Supplemental Document

Optics Letters

Performance of a U-Net-based neural network for predictive adaptive optics: supplement

JUSTIN G. CHEN, VINAY SHAH, AND LULU LIU*

MIT Lincoln Laboratory, 244 Wood St., Lexington, Massachusetts 02421, USA
**Corresponding author: lulu.liu@ll.mit.edu*

Supplement DOI: https://doi.org/10.6084/m9.figshare.14396045

Parent Article DOI: https://doi.org/10.1364/OL.422656

Performance of a U-Net-Based Neural Network for Predictive Adaptive Optics – Supplemental Information

JUSTIN G. CHEN,[1] VINAY SHAH,[1] AND LULU LIU[1,*]

[1] *MIT Lincoln Laboratory - 244 Wood St. Lexington, MA 02420*
[*] *lulu.liu@ll.mit.edu*

1. AEOS Open-Loop WFS Data

We were provided with 9 s of open-loop wavefront sensor data from the AEOS telescope. In this section we provide some more details about this data.

Fig. S1 shows the azimuth, elevation, and range tracks of the LEO target for the approximate duration of the data acquisition period.

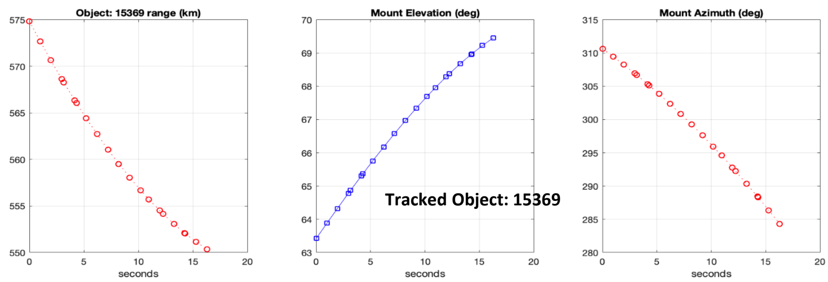

Fig. S1. Cosmos 1606 track in azimuth, elevation, and range for duration of data acquisition period.

Fig. S2 shows the fit to extract an effective Greenwood frequency of 140 Hz. The effect of global piston subtraction can be seen to remove power at lower temporal frequencies.

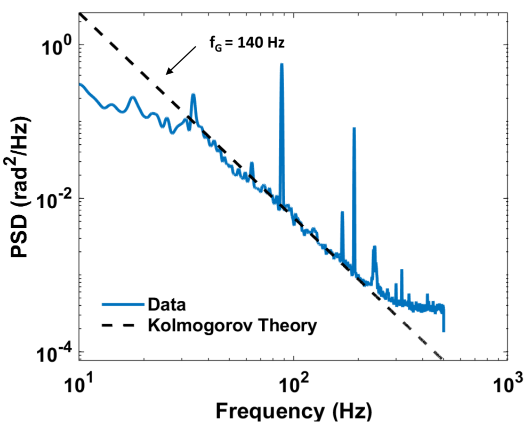

Fig. S2. PSD of per-pixel SHWFS time series data averaged over all 32×32 pixels. Fit to Kolmogorov theory yields a Greenwood frequency of 140 Hz under slewing conditions.

Fig. S3 shows the fit to extract an effective Fried parameter at observational wavelengths. Custom light propagation software was used to simulate the frame-by-frame focal plane image of a point-source with the captured wavefront perturbations. These frames are averaged together to obtain an approximate PSF, which is then corrected based on exposure length.

