
Learning the solar latent space: sigma-variational autoencoders for multiple channel solar imaging

Edward J. E. Brown
University of Cambridge
edward.j.e.brown@gmail.com

Stefano Bonasera
University of Colorado, Boulder
stefano.bonasera@colorado.edu

Bernard Benson
The University of Alabama in Huntsville
bb0008@uah.edu

Jorge A. Pérez-Hernández
Mexico National Autonomous University
jperez@icf.unam.mx

Giacomo Acciarini
University of Strathclyde
giacomo.acciarini@gmail.com

Atılım Güneş Baydin
University of Oxford
gunes@robots.ox.ac.uk

Christopher Bridges
University of Surrey
c.p.bridges@surrey.ac.uk

Meng Jin
Lockheed Martin Solar
Astrophysics Lab
jimmeng@lmsal.com

Eric Sutton
University of Colorado, Boulder
eric.sutton@colorado.edu

Moriba K. Jah
University of Texas, Austin
moriba@utexas.edu

Abstract

This study uses a sigma-variational autoencoder to learn a latent space of solar images using the 12 channels taken by Atmospheric Imaging Assembly (AIA) and the Helioseismic and Magnetic Imager (HMI) instruments on-board the NASA Solar Dynamics Observatory. The model is able to significantly compress the large image dataset to 0.19% of its original size while still proficiently reconstructing the original images. As a downstream task making use of the learned representation, this study demonstrates the use of the learned latent space as an input to improve the forecasts of the F30 solar radio flux index, compared to an off-the-shelf pretrained ResNet feature extractor. Finally, the developed models can be used to generate realistic synthetic solar images by sampling from the learned latent space.

1 Introduction

The NASA mission Solar Dynamics Observatory (SDO) and the on-board Atmospheric Imaging Assembly (AIA) [1] and Helioseismic and Magnetic Imager (HMI) [2] instruments, launched in 2010, have been taking high resolution images of the solar disk every 12 seconds. The AIA images include nine extreme ultra-violet (EUV) channels with wavelengths at 94 Å, 131 Å, 171 Å, 193 Å, 211 Å, 304 Å, 335 Å, 1,600 Å, 1,700 Å, and the HMI images include three channels measuring the strength of the magnetic field in three directions B_x, B_y, B_z , where the $+x$ direction points to the solar west, $+y$ to the north, and $+z$ out of the image plane towards Earth. See Figure 1 for an example data-point. In the last few years, there have been various data-driven studies enabled

by the release of the SDOML dataset [3] curated for machine learning (ML) applications of SDO data. This dataset allowed rapid development of computer vision techniques for SDO data because it had been pre-processed, amongst other things, to calibrate the pixel intensities, remove corrupted images, and calibrate between images geo-spatially. Examples include the work by Park et al. [4] which translates from the HMI channels to the AIA channels and by Upendran et al. [5] which uses solar images to forecast physical phenomena such as the solar wind speed. This study implements the sigma-variational autoencoder [6], an extension of the original variational autoencoder (VAE) architecture [7] with the 12 channels of solar images using a convolutional encoder-decoder pair. Compression, reconstruction, feature extraction and a solar generative model are studied.

