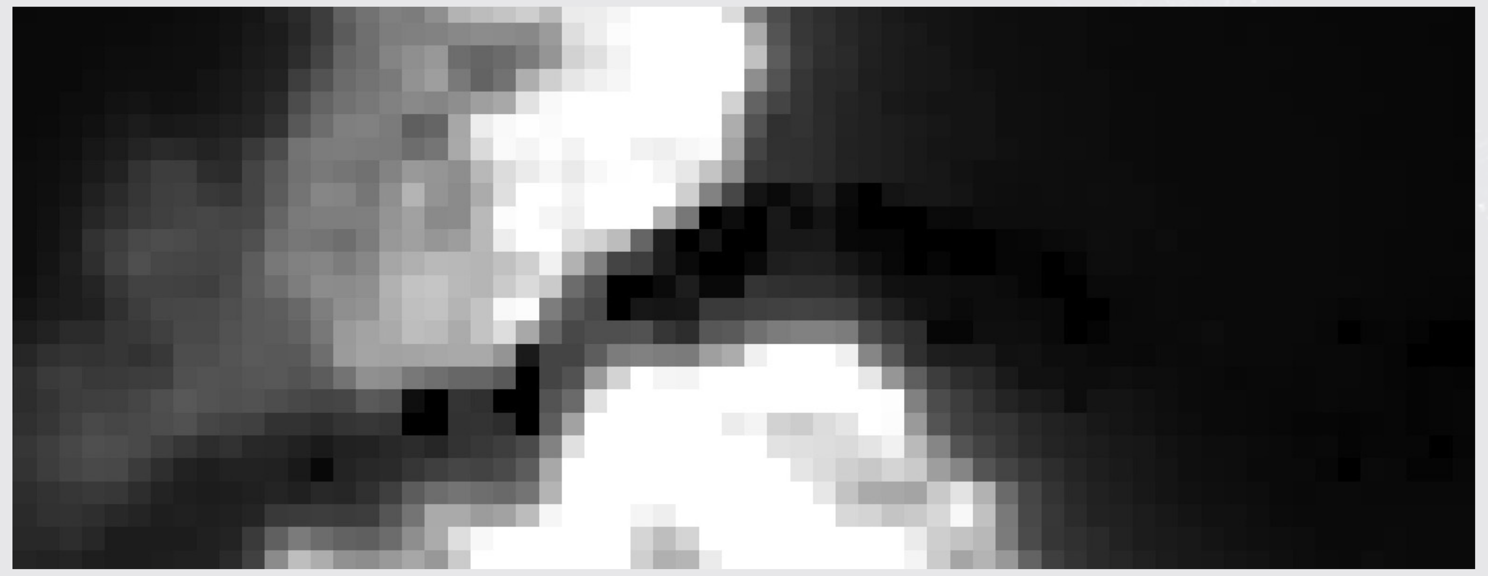
Earth at Night in HD: Predicting High-Resolutions Nighttime Radiance from Daytime Satellite Imagery

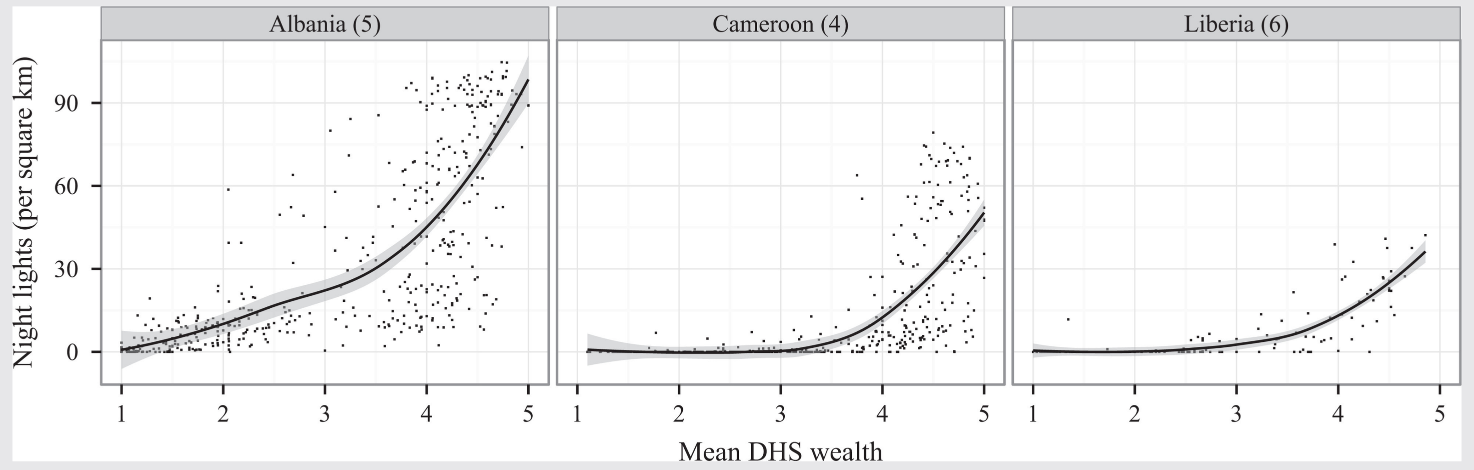
Isaiah Lyons-Galante | University of Colorado Boulder | CSCI 5922 Neural Networks | Final Project