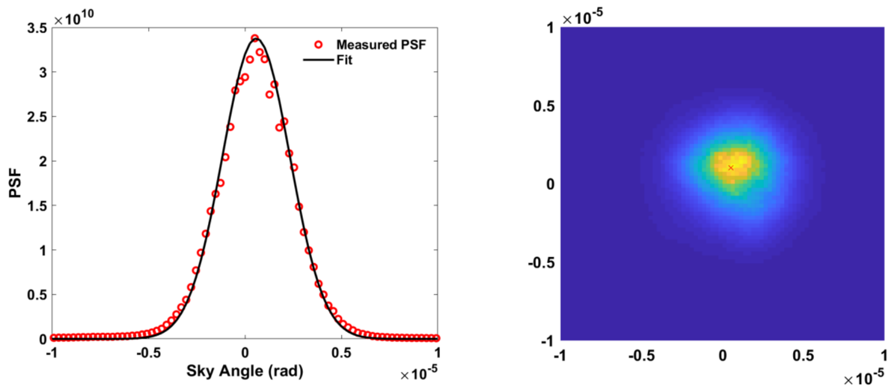

Fig. S3. Average PSF simulated over 1000 frames of AEOS wavefront data fit to a gaussian seeing profile.

Remaining atmospheric parameters were estimated by fitting the PSD and temporal structure function of a simulated AEOS atmosphere to that of the real data (see Fig. S4). Doing so, we were able to additionally estimate that the von Karman turbulence outer scale was approximately 20 m, the isoplanatic angle was 20 μrad, the per-frame wavefront error was approximately $\lambda/15$, and the autoregressive scaling parameter (α) was approximately 0.997, indicating some deviation from frozen-flow characteristics [1].

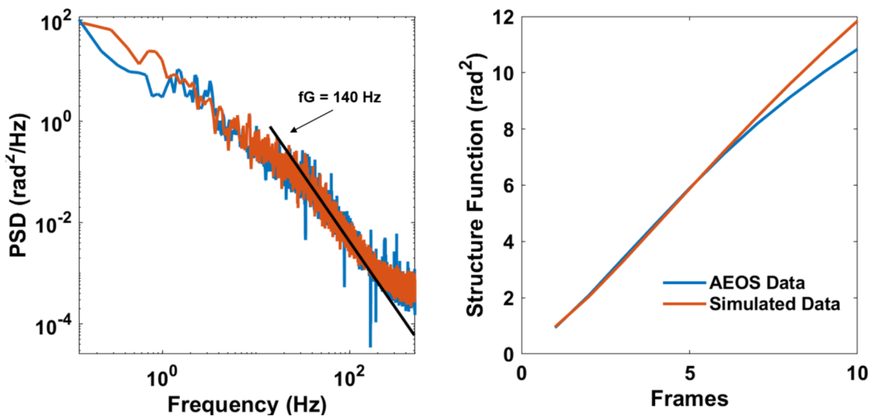

Fig. S4. A family of simulations were run with a variety of atmospheric parameters with fixed r_0 and f_G.

2. Atmospheric Phase Screen Simulations

The simulation begins by creating oversized phase screens for each of the user specified layers of the modeled atmosphere. The layers are sized according to the velocity of each layer and the total simulation time. Each layer is randomly generated and follows a Von Kármán spatial power spectrum [2]. The layer parameters (layer heights, r_0, velocities) are randomly selected from a distribution of reasonable values and constrained to fit a Hufnagel-Valley C_n^2 profile.

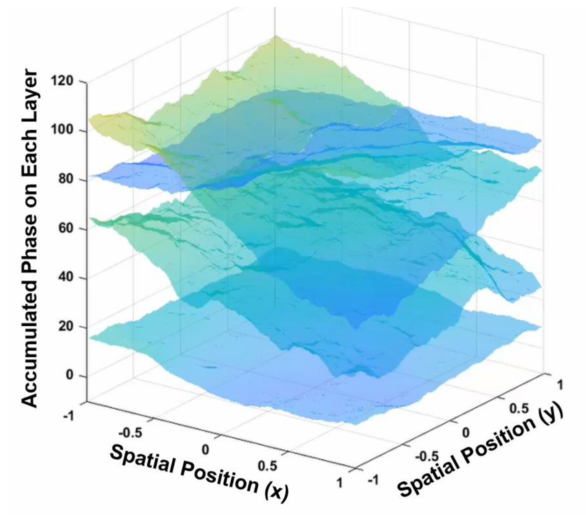

Fig. S5. Depiction of frozen flow simulation with discrete phase screen layers. Layers are displaced vertically for display. The displacement does not represent physical height of each layer though their order is preserved.

