COSC 440 Exoplanet Project Report

Problem

In part 2, a 2D convolutional autoencoder is trained to denoise the 2D star images (target pixel files). However, the 2D autoencoder doesn't reduce much background noise, smooths over transits and creates lower signal-to-noise (SNR) metric. This is because the noise exists mostly between the image frames. Hence the 2D autoencoder fails to capture the important features of the training data by just learning through the content within the images.

Propose

To solve this, in part 3, we convert each sequence of 2D target pixels files to a 1D lightcurve flux timeseries, and implement a 1D convolutional autoencoder to denoise windows of the flux timeseries so that information between frames and timestamps could be learned. We expect the trained 1D convolutional autoencoder to produce higher SNR metric and reduce more background noise while preserving the important transits.

Network

The 1D convolutional autoencoder consists of an encoder and a decoder. The encoder is made of several 1D convolutional layers that convert the input noisy flux timeseries (lightcurve) into an embedding representation. Max pooling layers are added between the convolutional layers to help extract sharp features, reduce variance and computations. The decoder effectively mirrors the architecture of the encoder and reconstructs the original lightcurves from the embedding representation by transposing the 1D convolutions. The architecture is illustrated in *Table 1.0* below.

Table 1.0: 1D Autoencoder Architecture

Layers	Dimension (batch_size, window_size, channels)
`input`	`(10, 2048, 1)`
`conv1D_1`	`(10, 1024, 10)`
`maxpool1D_1`	`(10, 512, 10)`
`conv1D_2`	`(10, 256, 10)`
`maxpool1D_2`	`(10, 128, 10)`
`conv1D_3 (latent layer)`	`(10, 64, 1)`
`conv1DTranspose_1`	`(10, 128, 10)`
`conv1DTranspose_2`	`(10, 256, 10)`
`conv1DTranspose_3`	`(10, 512, 10)`
`conv1DTranspose_4`	`(10, 1024, 10)`
`conv1DTranspose_5 (output)`	`(10, 2048, 1)`

Method

The training dataset contains 30 different flux timeseries split in 9:1 for training and validation. Another set of 3 different lightcurves is used to test the final trained model. To prepare this training dataset, for each star, its sequence of target pixel files is converted into a 1D timeseries of flux measurements. Each flux timeseries is extra padded with 0s (darkness) to be able to reshape into the training window size. The flux values are also normalized between 0 and 1 using the MinMax formula. Windows of the flux timeseries are then shuffled to avoid memorization. Finally, these windows of the flux timeseries are passed onto the 1D autoencoder to be trained in batches over a number of epochs. The total loss is computed using the validation flux timeseries after each epoch to measure the performance of the autoencoder to tune the training hyperparameters, hence adjusts the learning process to minimize the difference between the original lightcurve and the reconstructed one. Since the dataset is small and the loss converges pretty quickly, small epoch size of 25 and batches size of 10 are intentionally chosen to prevent overfitting. The training window sizes are discussed in Result point 2 below.

Result

1. 2D VS 1D Autoencoder

With the same training and test dataset, the 1D autoencoder-trained model successfully removed most of the noise while preserving the transit depth, which can be seen from *Figures 1.1-1.4* below. There is a significant increase in SNR after converting the 2D network to 1D (*Table 1.1 &1.2, Figure 1.1.1 &1.2.1*). Using HAT-P-11 as an example, the lightcurve generated by the original target pixel files has an SNR of 39, while our 1D network on average produces an SNR of 120, and it is also higher than the Tess generated SNR 75. Although our evaluation metric is SNR, we can also note (*Table 1.2*) that the chi2 value has decreased significantly from original target pixel files and 2D autoencoder to 1D autoencoder.

The 1D Autoencoder also trains much faster than the 2D one, with a final loss under 60 most of the time.