Background & Motivation

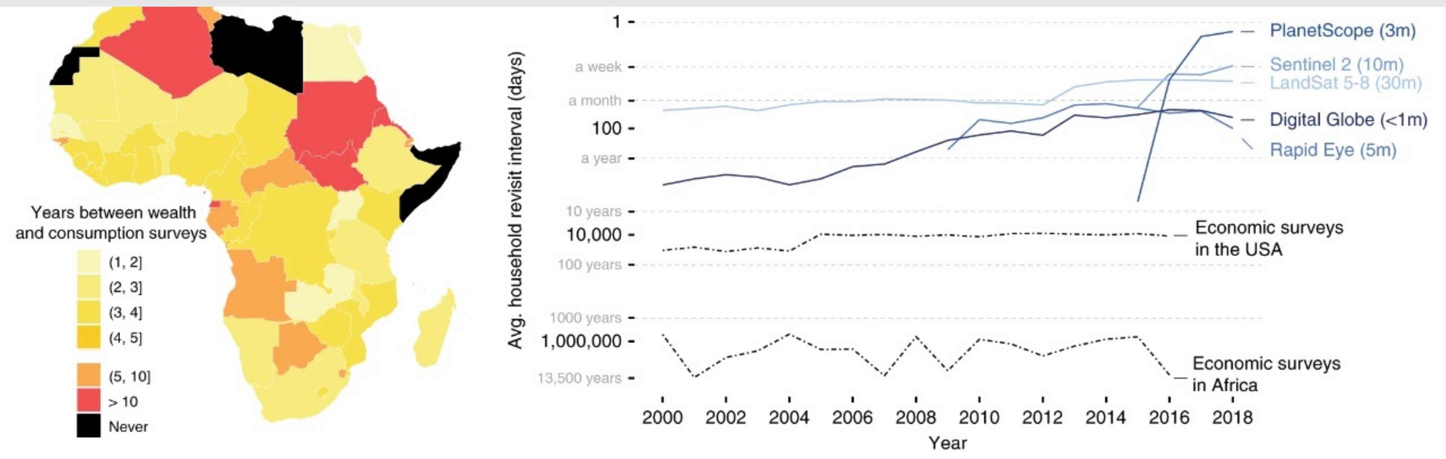
Humans have left a very visible footprint on the Earth that allows us to study populations from space. Global high-resolution datasets like Landsat[1] are available for public use through platforms like Google Earth Engine.[2]



At night, human settlements emit faint light into space that is captured nightly by the VIIRS instrument aboard the Suomi NPP satellite.[3] However, the relatively weak signals reduce the resolution to about 500 meters squared.



Research shows that night lights correlate strongly with mean household wealth as measured by ground surveys.[4] This means that night lights can serve as a proxy for measuring livelihoods. This is especially useful in Sub-Saharan Africa, where population surveys are few and far between.[5] Having higher-resolution nightlights would enable fine-grained impact assessment to see how individual communities are affected by economic development programs or world events.