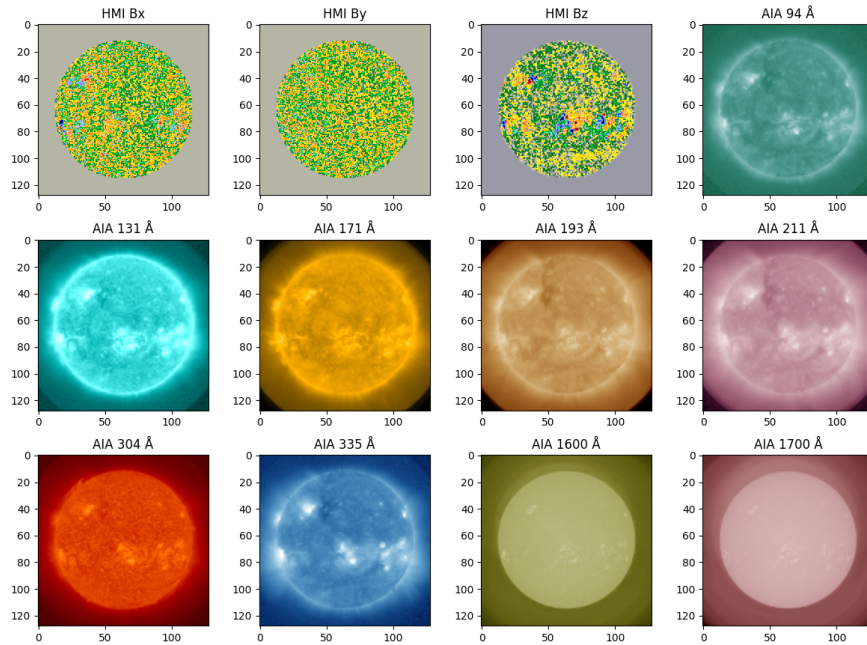


Figure 1: Solar images taken by the Solar Dynamic Observatory [1, 2]

2 Methodology

2.1 Sigma Variational Autoencoder

We use a sigma-variational autoencoder (sigma-VAE [6]) to reconstruct 12 channels of solar images through a bottleneck representing a latent space. Figure 2 provides a schematic diagram show’s the model architecture.

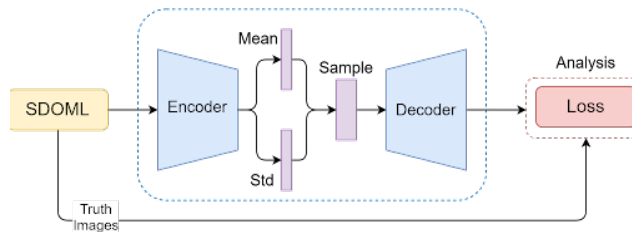


Figure 2: Variational autoencoder diagram

In this model, the encoder provides a mean and standard deviation as an output of different sizes referred to as the “bottleneck size”. A sample is generated by forming a normal distribution parameterized by the mean and standard deviation and drawing from this distribution. This sample is passed to the decoder which attempts to reconstruct the original image. A consequence of training a VAE is learning a deep generative model involving the learned latent space and the decoder. Assuming the learned latent space has a standard normal distribution, one can sample from the learned latent space and can feed this sample to the decoder to generate a synthetic image consistent with the properties represented in the training data. The encoder architecture consists of eight convolutional layers [8]

following the same pattern shown in Table 2 in the appendix. Each layer is followed by a batchnorm layer [9] and a rectified linear unit [10]. A linear layer takes the output of the encoder to produce the mean and similarly for the standard deviation. The decoder architecture is an exact mirror of the encoder except using transposed convolutional layers, and there is no batchnorm nor relu unit in the final layer. The feature that makes this a ‘sigma’ VAE is the form of the loss function. A typical loss function for a VAE is formed of the addition of the reconstruction loss (typically the mean-squared error between the sample and the reconstruction) and the KL divergence between the latent space versus a unit normal Gaussian distribution. Equation 1 shows the loss function for a β -VAE as described in [11]. Essentially, the practitioner manually chooses a β value to weight the different loss terms relative to each-other.

$$L = \beta \text{KL} + \frac{D}{2} \text{MSE} \quad (1)$$

where D is the dimensionality of the sample, β is a chosen scaling factor, KL is the KL divergence between the latent space and the unit normal gaussian, and MSE is the mean squared error between the sample and the reconstruction.

$$L = \text{KL} + \frac{D}{2\sigma^2} \text{MSE} + D \log \sigma \quad (2)$$

Equation 2 shows the updated loss function. The sigma VAE difference is to add a learnable ‘variance’ (σ) as a scaling factor to the MSE but the size of that scaling factor is regularised in an additional additive log term. More details on the theory behind this can be read in [6].

Models for bottleneck sizes 256, 512, 1,024 and 2,048 are trained. The SDOML dataset is available online at <https://pur1.stanford.edu/vk217bh4910>. For our initial study, we resize the images to 128x128 pixels from the original resolution of 512x512.