The user specifies the number of subapertures (wavefront sensing elements) as well as their 2-dimensional locations. At every time instance, for each subaperture, the corresponding region of each phase layer is sampled, stacked, and summed over all atmospheric layers, creating a phase profile across the extent of the subpaerture. A plane is fit to the summed phase measurement: we assign its mean value to the piston for that subaperture, and its slope in X and Y to the atmospheric tilts across the subaperture.

Since each time instance is independent from the others, this simulation is considered “embarrassingly parallel” and lends itself to easily run on a supercomputer. Thus, the simulation is divided up in time, with each compute node computing the simulated measurements for its own time chunk [3].

Variation of the telescope slew rate was responsible for much of the variability in performance of the neural network in testing, so we varied the telescope slew rates for the 10 simulated datasets we use for training and evaluation purposes. Figure S6 shows simulated slew rates of the 10 simulated datasets used in training and a division intwo two separate groups. The Group 1 files are held out as a "Dissimilar Group" of files as shown in the Simulated Data Results Figure in the main paper, and used to train the network that’s tested on the Focus Dataset which is a part of the Group 2 files.

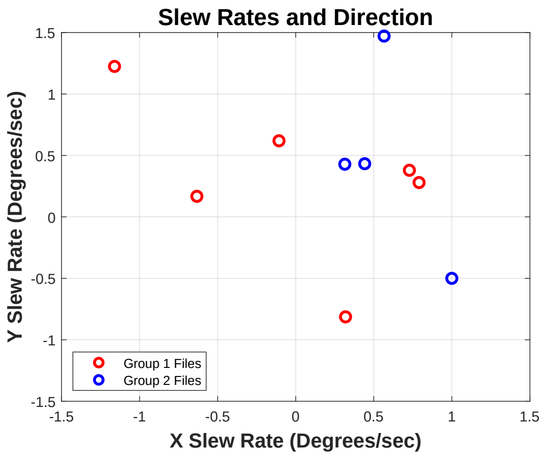

Fig. S6. Random slew rates and directions of simulated data used in training the neural networks.

3. Spatial Error Distribution

In the main paper we report the results in figures as phase error vs. the predict-ahead frame number, representing the temporal dependency of the prediction performance of our neural network as we try and predict further in the future. Another way we can look at the performance of the neural network would be the spatial dependency. Figure S7 shows the spatial distribution of the mean-squared error in phase for the neural network trained on the on-sky AEOS data and simulated datasets, and fine tuned on the AEOS d ataset. The edges of the aperture have the highest error, while the pixels towards the center have lower error. Additionally, the top right edge of the aperture has lower error than the other areas around the aperture matching the direction of the apparent wind caused by the slewing of the telescope (bottom left to top right). This all reinforces the main idea that because of the approximately frozen-flow nature of the atmosphere, we are able to use data from previous frames to predict the phase values in future frames. There is more information in the center and the downwind side of the apparent wind from slewing, and thus the error is lower in those locations.

4. Data Augmentation and NN Architecture

Frozen-flow atmospheric simulations require a nontrivial amount of computational resources and time in order to run. Data augmentation is something that can be done easily at runtime with negligible additional resource commitments. It expands the training data, and in this case directly expands the effective number of wind and slew directions.

And as Fig. 4 in the main text shows, we ultimately found that the addition of a diversity of simulated datasets to the training set, beyond the data augmentation step, did little to improve performance. This is likely due to the fundamental similarities, and therefore redundancies, in all simulated data sets, in that they are a fixed integer number of frozen turbulence layers flowing at constant velocities, and speaks to the limitations of some of these models.