Table 1.1 SNR: Original VS 2D VS 1D VS Tess

SNR	Original target pixel file	2D Autoencoder	1D Autoencoder	Tess
HAT-P-11	39.7834	5.5241	120.1229	75.6941
HAT-P-14	58.5726	11.0125	161.8674	81.3520
HAT-P-2	74.8542	6.9746	150.9772	141.9700

Figure 1.1.1 SNR: Original VS 2D VS 1D VS Tess

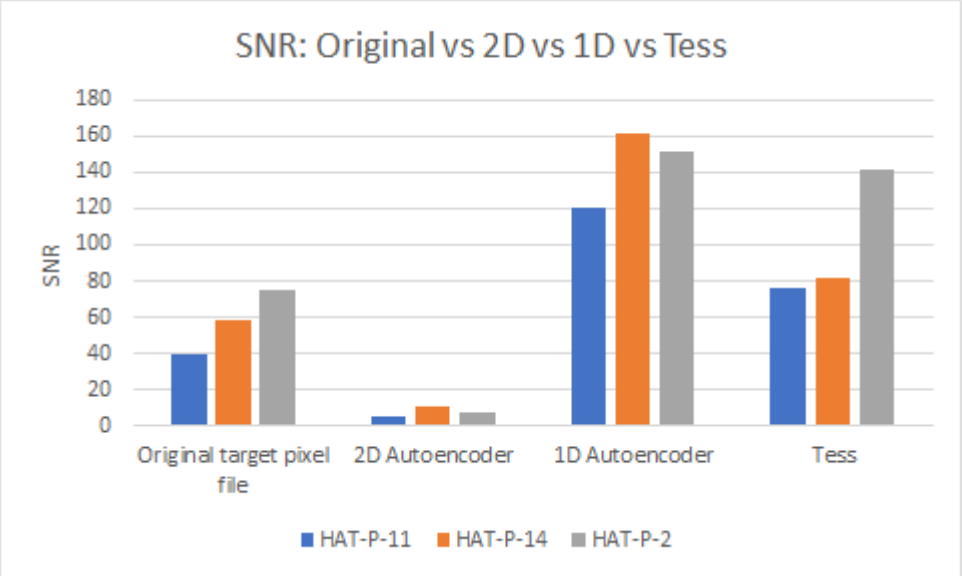


Table 1.2 chi2: Original VS 2D VS 1D VS Tess

chi2	Original target pixel file	2D Autoencoder	1D Autoencoder	Tess
HAT-P-11	16865.4900	18190.1397	8293.1048	13070.8686
HAT-P-14	14171.3196	17215.4462	5542.5304	11895.2196
HAT-P-2	13584.9606	18097.1547	6999.6032	4970.3757