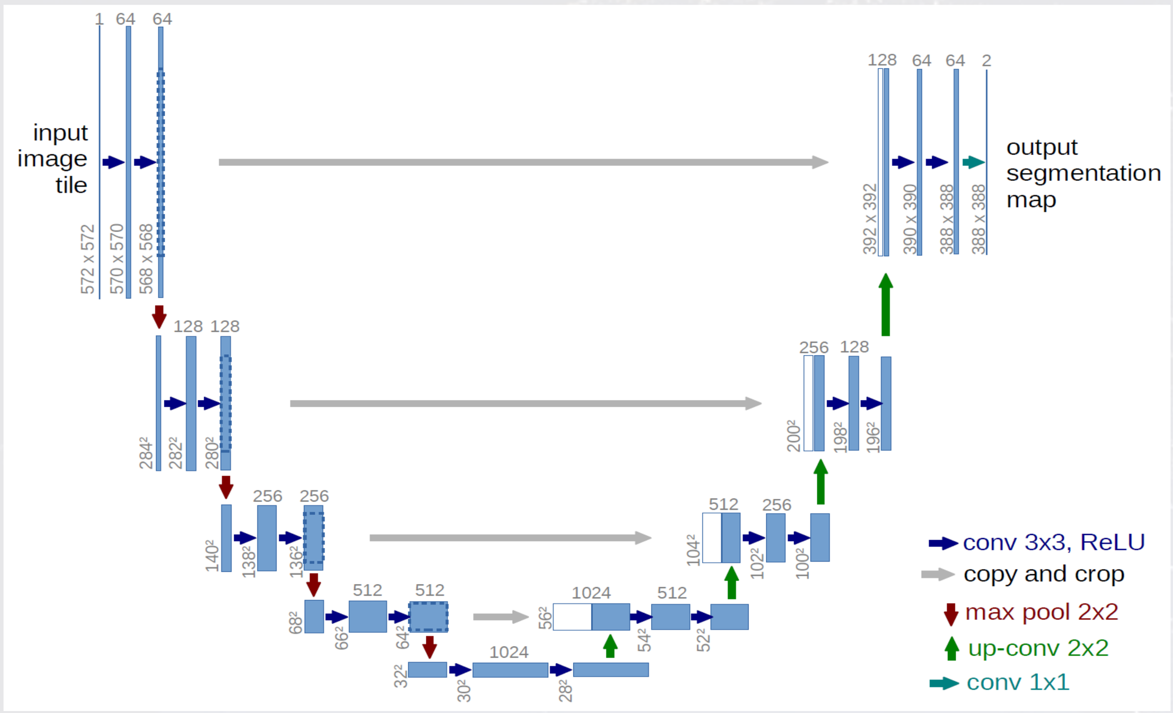


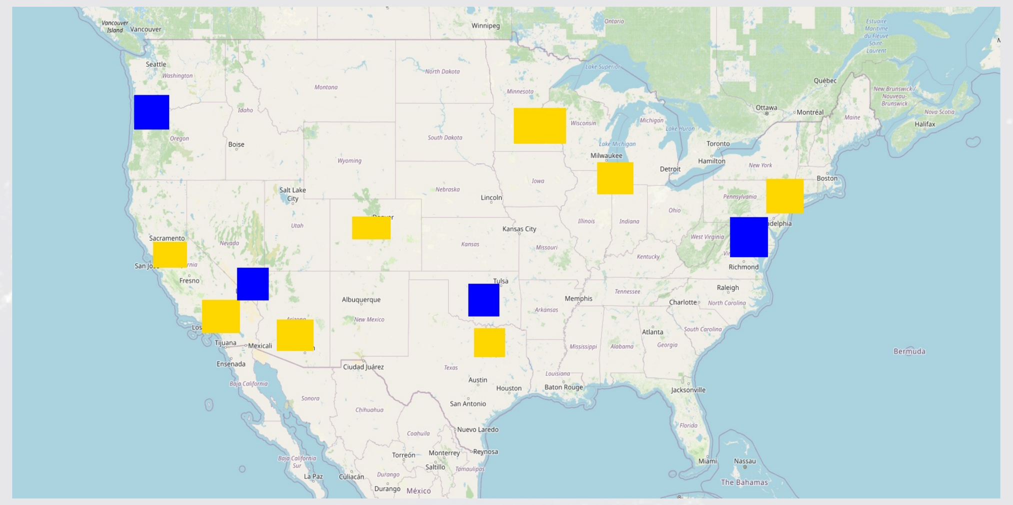
Methods

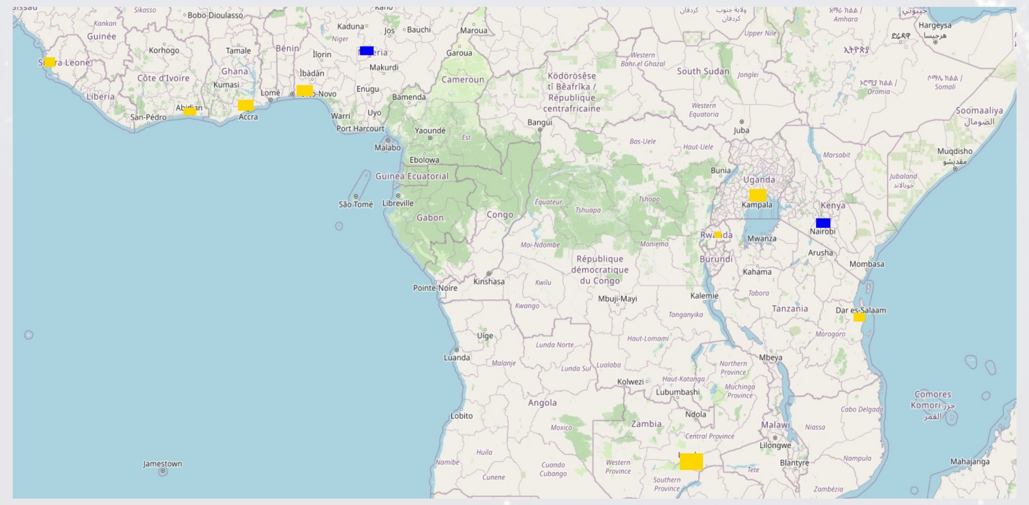
Because I are trying to predict one image from another, I can leverage a Fully Convolutional Network (FCN)[6] with no dense layers. This allow us to use input imagery of any resolution. The U-Net[7] architecture, named for the shape of the diagram, is the most popular FCN and is well suited to our task. The output layer is modified to perform regression instead of classification so that I get a continuous radiance value instead of segment class.