5. Network Architecture Ablation Study and Comparison

We implemented several different neural networks for the purpose of serving as an ablation study and also to some other different architectures of neural networks. The ablation study was used to demonstrate the performance of the U-Net architecture over simple convolutional

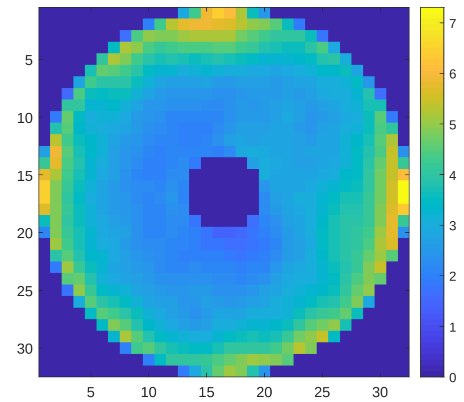

Fig. S7. Spatial phase MSE in radians^2 on future frame 4 for the AEOS dataset, as calculted from the results from the fine-tuned AEOS neural network. Note the lower error at the top right which is the leeward direction of the apparent wind, opposite the direction of slewing. Also in general there is worse performance at the edges of the visible aperture because there is less information to predict the future phases with. There is lower error towards the center of the aperture.

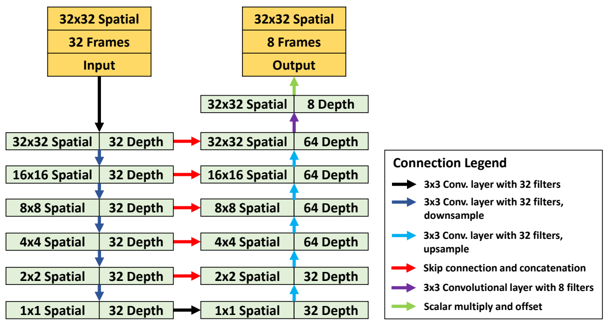

Fig. S8. Diagram of the U-Net architecture used for predictive AO. Arrows represent connections. Boxes report the spatial and depth size of the features at each layer. Not shown are the leaky ReLU and batch normalization steps used for each layer [4].

Fig. S9. An example of *data augmentation*, which we use to diversify our datasets.

neural networks. The goal of an ablation study is to remove certain features of the neural network architecture to see how the performance either suffers or improves as a result. The other architecture of neural networks we implemented were recurrent neural networks using Long Short Term Memory (LSTM) cells [5]. One network follows the implementation used in [6] and we call it a VectorLSTM because the spatial dimensions are collapsed into a vector, and for the other network we use convolutional LSTM cells as in [7] and implement a similar network. The difference between these networks is that the Vector LSTM network loses explicit spatial connections, while the ConvLSTM network has both spatial convolutional structure as well as temporal structure with the LSTM cells.

The networks were trained on the full 10 files of simulated data and evaluated on the full set of simulated data. 5 different training runs were used for each network to get an idea for training and model variability, and fine tuning was not used. The metric being compared in Figure S10 is the average Mean Squared Error in phase (radians^2) when compared to the ground truth frames, over an average of all the simulated data files and all the future predicted frames. On the x-axis we plot the total number of trainable parameters as a simple representation of the size of the neural network. We also compare the Mean Square Error to the average evaluation time per frame as shown in Figure S11. We run the networks through our 1000 frame evaluation set, 1000x times and use that to calculate an average "per-frame" evaluation time, which really represents a single instance of using 32 history frames to predict 8 future frames.

