lockheed Martin Solar and Astrophysics Lab, Palo Alto, CA

Abstract

Denoising solar images is crucial to ensuring accurate physical conclusions are drawn from the data. Convolutional autoencoders—neural networks that compress data into lower-dimensional representations and learn to construct the original image from them—have shown promise in denoising. Using open-source packages zarr, xarray, dask, and torch, we trained and tested a denoising autoencoder on ~5000 solar images taken by the Atmospheric Imaging Assembly (AIA) onboard Solar Dynamics Observatory (SDO). Our investigation reveals that while the autoencoder successfully reconstructs the intricate structures within the solar images, it encounters challenges in accurately reproducing the intensity levels of the most luminous and dimmest regions. This finding underscores the complexities of solar image denoising and opens avenues for future research and improvement.

Our Mission

Understanding the physical mechanisms behind our sun's variable activity can help society prepare for—and maybe even predict—periods of high activity.

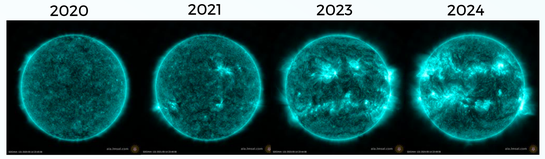

Fig. 1. The sun imaged at 131Å on May 14th of each year.

The Atmospheric Imaging Assembly (AIA) aboard NASA's Solar Dynamics Observatory (SDO) studies the sun's atmosphere in the extreme ultraviolet (EUV), revealing magnetic field structure and its role in producing the solar wind and energetic particles.

AIA Statistics:
- Image cadence of ~12s
- Online for over 14 years
- 300 million 4k snapshots
- >25PB of data

Fig. 2. AIA telescope assemblies

Denoising these images is critical to ensuring accurate physical conclusions are drawn from the data. Further, it's important to consider efficiency when dealing with monumental amounts of data.

Hence, our scientific goals are two-fold:
1. Explore the use of machine learning to reduce sources of noise in AIA data
2. Do so in a way that efficiently links data access and data analysis

Why Python?

Several open-source Python packages have been instrumental in developing our pipelines:

xarray dask Zarr PyTorch

Python enables us to jointly use these packages to:
1. Create labeled, high-dimensionality arrays (xarray)
2. Write these arrays in parallel (dask) as chunked datasets (zarr)
3. Load these to ML via a custom data loader (torch)

Methodology

We utilize the HelioCloud, a cloud-based computing resource with access to 4-minute cadence AIA data as FITS files. We prep 15 days of the 94Å channel data (~5000 images) by compressing the HD images to 128x128 pixel resolution (Fig. 3) and writing the datasets as zarr objects. Our torch dataloader will then select only the necessary images for a training batch (Fig. 4).

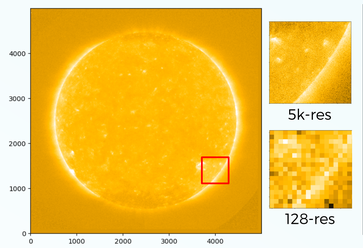

Fig. 3. 5k versus 128 resolution with cutouts.

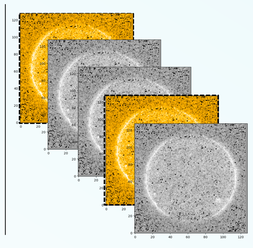

Fig. 4. Selected zarr data by torch

We add artificial noise sampled from a zero-mean, unit variance Gaussian to form noisy images (dirty) from our existing images (clean). The dirty images serve as the inputs to our autoencoder, while the clean images serve as the target (Fig. 5). We utilize a general autoencoder architecture in torch (Fig 6.) and train the model to identify and remove the added artificial noise.