With the model in place, I then need to define our training data. In this project, four different training datasets were tested: both large training regions with 1000 image patch samples from each, as well as small regions with 100 image patch samples from each. Each of these were pulled from the United States and the African continent. For each, there were 8 unique training regions, and 2 evaluation regions.

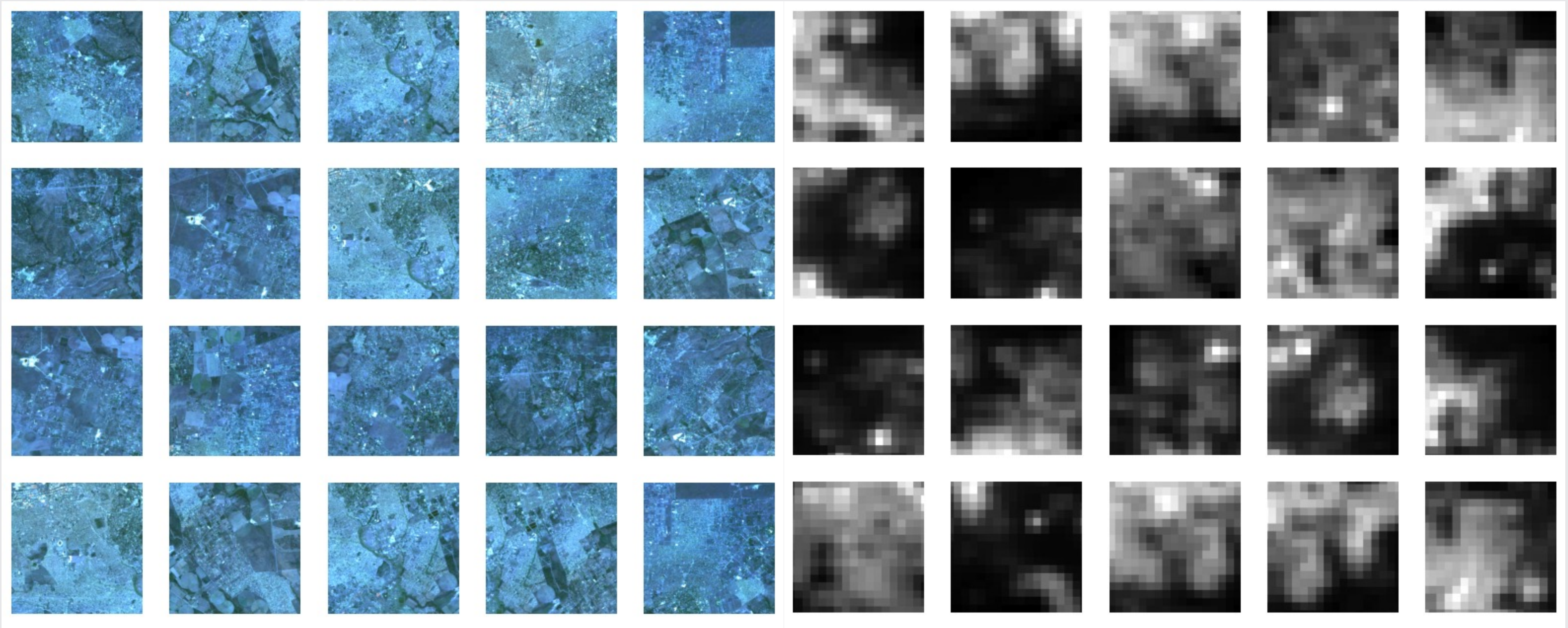
For both areas, the polygons were centered around urban areas, as these have the greatest range of nighttime radiance. For consistency, the evaluation areas were also urban areas. Because the target is to predict nightlights in under-surveyed areas, the evaluation sites were Abuja, Nigeria, and Nairobi, Kenya.