2.2 Feature extraction: F30 forecasts

F30 is a radio measurement at 30 cm wavelength of radio instruments monitoring the Sun. It is treated as a proxy for solar ultra-violet irradiance data. It has uses in applications such as thermospheric density modelling [12]. Previous works using solar images as inputs to forecast space weather phenomena at Earth use pre-trained feature extractors such as GoogleNet [13] or ResNet [14], to extract solar features, such as Upendran et al. [5]. However, in this work we propose that a VAE trained on all 12 channels (EUV and HMI) is able to capture better representational features by encoding and decoding the EUV and HMI channels jointly. For this study, we compare features outputted from a ResNet using the 211 Å channel and our VAE using all 12 channels. Features are extracted from the VAE by using the mean outputted by the encoder. Features are extracted from the ResNet model by using the outputs from the second to last layer. An Extra Trees Regressor model, with a choice of [250, 500, 750, 1000, 1500, 2000] estimators, is trained on both sets of extracted features. Otherwise, the default setting from scikit-learn v1.0 is used [15]. This algorithm was chosen due to its simplicity of implementation, lack of need to scale input data, and robustness to over-fitting due to its ensemble of weak models nature. The dataset comprises every day from June 2010 to December 2018. The validation dates are F30 values for every day in March 2011–2018, the test dates are for every day in January 2011–2018 and everything else is used as training data. Furthermore, the results from a simple persistence model are included as a further point of comparison. The F30 data is available at https://lasp.colorado.edu/lisird/data/cls_radio_flux_f30/.

2.3 Reconstruction and compression statistics

The reconstructed images are evaluated against the original images using four metrics. First is the pixel correlation metric, which is the Pearson correlation of the original pixels to the reconstructed pixels, the percentage of pixels that are within 10% and 50% of their original value (ppe10 and ppe50) respectively, and the root mean squared error in the original pixel space. For this task, a model is trained on the whole dataset without test and validation set. The purpose is not for generalisation, it is to show that decent reconstructions can be made, but from a dataset size a tiny fraction of the original.

3 Experiments

3.1 Reconstruction

Section D in the Appendix contains the reconstruction metric for reconstructed images for the different channels and bottleneck sizes. Notably, the longer wavelengths 1,600 Å and 1,700 Å have pixel

correlations greater than 0.98. Furthermore, the correlation reconstruction metrics are very favourable for the EUV images as a whole with >0.91 for all sizes and channels. Lastly, the performance for B_x and B_y is clearly poor. However this is because of the lack of structure present in these images—there is a significant amount of noise that is not reconstructable with respect to correlation. A reconstruction is included in the appendix in Figure 5 in Section C. The original images are in Figure 1.

The last row in Table 1 in the Appendix shows the overall dataset reduction size. 0.19% represents a significant dataset reduction. Furthermore, considering that some EUV images were in the high 90s in pixel correlation, this method can be applied to significantly reduce the dataset size while also being able to reconstruct original image. This is significant for the field because it has implications for more efficient image storage/transmission on-board satellites- since they only have to store the reduced features. Furthermore, it increases accessibility to what is otherwise a prohibitively large dataset.

3.2 Generative model

Figure 3 shows images generated from sampling from the learned latent space. This is done by assuming each element of the bottleneck vector is unit normally distributed, and sampling from each element independently. The sampled vector is given to the decoder and an image is generated. For comparison, see Figure 1 for a real set of images. Notably, features appear in multiple channels in the same location on the solar disk. There is a variety of features from corona to active regions. True to their nature, the 1,600 Å and 1,700 Å images have less structure in them, but still feature a brighter spot where the other AIA images have one (left side). We believe that this method for generating solar images will have uses in the ML and heliophysics community firstly as simulated solar conditions for models to consume, and secondly as a source of synthetic training data.