Table 1.2.1 chi2: Original VS 2D VS 1D VS Tess

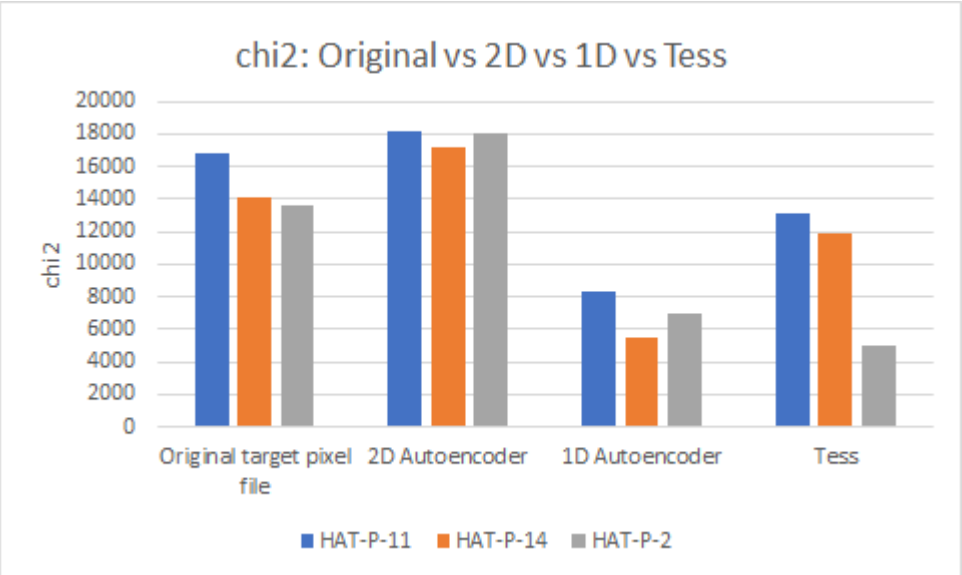


Figure 1.1 Original target pixel file generated lightcurve

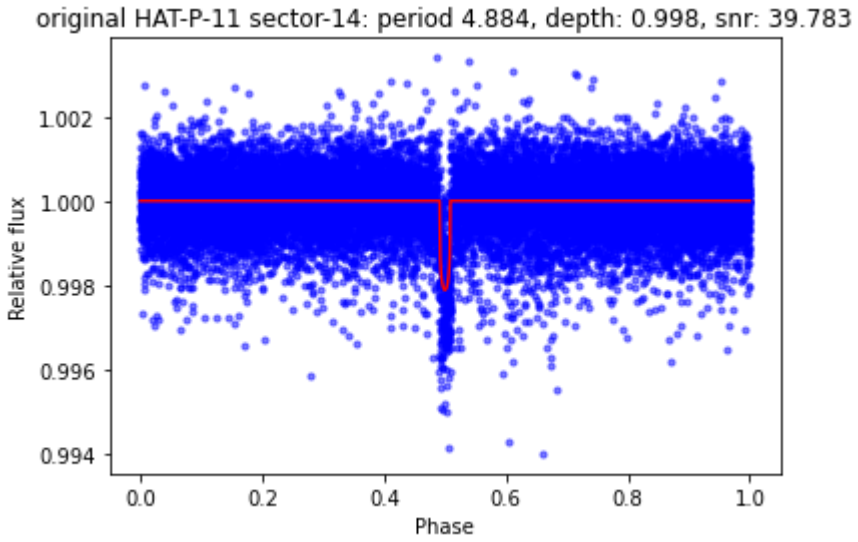


Figure 1.2 2D Autoencoder generated lightcurve

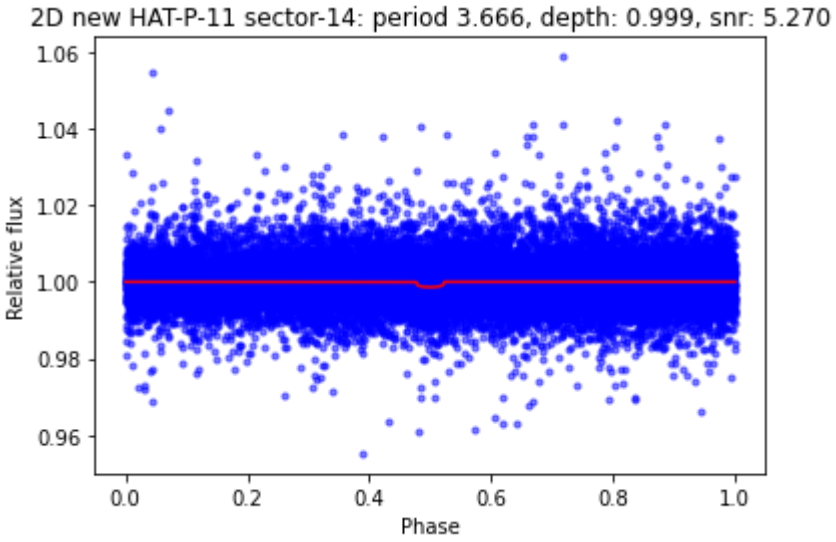


Figure 1.3 1D Autoencoder generated lightcurve

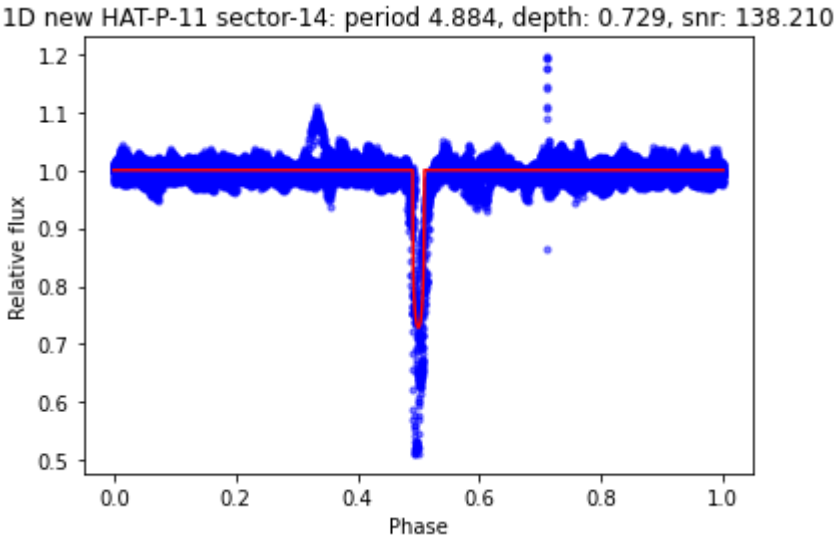
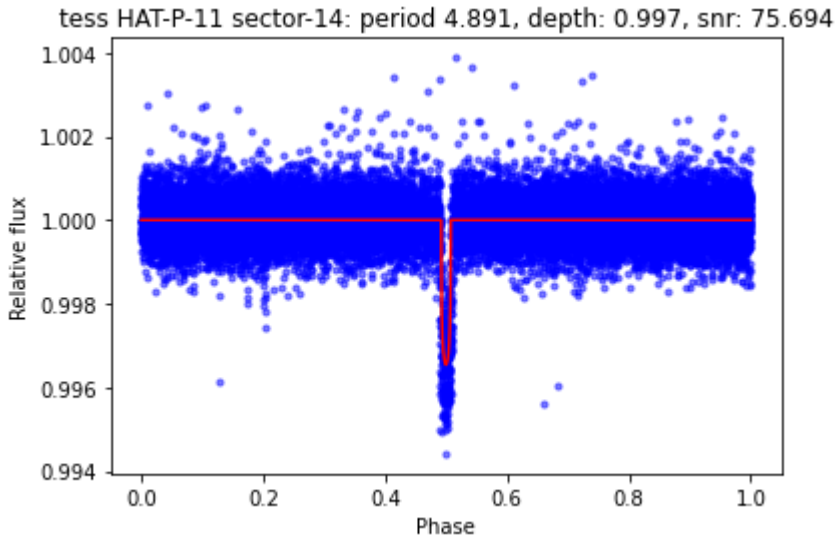


Figure 1.4 Tess generated lightcurve



2. Window size vs SNR

With the same training and test dataset, same batch size and epochs, 6 different window sizes 32, 128, 512, 1024, 2048, 4096 were experimented on the refined 1D network in 5 runs as illustrated in *Table 2.1 & 2.2*, *Figure 2.1.1 & 2.1.2 and Figure 2.1-2.6* below.

The small window size 32 performs poorly, probably because it is not able to capture enough information between timesteps, which falls into the same crisis that the 2D network has faced. Intermediate window size 128 and 512 sometimes could face the same bottleneck as the small sizes hence the performance is average. The SNRs generated by size 1024-4098 fluctuate quite a bit. This may be due to the fact that large window sizes sometimes could contain too much information for the autoencoder to digest thus an unstable learning performance.