On the left, I have the large image patches from North America, and on the right from the African. Training patches are in yellow, and evaluation regions are in blue. The aim of having different continents as inputs was to test how transferrable the results are. Do well-lit areas in the US look like well-lit areas in Africa? For daytime imagery inputs, I pulled 9 bands from Landsat, 7 optical bands and 2 thermal bands. I created a cloud-masked image composite to create high-quality, cloud free input. For nightlight output, I also create a median composite of the average radiance band from stray-light corrected nightlights.[8] The aim is that by pairing each daytime image patch of 256 x 256 with the corresponding nighttime image patch, I can predict nighttime image patches of 128x128 when the original resolution was about 16x16. This is almost 60X improvement in resolution. Below on the left, I have a set of example input images from Landsat in false-color. The level of detail is significant compared with the coarse nightlight image patches on the right.



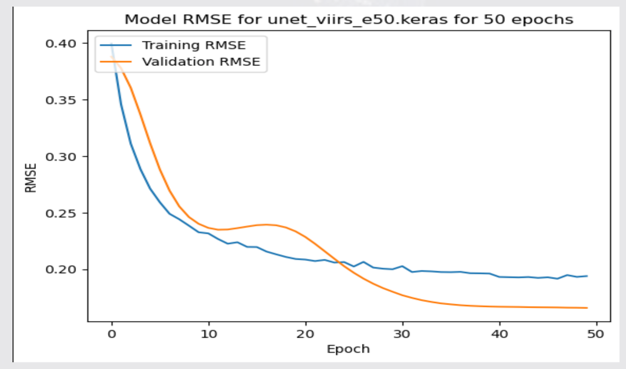
The other parameters that were experimented with were the number of epochs and the optimizer algorithm. For epochs, values between 1 and 50 were tried for most models. For the optimizer, a single experiment was run with one of the top performing models to see how it would compare. The imagery was compiled in Google Earth Engine via their Python API[9], exported to a Linux computer with a 48GB GPU. The model was built with tensorflow[10], code is available on GitHub[11].