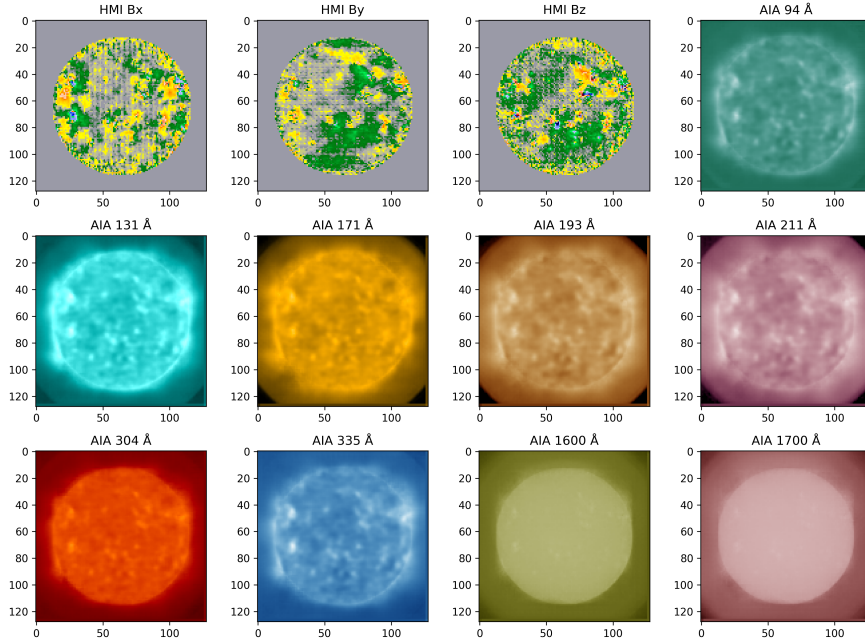


Figure 3: Generated images sampling from the latent space

3.3 Feature extraction: F30 forecasts

Figure 4 shows results for a comparison between the forecast accuracy of the autoencoder vs the ResNet as a feature extractor. Notably, the autoencoder features improve on the ResNet values significantly at lower forecast horizons, 1-2 days. Furthermore, the autoencoder outperforms the persistence model, while the ResNet does not. The F30 index is highly auto-regressive at daily temporal resolutions, hence why our model barely outperforms the persistence model. However, our model has no access to that persisted value, so the fact that it is able to compete with such a highly auto-regressive feature is impressive. Notably, at four days it improves the forecast by 8.5% in RMSE over the persistence model.