Hoping to give the small window sizes a better variety of samples, the batch size was then increased from 10 to 30 and experimented with the 6 window sizes again. However, the SNR results were rather similar to previous. Therefore, it could be deduced that the window size plays a more significant role in this training scenario. This might be explained as most of the information is stored horizontally between the timesteps, not vertically between batches. Window size 2048 was eventually chosen for our 1D network according to the experimental results. However, given its instability, it's highly likely that it will also create lower SNR metrics below 100 during training. The combination of window size 512, batch size 30 and epoch size 25 might be a safer choice if want a more stable performance.

Table 2.1 Window Size VS SNR average 5 runs

window_size /SNR	32	128	512	1024	2048	4096
HAT-P-11	98.1572	106.4019	107.4195	104.7939	121.1229	108.2273
HAT-P-14	136.6755	142.4356	145.2930	140.6513	161.8674	152.3139
HAT-P-2	91.1075	125.5436	129.1399	127.8810	150.9772	136.7945

Figure *2.1.1 Window Size VS SNR average 5 runs*

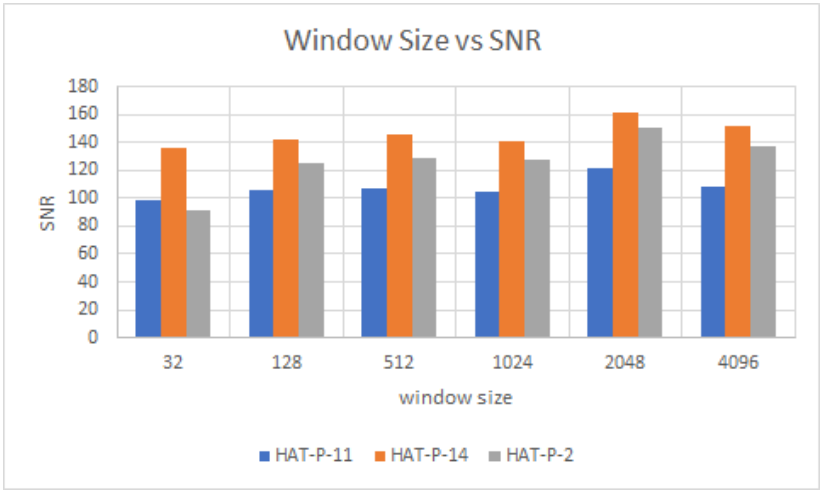


Table 2.2 Window Size VS SNR range HAT-P-11

window size	SNR range on HAT-P-11(5 runs)
32	97 - 101
128	100 - 107
512	101 - 109
1024	97 - 115
2048	109 - 138
4096	96 - 112