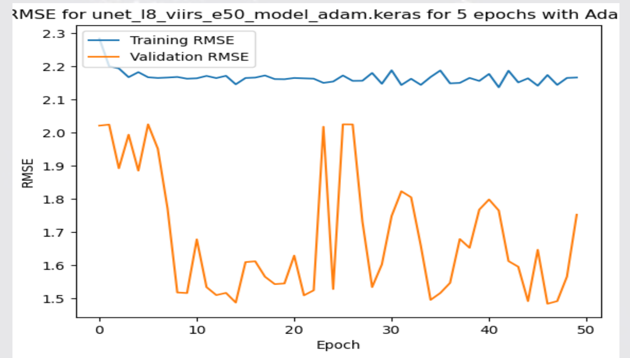
Results

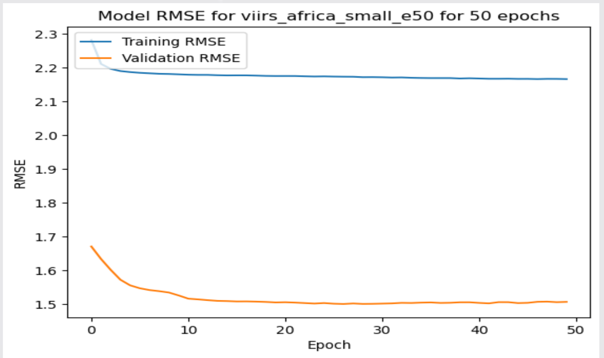
The top performing model was the one trained on the small polygons around African cities for 20 epochs using Stochastic Gradient Descent, achieving an RMSE of 1.504. Surprisingly, increasing the samples in the training dataset by 10X did not improve on the results. Another surprise is that the train RMSE was consistently lower than the validation RMSE. This is highly atypical, and suggests that there is an issue with the training strategy. The full table of results is below.

Name	Train Region Size	Train Samples	Optimizer	Epochs	Train Time	Train RMSE	Eval RMSE
viirs_africa	Large	8000	SGD	1	0:02:00	2.485	1.725
viirs_africa	Large	8000	SGD	20	0:15:00	2.184	1.545
viirs_africa	Large	8000	SGD	50	0:39:48	2.178	1.542
viirs_africa_small	Small	800	SGD	1	0:00:30	2.380	1.648
viirs_africa_small	Small	800	SGD	5	0:00:45	2.240	1.554
viirs_africa_small	Small	800	SGD	20	0:02:29	2.224	**1.504**
viirs_africa_small	Small	800	SGD	50	0:05:00	2.221	1.508
viirs_africa_small_adam	Small	800	Adam	50	Not recorded	**2.192**	1.752
unit_viirs	Small	900	SGD	5	Not recorded	2.591	1.796
unit_viirs	Small	900	SGD	10	Not recorded	2.659	1.878
unit_viirs	Small	900	SGD	50	Not recorded	2.764	1.960
viirs_usa	Large	8000	SGD	50	Not recorded	2.489	1.554

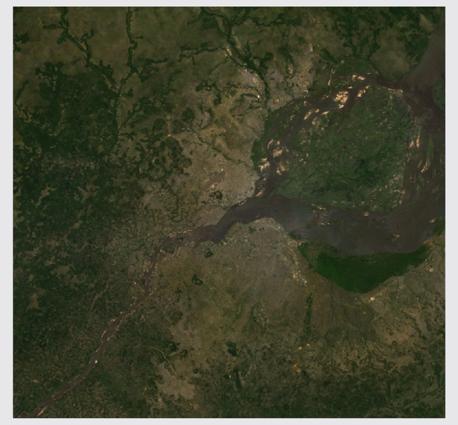
On these RMSE by epoch graphs, we have a few interesting results. (1) shows the model training and validating in the USA, the only one where validation error followed a more regular trend. (2) shows the training with the Adam Optimizer, showing quite unstable training. (3) shows the stable training of one of the top performing models. However, it reaches quite close to the minimum in just about 15 epochs, and does not improve from there.

