Using Semantic Segmentation to Identify Photovoltaic Panels in Satellite Imagery

Brigid Blakeslee (bab694), Mitchell Lloyd (mel652), and Yining Wang (yw2178)
Google Drive Link, GitHub Link

CSCI-GA 2271 Computer Vision, New York University
Dec 16, 2021

Abstract

In this project we present a means for identification of photovoltaic (PV) panels in satellite imagery. This capability offers significant potential benefit when identifying locations for the future installation of photovoltaic panels. We leverage an open source dataset of satellite imagery accompanied with ground truth data in the form of CSV files of polygons demarcating the presence of photovoltaic panels. We have developed an approach using semantic segmentation to identify these panels. Semantic segmentation is performed using a U-Net architecture. We present results of this approach, along with a comparison of performance resulting from changes in data preparation and augmentation.

1. Introduction

In attempting to assess deployment of renewable energy technologies across residential areas – a problem for which little centralized public data exists – convolutional neural networks can play a useful role. Public aerial imagery is widely available (USGS, NASA WorldView, etc.), presenting an opportunity for researchers in computer vision to automate detection and classification of such technologies. Photovoltaic (PV) solar array panels are one such technology that lends itself well to this approach. In this paper, we discuss a working model that can detect pixel-level PV panel boundaries at both industrial- and residential-scale deployments utilizing a modified U-Net architecture.

1.1. Related Work

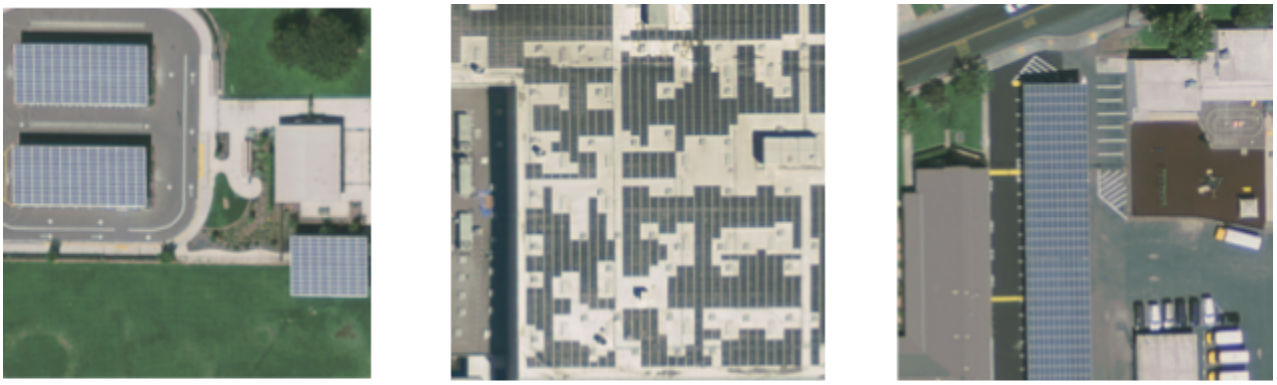
The primary source for related model inspiration is of course U-Net itself, as described in Ronneberger et al.'s seminal report on pixel-wise image classification for biomedical imagery [6]. This is not the only approach one could research, however: Mask-RCNN has been used to great effect in instance segmentation [5], for instance. Given our conceptualization of a semantic segmentation model, however, U-Net has the advantage in relative simplicity. Since its inception, U-Net has been applied to a number of problems [8,9]: Most notably for our interests is Hou et al.'s application of U-Net to aerial imagery in order to assess the distribution of solar farms in China [7]. This is, however, a large-scale focus, and loses the granularity of being able to assess smaller residential-scale deployments. Interesting work has also been done on loss function deployment, such as discriminative loss for semantic segmentation [5]. In our model, however, as will be discussed below, strong results can be obtained without the complexity of a discriminative loss implementation with simple Binary Cross Entropy loss.

2. Background

2.1. Datasets

The datasets employed in this project were first developed by the Duke University Energy Initiative, led by Bradbury et al. [1,2]. These datasets include satellite imagery captured in the cities of Modesto, Stockton, Oxnard, and Fresno, California and contain over 19,000 PV panels.

Each dataset contains high resolution TIFF images accompanied with XML files to support their display. Additionally, the dataset provides several CSV files containing polygon data for the PV array locations. These CSVs include data such as the centroid for each polygon (in latitude/longitude coordinates as well as in



Sample input images

pixel coordinates), the coordinates of the northwest corner and southeast corner of the image patch that a polygon is located in, the area covered by each polygon (in meters and in pixels) and the coordinates for each vertex comprising a given polygon (in both pixel coordinates and latitude/longitude coordinates). Additionally, there are two JSON files included with the data.