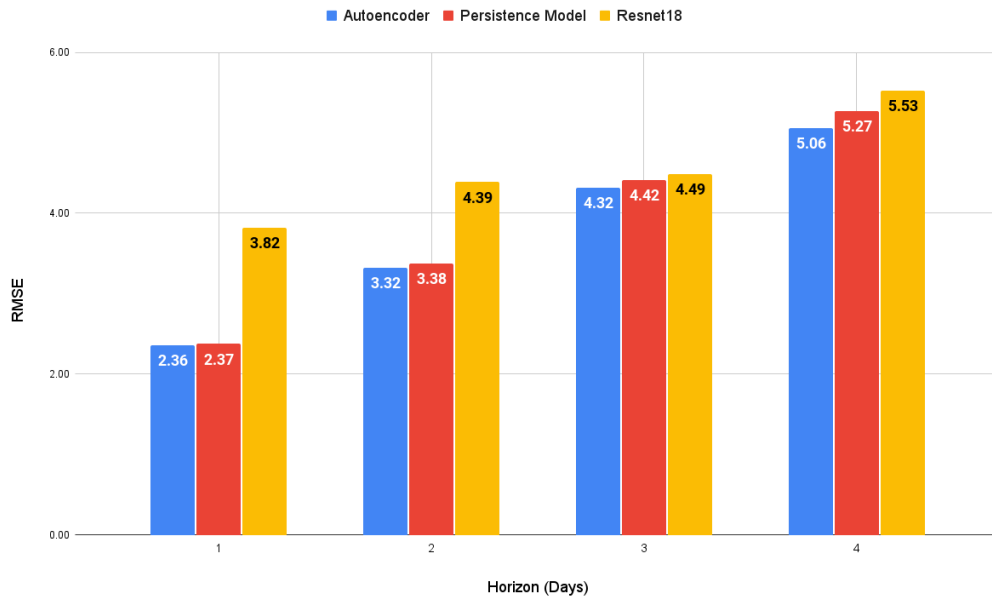


Figure 4: F30 forecasts: Variational autoencoder vs persistence model and ResNet

4 Future work

Concerning the F30 forecasts, the presented proof-of-concept comparison with a single-channel ResNet is not ideal and we would like to extend this to features generated by ResNet18 for each channel individually and concatenated for fairer comparison. The ability to condense all the channels in one model run is seen as a very positive feature of the model however. Another limitation could be the choice of January and March for the test and validation sets respectively. The study reached approximately 30 days of GPU time on V100 Nvidia GPUs. Limiting this number generally meant reducing the number of experiments that could be run. A more in-depth comparison would do a cross validation with each fold having one or two test months. This would eliminate any concerns about randomness between test sets. However, this results in training the autoencoder multiple times for different folds, and was not conducted to limit the computational overhead. Lastly, the visual artefacts in the reconstructed images and generated images from the HMI channel clearly show more work is needed to design better calibrated decoder architectures. This is all the subject of our ongoing work to extend the initial results presented at this workshop.

5 Conclusion

This study presents a sigma-VAE that learns to reconstruct 12 channels of extreme ultra-violet and magnetic images from the SDOML dataset. The trained model can reduce the SDOML dataset to 0.19% of its original size whilst maintaining high fidelity reconstruction of the EUV images, and to some extent the z -component of the magnetic field. This is significant for on-board satellite data storage/transmission and accessibility to solar data. Furthermore, the generative model of the Sun is able to generate realistic solar images by sampling from the latent space. Lastly, the latent space of the solar images can be used as an extracted feature to forecast solar related indices, as evidenced by its ability to forecast F30.

6 Acknowledgments

This work has been enabled by the Frontier Development Lab (FDL.ai). FDL is a co-operative agreement between NASA, the SETI Institute (seti.org) and Trillium Technologies Inc, in partnership with Google Cloud and Intel. We would like to thank Drs. L. Guhathakurta, M. Jin, J. van den Ijssel, E. Doornbos, A. Muñoz-Jaramillo, A. Vourlidis, I. Telezhinsky and T.S. Kelso for sharing their technical expertise and James Parr, Jodie Hughes and Belina Raffy for their support.

References

- [1] James R Lemen, David J Akin, Paul F Boerner, Catherine Chou, Jerry F Drake, Dexter W Duncan, Christopher G Edwards, Frank M Friedlaender, Gary F Heyman, Neal E Hurlburt, et al. The atmospheric imaging assembly (aia) on the solar dynamics observatory (sdo). In *The solar dynamics observatory*, pages 17–40. Springer, 2011.
- [2] P. H. "Scherrer, J. Schou, R. I. Bush, A. G. Kosovichev, R. S. Bogart, J. T. Hoeksema, Y. Liu, T. L. Duvall, J. Zhao, A. M. Title, C. J. Schrijver, T. D. Tarbell, and S." Tomczyk. *The Helioseismic and Magnetic Imager (HMI) Investigation for the Solar Dynamics Observatory (SDO)*, pages 207–227. Springer US, New York, NY, 2012. ISBN 978-1-4614-3673-7. doi: 10.1007/978-1-4614-3673-7_10. URL https://doi.org/10.1007/978-1-4614-3673-7_10.
- [3] Richard Galvez, David F. Fouhey, Meng Jin, Alexandre Szenicer, Andrés Muñoz-Jaramillo, Mark C. M. Cheung, Paul J. Wright, Monica G. Bobra, Yang Liu, James Mason, and Rajat Thomas. A machine-learning data set prepared from the NASA solar dynamics observatory mission. *The Astrophysical Journal Supplement Series*, 242(1):7, may 2019. doi: 10.3847/1538-4365/ab1005. URL <https://doi.org/10.3847/1538-4365/ab1005>.
- [4] Eunsu Park, Yong-Jae Moon, Jin-Yi Lee, Rok-Soon Kim, Harim Lee, Daye Lim, Gyungin Shin, and Taeyoung Kim. Generation of solar UV and EUV images from sdo/hmi magnetograms by deep learning. *The Astrophysical Journal*, 884(1):L23, oct 2019. doi: 10.3847/2041-8213/ab46bb. URL <https://doi.org/10.3847/2041-8213/ab46bb>.
- [5] Vishal Upendran, Mark C. M. Cheung, Shrahan Hanasoge, and Ganapathy Krishnamurthi. Solar wind prediction using deep learning. *Space Weather*, 18(9):e2020SW002478, 2020. doi: <https://doi.org/10.1029/2020SW002478>. URL <https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2020SW002478>. e2020SW002478 10.1029/2020SW002478.
- [6] Oleh Rybkin, Kostas Daniilidis, and Sergey Levine. Simple and effective vae training with calibrated decoders, 06 2020.
- [7] Diederik P. Kingma and M. Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2014.
- [8] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.
- [9] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. pages 448–456, 2015. URL <http://jmlr.org/proceedings/papers/v37/ioffe15.pdf>.
- [10] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.
- [11] Irina Higgins, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017.
- [12] Dudok de Wit, Thierry and Bruinsma, Sean. The 30 cm radio flux as a solar proxy for thermosphere density modelling. *J. Space Weather Space Clim.*, 7:A9, 2017. doi: 10.1051/swsc/2017008. URL <https://doi.org/10.1051/swsc/2017008>.
- [13] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. URL <http://arxiv.org/abs/1409.4842>.
- [14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.
- [15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