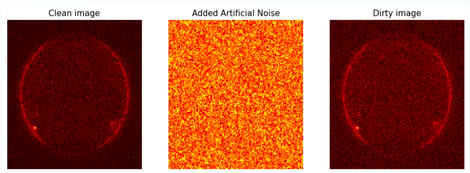

Fig. 5. Clean image (observed by AIA) + artificial noise = dirty image

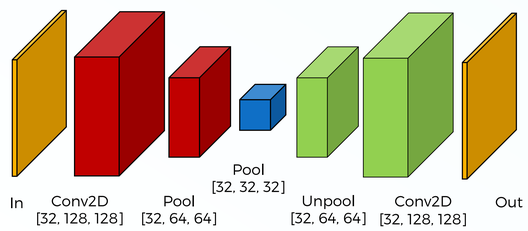

Fig. 6. Best-performing autoencoder. Each layer is separated by a hyperbolic tangent activation function (not pictured).

We follow a standard training procedure, with an Adam optimizer, a constant learning rate (1e-4), a mean-square error loss function, and a validation step after each epoch of training to identify overfitting.

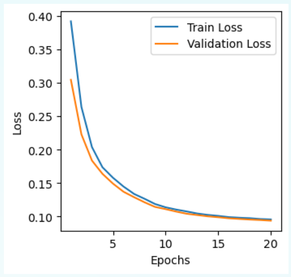

Fig. 6. Smooth decreasing loss shows healthy learning.

Results

The results of the best-performing autoencoder are shown below (Fig. 7). We find that this autoencoder recovers much of the structure, such as the outline of the solar disk and bright/dim spots. However, the model fails to reproduce regions of high/low counts.

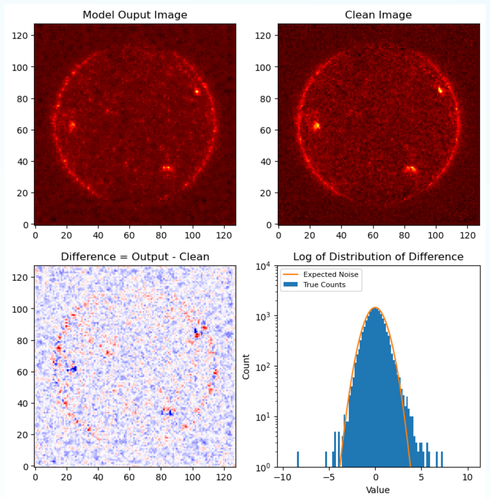

Fig. 7. Results of testing the autoencoder on withheld test data.

Discussion

1. Reducing sources of noise

While we do observe that the signal difference between the model output and the clean images is mostly random noise as expected, we resolve underlying, unwanted solar disk structure. This behavior is likely due to the fact that active regions of the sun produce a signal orders of magnitude larger than the average signal from the solar disk.

Future work aims to improve performance of the autoencoder by understanding how this "spiky" data can be properly recovered. We plan to test different data normalization methods, such as logarithmic scaling, and test on images with higher activity to increase spiky data counts.

2. Efficiently linking data access and analysis

We explored the use of packages such as xarray, dask, zarr, and torch to build an efficient pipeline for autoencoder training. This pipeline will likely be efficient in training networks on cutout images—for instance, 100x100 pixel active regions—as torch can load only the chunks that contain the cutout region. In future work, we aim to quantify this benefit through benchmark testing among different cutout sizes between FITS data and zarr-formatted data.

Acknowledgements

This work is supported by NASA's SDO/AIA contract (NNG04EA00C) to LMSAL. The author thank Mark Cheung, Meng Jin, and Nabil Freij for their continuous support and guidance.

Citations

[1] Hoyer, S. &. (2017). xarray: N-D labeled Arrays and Datasets in Python. Journal of Open Research Software.
[2] Miles, A. K. (n.d.). zarr-developers/zarr-python: v2.16.1. Zenodo.
[3] Rocklin, M. (2015). Dask: Parallel computation with blocked algorithms and task scheduling. Proceedings of the 14th python in science conference.
[4] Paszke, A. a. (2017). Automatic differentiation in PyTorch.
[5] Lemen, J. R. et al. (2012). The Atmospheric Imaging Assembly (AIA) on the Solar Dynamics Observatory (SDO). doi:10.1007/s11207-011-9776-8