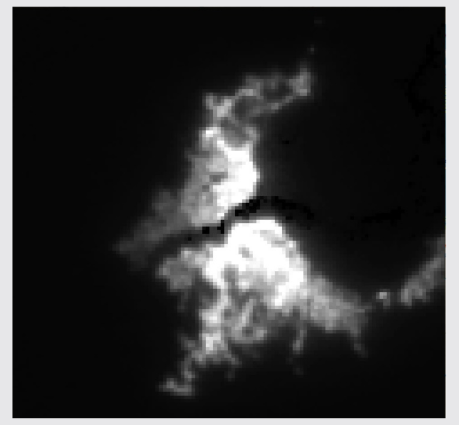


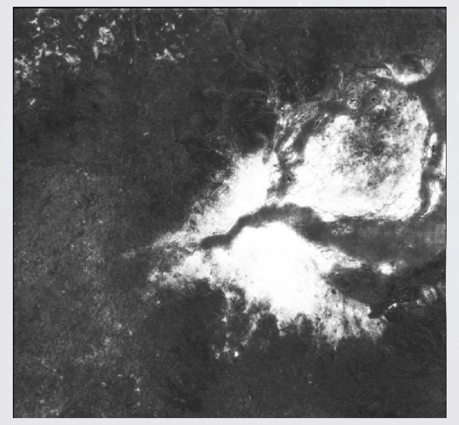


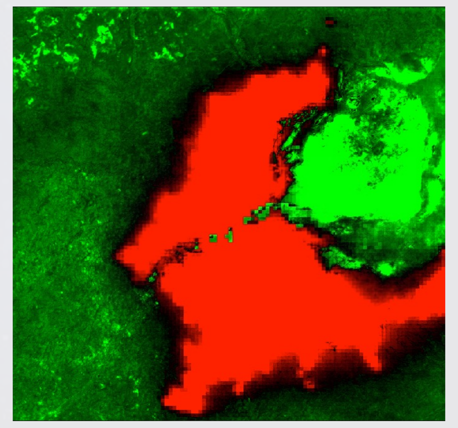


Finally, for each model, I ran a prediction pipeline for an untrained area, the twin cities of Kinshasa and Brazzaville between the Congo and the Democratic Republic of Congo for a qualitative analysis. (1) shows the true color Landsat composite, (2) shows the VIIRS nightlights, (3) shows the predicted nightlights, and (4) shows the difference between (2) and (3). On this prediction, the model is picking up the features of urbanization in the cities, but it falsely classifies the island in the river delta as well lit while it is not populated.



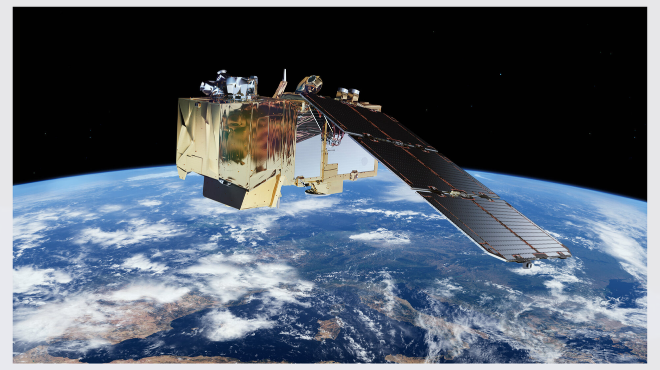






Conclusions & Future Work

1. **Training Data Size:** increasing the size of the training data did not improve model performance as expected. I need to dig further into this as extra labeled training data should improve performance.
2. **Higher Resolution Imagery:** Because the results were not as accurate as I expected, I did not get a chance to try imagery of even higher resolution such as Sentinel-2[12] or Google Earth base maps.
3. **More Epochs are Better, up to a Point:** Most models plateaued well before 50 epochs. This is likely related to the training data size. I would like to investigate further the relationship between training sample size, model parameters, batch size, learning rate, and number of epochs to come with some rules of thumb.
4. **Predicting Livelihoods:** Once I have a model that predicts nightlights with decent accuracy, I would like to fine tune the model with ground surveys of household assets to be able to assess economic status.



Footnotes:

1. Landsat 8: https://www.usgs.gov/landsat-missions/landsat-8
2. Google Earth Engine: https://earthengine.google.com/
3. Suomi NPP VIIRS Instrument: https://ncc.nesdis.noaa.gov/VIIRS/
4. Using night light emissions for the prediction of local wealth: https://journals.sagepub.com/doi/full/10.1177/0022343316630359
5. Using publicly available satellite imagery and deep learning to understand economic well-being in Africa: https://www.nature.com/articles/s41467-020-16185-w
6. Fully Convolutional Networks for Semantic Segmentation: https://arxiv.org/abs/1411.4038
7. U-Net: Convolutional Networks for Biomedical Image Segmentation: https://arxiv.org/abs/1505.04597
8. VIIRS DNB Monthly Stray-Light Corrected Composite: https://eogdata.mines.edu/products/vnl/#monthly
9. GEE Python API: https://developers.google.com/earth-engine/tutorials/community/intro-to-python-api
10. Tensorflow: https://www.tensorflow.org/about/bib
11. GitHub Code Repo: https://github.com/isaiahlg/unet-viirs
12. ESA's Copernicus Sentinel-2: https://sentinel.esa.int/web/sentinel/missions/sentinel-2