A Compression statistics

Table 1: Dataset compression

Bottleneck size	256	512	1,024	2,048
Original dataset size (MB)	238,174.5	238,174.5	238,174.5	238,174.5
Model Size (MB)	131.1	155.3	203.3	299.3
Reduced Dataset Size (MB)	310.1	620.2	1240.5	2481.0
Total Size After (MB)	441.2	775.5	1443.8	2780.3
Percentage of Size (MB)	0.19%	0.33%	0.61%	1.17%

B Encoder architecture

Table 2: Encoder convolutional layers

filter number	filter size (square filter)	stride
64	4	2
128	4	2
256	4	1
512	4	1
512	4	1
512	3	1
512	3	2
512	3	2

C Reconstructed images

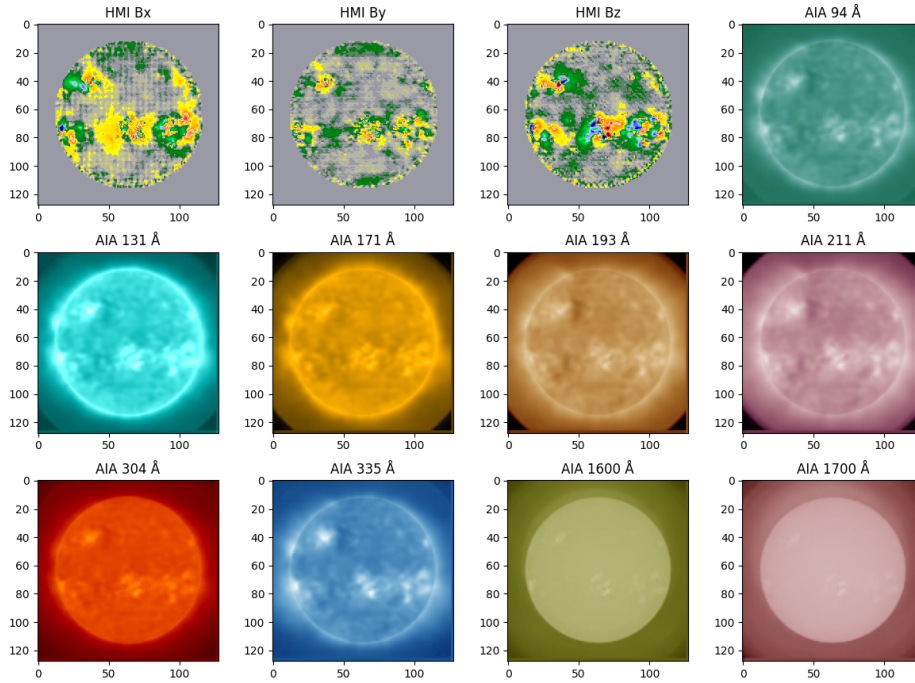


Figure 5: Reconstructed images

D Reconstruction metrics

Table 3: Bottleneck size 256 reconstruction metrics

image type	pixel correlation	ppe50s	ppe10s	rmse pixel value
94 Å	0.92	0.98	0.51	3.28
131 Å	0.94	0.97	0.52	10.46
171 Å	0.94	0.94	0.48	209.46
193 Å	0.96	0.96	0.54	228.97
211 Å	0.96	0.96	0.55	95.08
304 Å	0.92	0.95	0.46	74.87
335 Å	0.95	0.98	0.54	8.26
1600 Å	0.99	0.99	0.80	22.33
1700 Å	0.99	0.98	0.86	230.50
B_x	0.52	0.05	0.01	17.75
B_y	0.36	0.04	0.01	16.61
B_z	0.84	0.08	0.02	5.39