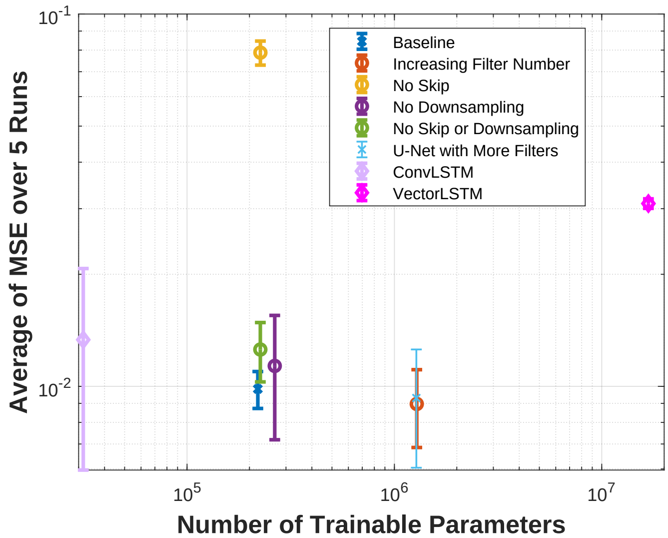

Fig. S10. Neural network architecture and ablation study with the U-Net described in the paper as the baseline.

The baseline U-Net network has some of the best performance amongst all of the networks. One deviation we make from the standard U-Net is that we fix the channel number or depth of the network as we downsample for simplicity, the *increasing filter number* network represents the results if we were to increase the depth of the network as we downsample spatially, and it shows a small but modest improvement, at the expensive of an order of magnitude more trainable parameters and much greater model size. If we use the same baseline U-Net network but just add more filters (depth) all around, performance is very similar. Performance is greatly hindered if we ablate the skip connections, showing how important they are for passing information at different spatial scales from the "input" end of the network to the "output" end. The performance hit is less pronounced if we have neither skip connections nor downsampling, as there isn't severe bottlenecking in the network without downsampling. Without downsampling there is a small performance hit compared to baseline. The evaluation times for all the neural networks in the ablation study are relatively similar as seen in Figure S11, all close to 0.3 milliseconds for processing one "frame" of data.

Comparing the baseline U-Net convolutional neural network to the recurrent LSTM architectures, the VectorLSTM, and ConvLSTM, the performance of the ConvLSTM and the baseline U-net are somewhat similar (the ConvLSTM results have one poor performing outlier amongst

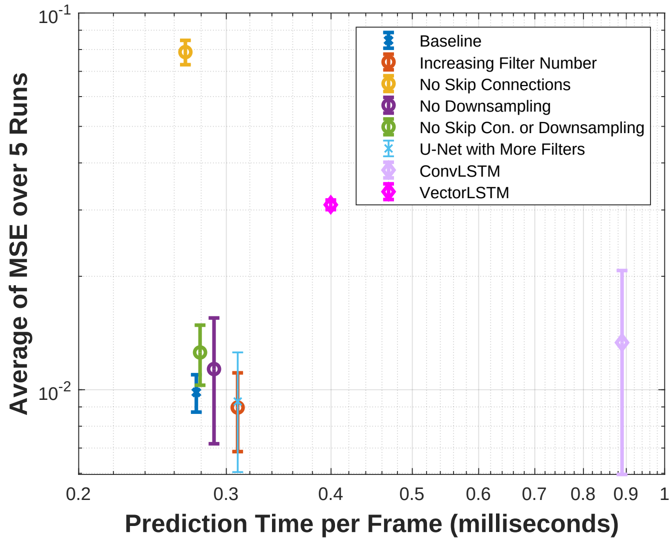

Fig. S11. Performance vs. evaluation time for different neural network architectures including an ablation study on the U-Net baseline network.

the 5 training runs), however the vector LSTM underperforms. Interestingly enough, compared to the baseline network, the ConvLSTM was able to achieve similar performance with relatively few trainable parameters, while the VectorLSTM has quite a few but much worse performance. The variability in the ConvLSTM network is likely due to the limited number of trainable parameters used, which is a limitation of the relatively large resulting model size when additional filters were used. When comparing evaluation time however, both the VectorLSTM and the ConvLSTM perform slower, with the ConvLSTM performing likely too slowly to be used in real-time in actual implementation (0.9 ms per frame on 1000 Hz data). In training, even with the much lower number of trainable parameters, the ConvLSTM was much more time consuming to train, with each training run taking approximately 12 hours to run compared to 2 hours for the baseline U-Net network.