Figure 2.2.1 Window Size VS SNR range HAT-P-11

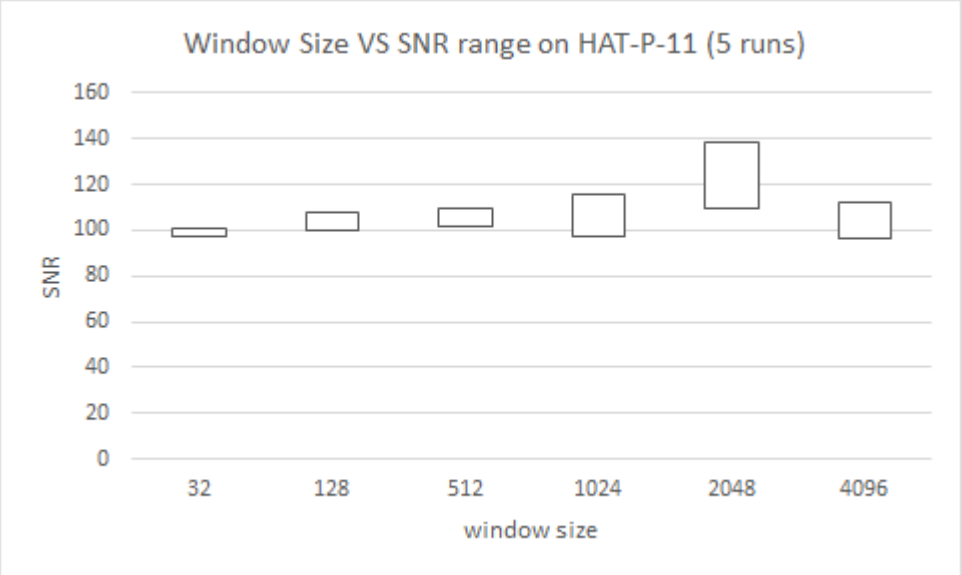


Figure 2.1 Window Size 32 with batch size 10 and epochs 25 on 1D network

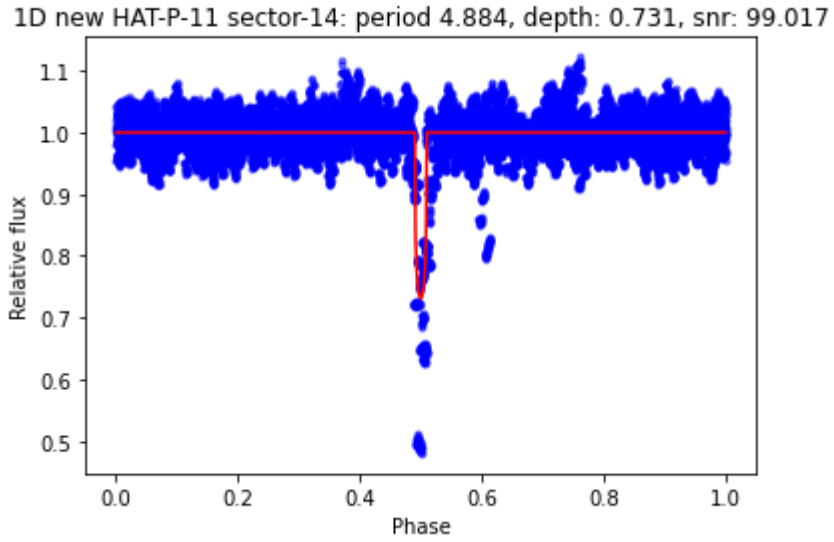


Figure 2.2 Window Size 128

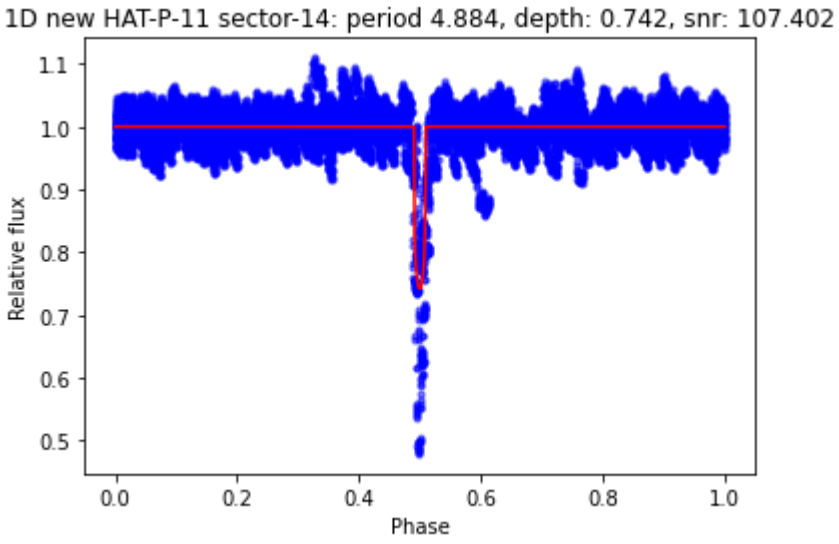


Figure 2.3 Window Size 512

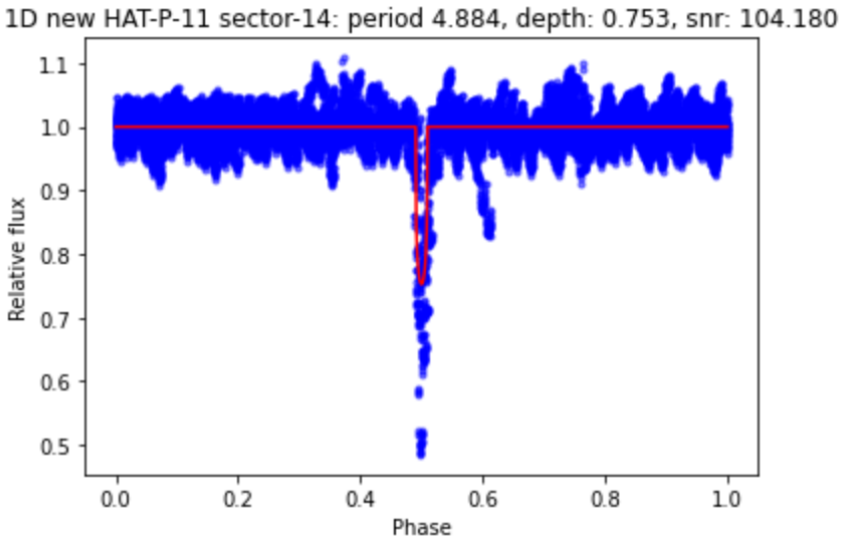


Figure 2.4 Window Size 1024

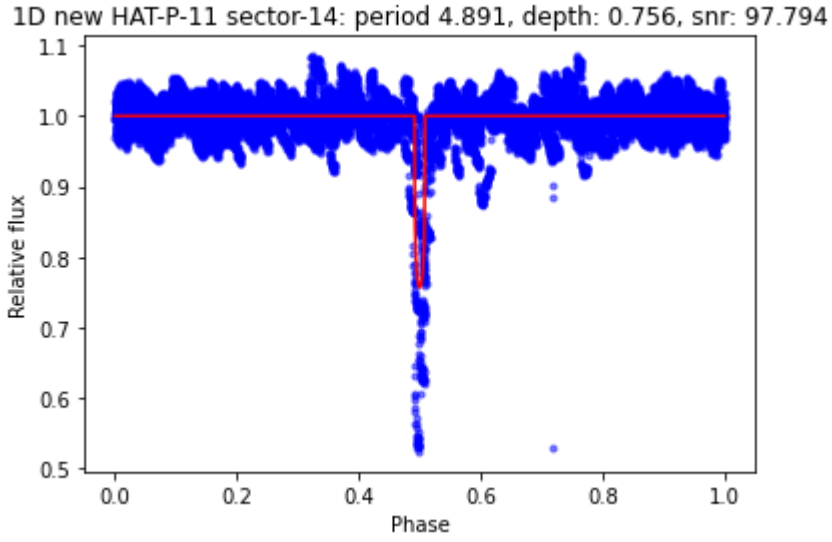


Figure 2.5 Window Size 2048

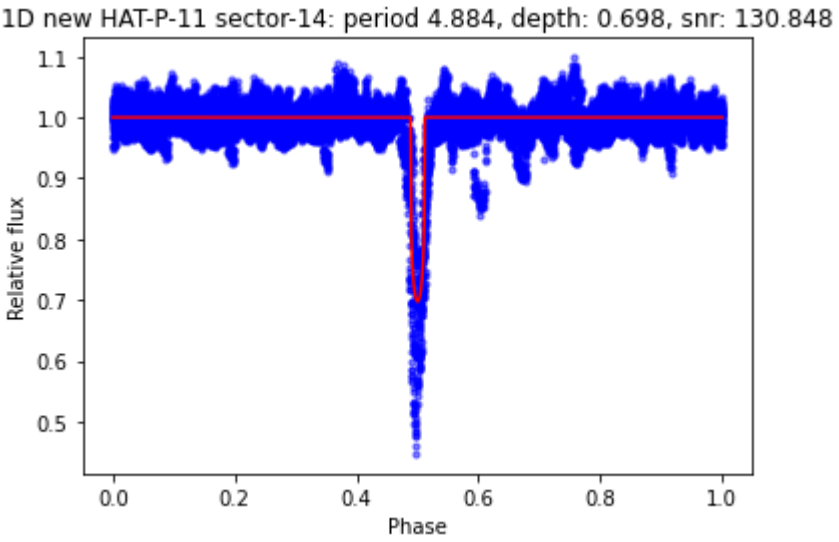
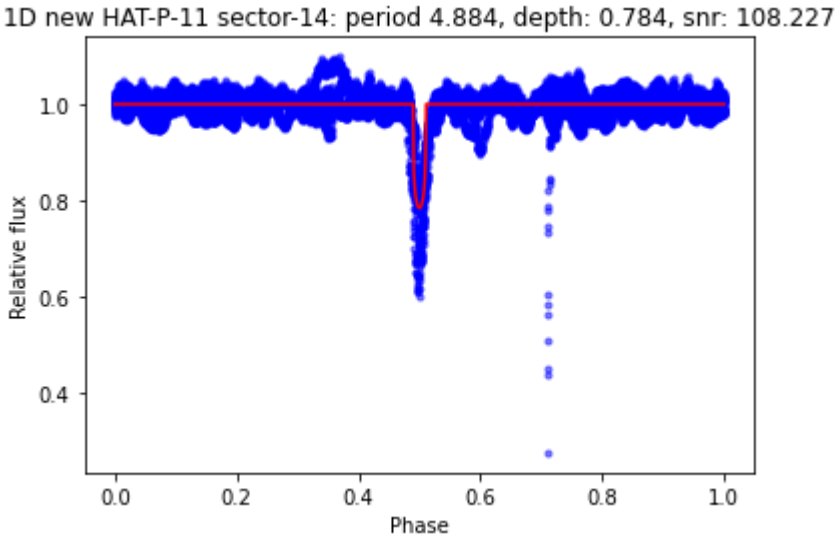


Figure 2.6 Window Size 4096



Conclusion

From the results above, we can conclude that compared with the lightcurves generated by the 2D Autoencoder and the original noisy target pixel files, our refined 1D Autoencoder effectively removed most of the background noise while being able to preserve and classify the import features like transits, along with a significant increase in SNR, decrease in chi2 and a robust training performance.

In this work, we applied an unsupervised 1D Autoencoder network to the analysis of supernovae lightcurves but it can also be applied to the analysis of other transient and timeseries objects, performing tasks such as denoising, forecasting, and anomaly detection.

While the current 1D Autoencoder has demonstrated the proposed hypothesis, the network has its limitations. The training and test datasets are small hence insufficient coverage of samples and lower precision in estimation. Furthermore, the network could also be improved by performing variational or generative sampling on the latent layer and incorporating Kullback–Leibler divergence into our loss function to optimize the reconstruction. These would be the next steps for this project.

Code

https://colab.research.google.com/drive/1AseY_QzJQwLiE1yOoNNWsP-1oZb0p26x?usp=sharing

Reference

https://arxiv.org/pdf/1901.01298.pdf

Student ID: 97245310
Name: Yuezhang Zhu