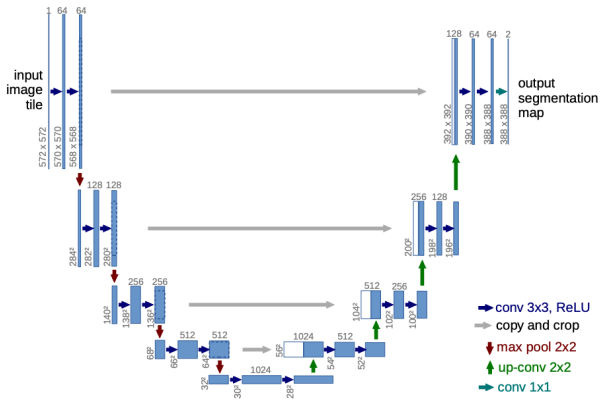
2.2. Object Detection in Satellite Imagery

Satellite imagery is used for a variety of applications. In this effort, we are focused on its use for detecting the presence of PV panels. Other environmentally-focused applications might use satellite imagery for assessing forested areas [11]. Beyond environmental use cases, there is clear applicability for surveillance and defense, as well as population and building-development studies.

In the case of detecting PV panels in satellite imagery, one particular challenge seems to be the significant differences in the environment within which they may be placed. For example, when installed in an urban environment, we might expect PV panels to be placed on structures such as rooftops. In a suburban environment, they might be installed on a residential roof or in a parking lot. In rural environments, they may be located far from man-made structures or infrastructure. We recognize how these disparate environments and conditions might create challenges when identifying PV panels in different scenes or if we were to extend this project to identifying an optimal site for their installation [12]. We mitigate this potential challenge by limiting the training and testing data to urban/suburban sites, as provided by the dataset presented in [1, 2].

2.3. Semantic and Instance Segmentation

We propose a methodology based on semantic segmentation. Semantic segmentation is primarily concerned with the segmentation and labeling of particular objects in a scene. The labeling can be done so as to label all pixels comprising objects of a particular image class as belonging to that class; in our case, either belonging to a 0 “background” class or a 1 “panel” class. This is in contrast to instance segmentation, where segmentation is performed so as to label pixels as belonging to a particular *instance* of that object class. Instance segmentation was beyond the scope of our particular approach, but remains an excellent follow-up for further research.



Vanilla UNet model implementation [6]

Taking inspiration from Ronneberger et al. [6], we chose to explore and implement a variation of the U-Net model for semantic segmentation. The original implementation of U-Net focused on biomedical applications for 3-channel images of size 572x572. In its declining “encoder” phase, at each model level, the image passes through two convolution + RELU layers before being passed through a max pooling layer on its way to a subsequent level. In effect, each level *doubles* the channels in the image while *halving* the dimensions. This continues until a bottom level is reached, at which point the model reverses in its ascending “decoder” phase, where the model upconverts the image before concatenating the upconversion tensor with the output from the *same* level on the encoder side. In the original model, this requires a center crop to ensure compatible dimensionality. The model then proceeds through two additional convolution + RELU layers at each level before upconverting to a higher level, and the cycle repeats. At its ultimate step, the image passes through a sigmoid function for class probability. The original model results in a 2-channel image of size 388x388.

2.4. Model Modifications

In considering an adaptation of the original U-Net model for our purposes, we began with three important modifications. First, as we intended our binary “test” images to be of the same dimensionality as our input images, we had to handle the fact that the original U-Net model reduces dimensionality by 2 at each convolution layer, ultimately resulting in a 388x388 image prediction that differs considerably from its 572x572 original. This effect is rectified by introducing padding of 1 to each convolution. Second, the previous change reduced the need for a “crop” action before the concatenation of encoder-level output with upconverted decoder-level output, as the dimensionality is preserved across levels.

Third, while the original model outputs 2-channel images, our intention was to cross-reference outputs against *single* channel binary masks, so we reduced channel output to 1.

Taken together, with our intention to feed in images of size 256x256 (discussed below), this ensures that a 3-channel, 256x256 input image is output as a 1-channel, 256x256 probability matrix.