6. Comparison against Analytical Predictive Method

We have included also a comparison with a more "traditional" predictive AO method based on a linear least-squares (LLSq) covariance-based estimator (similar to [8]). This estimator was optimized by hand iteratively over each free parameter (i.e. the width and lookback depth of the influence kernel, the number of "training" frames included, and the noise tolerance of the matrix inversion step). Both estimators were trained and tested on the AEOS open-loop data. Figure S12 shows a comparison between results from the fine-tuned NN and this LLSq estimation method.

Funding

DISTRIBUTION STATEMENT A. Approved for public release. Distribution is unlimited.

This material is based upon work supported by the Under Secretary of Defense for Research and Engineering under Air Force Contract No. FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Under Secretary of Defense for Research and Engineering.

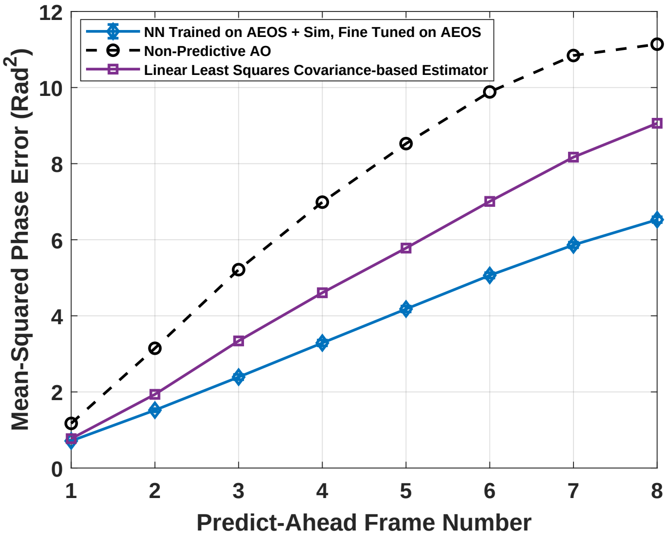

Fig. S12. AO performance comparison between the U-Net NN with fine-tuning and linear least-squares method using the on-sky AEOS dataset.

References

1. S. Srinath, L. A. Poyneer, A. R. Rudy, and S. M. Ammons, "Computationally efficient autoregressive method for generating phase screens with frozen flow and turbulence in optical simulations," Optics express **23**, 33335–33349 (2015).
2. P. Jia, D. Cai, D. Wang, and A. Basden, "Simulation of atmospheric turbulence phase screen for large telescope and optical interferometer," Monthly Notices of the Royal Astronomical Society **447**, 3467–3474 (2015).
3. A. Reuther, J. Kepner, C. Byun, S. Samsi, W. Arcand, D. Bestor, B. Bergeron, V. Gadepally, M. Houle, M. Hubbell *et al.*, "Interactive supercomputing on 40,000 cores for machine learning and data analysis," in "2018 IEEE High Performance extreme Computing Conference (HPEC)," (IEEE, 2018), pp. 1–6.
4. P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in "Proceedings of the IEEE conference on computer vision and pattern recognition," (2017), pp. 1125–1134.
5. S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation **9**, 1735–1780 (1997).
6. X. Liu, T. Morris, C. Saunter, F. J. d. C. Juez, C. González-Gutiérrez, and L. Bardou, "Wavefront prediction using artificial neural networks for open-loop adaptive optics," Monthly Notices of the Royal Astronomical Society (2020).
7. X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," arXiv preprint arXiv:1506.04214 (2015).
8. M. Lloyd-Hart and P. McGuire, "Spatio-temporal prediction for adaptive optics wavefront reconstructors," ESO Conference and Workshop Proceedings (1996).