Table 4: Bottleneck size 512 reconstruction metrics

image type	pixel correlation	ppe50	ppe10	rmse intensity
94 Å	0.92	0.98	0.51	3.26
131 Å	0.94	0.97	0.52	10.26
171 Å	0.94	0.95	0.48	203.55
193 Å	0.96	0.96	0.55	218.26
211 Å	0.96	0.97	0.56	91.16
304 Å	0.92	0.95	0.46	70.53
335 Å	0.96	0.98	0.54	7.88
1600 Å	0.99	0.99	0.80	22.25
1700 Å	0.99	0.98	0.87	230.76
B_x	0.52	0.05	0.01	17.66
B_y	0.36	0.04	0.01	16.59
B_z	0.84	0.08	0.02	5.29

Table 5: Bottleneck size 1024 reconstruction metrics

	pixel correlation	ppe50	ppe10s	rmse intensity
94 Å	0.92	0.98	0.52	3.25
131 Å	0.94	0.97	0.53	10.28
171 Å	0.94	0.95	0.49	203.03
193 Å	0.96	0.96	0.55	213.30
211 Å	0.96	0.96	0.57	89.77
304 Å	0.92	0.95	0.47	70.51
335 Å	0.96	0.98	0.55	7.85
1600 Å	0.99	0.99	0.80	22.18
1700 Å	0.99	0.98	0.86	234.50
B_x	0.53	0.05	0.01	17.59
B_y	0.37	0.04	0.01	16.56
B_z	0.84	0.08	0.02	5.20

Table 6: Bottleneck size 2048 reconstruction metrics

image type	pixel correlation	ppe50	ppe10	rmse intensity
94 Å	0.92	0.98	0.52	3.24
131 Å	0.94	0.97	0.53	10.22
171 Å	0.94	0.95	0.48	201.71
193 Å	0.96	0.96	0.55	213.18
211 Å	0.96	0.97	0.56	89.71
304 Å	0.92	0.95	0.47	70.84
335 Å	0.96	0.98	0.55	7.91
1600 Å	0.99	0.99	0.80	21.67
1700 Å	0.99	0.98	0.87	220.28
B_x	0.52	0.05	0.01	17.62
B_y	0.37	0.04	0.01	16.58
B_z	0.84	0.08	0.02	5.24

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to **[Yes]**, **[No]**, or **[N/A]**. You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? **[Yes]** See Section ??.
- Did you include the license to the code and datasets? **[No]** The code and the data are proprietary.
- Did you include the license to the code and datasets? **[N/A]**

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
 - (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? **[Yes]**
 - (b) Did you describe the limitations of your work? **[Yes]**
 - (c) Did you discuss any potential negative societal impacts of your work? **[N/A]**. Unclear what negative impacts there are here.
 - (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? **[Yes]**
2. If you are including theoretical results...
 - (a) Did you state the full set of assumptions of all theoretical results? **[N/A]**
 - (b) Did you include complete proofs of all theoretical results? **[N/A]**
3. If you ran experiments...
 - (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? **[No]**. This space is very crowded, we will publish the code once we have submitted to a journal. More than happy to discuss this point.
 - (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? **[Yes]**
 - (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? **[N/A]**
 - (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? **[Yes]**
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
 - (a) If your work uses existing assets, did you cite the creators? **[N/A]**
 - (b) Did you mention the license of the assets? **[N/A]**
 - (c) Did you include any new assets either in the supplemental material or as a URL? **[N/A]**
 - (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? **[N/A]**
 - (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? **[N/A]**
5. If you used crowdsourcing or conducted research with human subjects...
 - (a) Did you include the full text of instructions given to participants and screenshots, if applicable? **[N/A]**
 - (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? **[N/A]**
 - (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? **[N/A]**