3. Methodology

3.1. Data Preparation and Image Processing

In this project, we have leveraged the aforementioned dataset developed under the Duke University Energy Initiative [1,2]. As mentioned, this dataset consists of very large images of 5000x5000 pixels and comprehensive CSV files to capture polygon vertices corresponding to PV panels in these images, in addition to JSON files. Our work relies primarily on the images and CSV files.

To prepare the data, the 5000x5000 pixel images were subdivided into image patches of 256x256 pixels. This subdivision of the images leveraged OpenCV for reading the original, large-format images and saving the smaller image regions. Additionally, it was necessary to generate ground truth binary images of the PV panels to serve as target images. This was done by plotting the polygons included in the CSV as pixels of value 1 on a 5000x5000 array of 0-value background pixels using OpenCV tools for each image. These binary images were then subdivided in the same manner as the original images, and the names of the corresponding subdivided RGB images and binary images were listed in a CSV file.

3.2. Initial Model Performance & Tweaks

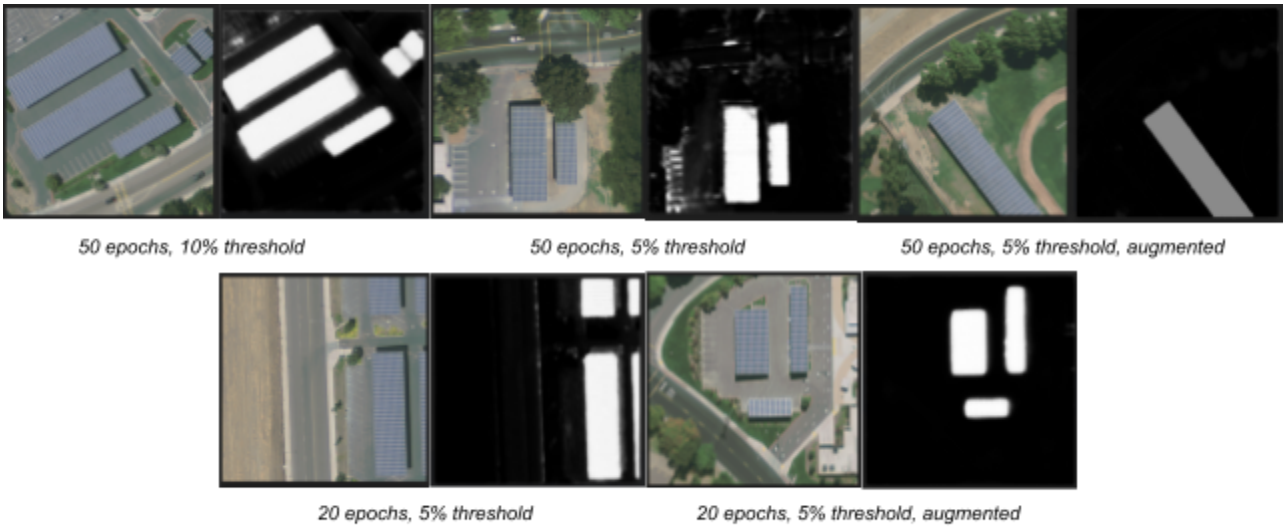
We initially tested the model against a training set where 90% of images contained a polygon (of indiscriminate size) and 10% of images did not. With the base model, we quickly experienced overfitting and the exploding gradients problem for the first 3 epochs (and halted thereafter). We then attempted to tweak the model by reducing channel conversions by a factor of 2: i.e., the first level converting from 3 to 32 channels (rather than 64), the second level converting from 32 to 64, etc. This did little to address the overfitting problem, but as loss was marginally reduced, we felt the channel conversion was a step in the right direction.

We then dug deeper into the work of various other researchers. As Hou et al. [7] has reported success in applying batch normalization to the convolution layers (before the RELU application), we tried the same for our model, both with batch size of 8 and 10. This solved the exploding gradients problem and produced an overall loss of as little as 0.0043, but when validating model performance we quickly learned why: The model was learning that, with relatively sparse images where polygon pixels make up very little of the overall image, a relatively low loss function could be achieved by simply predicting all pixels as “background”. We attempted to rectify this by moving to a binary cross entropy with logits function. This function allows a pos_weight parameter where one can more heavily weight the “succeeding” pixels – in our case, the class 1 polygons. We calculated a pos_weight by summing the total count of 1 pixels over the total count of 0 pixels for our training set, achieving a number as high as 93: Our images contained nearly 100 times as many 0 pixels as 1 pixels. It was here that we realized that our problem lay not with our attempted model tweaks thus far, but with our data processing approach.

3.3. Data Filtering and Augmentation

Realizing our previous error with regards to the imbalance of pixel values, we moved to consider how to handle the data to support effective training of the network. In spite of ensuring that only 10% of the images contained no PV panels, we were still faced with the fact that for many of the images containing PV panels there were very few white pixels. In short, it became apparent that balancing the dataset at the image level – images that contained any white pixels versus images that contained no white pixels – was insufficient. Instead, we recognized that we needed to balance the dataset at the pixel level as well. To address this, we found that by specifying a threshold of pixel cover – a minimum percentage of white pixels per image that must be exceeded for an image to be kept as part of the dataset – we could limit our dataset to those images with solar panels large and/or dispersed enough to allow the model to properly train. We then selected threshold levels of 10% and 5% for our testing and validation.

Thresholding on the percentage of white pixels in a given image subregion effectively stipulates a certain density or size of PV arrays. Considering this in a real-world context, we can envision that this may make our network more suited to non-residential installations, or areas where there are many neighboring residential installations, depending on the image subregion size. For



Sample results

example, a typical residential home with panels installed on the roof may or may not be sufficient to exceed this threshold, whereas a larger industrial installation or cluster of several residential installations may exceed this threshold.

Specifying a threshold as such had the effect of significantly reducing the number of training images available. To compensate for this, we augmented the input data with two rotations: both 90 and 180 degrees. This offers greater robustness when considering the variability of image appearance that may be possible, in addition to increasing the volume of data we ultimately have to train and test our network.

Data was augmented through rotation using OpenCV. The original binary and RGB images were rotated by 90 and 180 degrees and saved before being subdivided into image subregions of 256x256 pixels, as in the initial dataset generation. This augmented dataset, containing the original image pairs as well as the rotated image pairs, serve as input to the network. Rotations in increments of 90 degrees eliminate the potential challenges of portions of an image falling outside of the 256x256 square as a result of a different angular rotation.

While not used in this case, another image processing mechanism that could have been performed in the service of data augmentation would be image re-scaling. The relevance of re-scaling is somewhat contingent on the physical sensing modality available and whether a "zoom" functionality or viewing an area from a nearer distance is possible. Data augmentation through implementation of a color-balance shift to simulate cloud cover was also considered, but owing to our previous difficulties with this relatively untested data set, we opted to omit this element. While a color-balance shift would increase the amount of data we have to work with, it also introduces a variability that could present a challenge. This variation of color balance could be very useful in the future in making our system robust to changes in environmental conditions and assess its performance with images of varied appearance.

3.4. Training and Testing Data

As mentioned previously, the dataset [1,2] included images and ground truth for five different cities. We selected two of these cities – Modesto and Stockton – to serve as our dataset. 80% of the dataset is used as a training dataset, and the remaining 20% is used as a validation dataset. Another possible means for organizing training and testing data would be to use one or more cities for training and a yet-unseen city for testing. We can anticipate that were this deployed in a real-world setting, a network trained with data from some set of cities would be used to identify PV panels in images from a yet-to-be seen city. While we recognize the potential benefits of such a training and testing split, in this case we mixed images from two cities for both training and test.

Formal shape parameters are as follows: The input data to our model is an RGB image with the shape (1,3,256,256), the target binary image is of shape (1,1,256,256), and the predicted binary image from the model is of shape (1,1,256,256) as well.

During the training, the Adam optimizer was used for back propagation with the learning rate set to 1e-3 initially. The learning rate was adjusted with a step scheduler every 5 epochs, with a gamma of 0.1. The loss

function used is Binary Cross Entropy (as opposed to BCE with Logits, as discussed previously), given the model's output of a 0 to 1 binary probability between a 0 "background" class and a 1 "panel" class.

We tested the model with both 20-epoch and 50-epoch runs, both at the aforementioned panel-pixel density thresholds of 10% and 5%. We also tested the model using the original, un-rotated dataset as well as with a dataset three times as large, including images rotated by 90 degrees clockwise and 180 degrees, in addition to the original images.

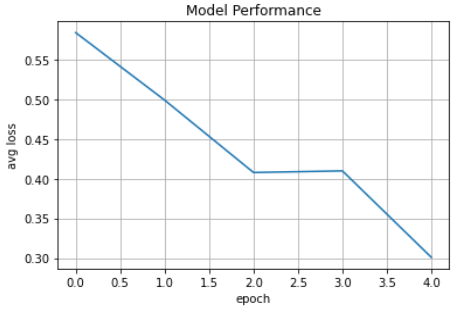
4. Results

4.1. Images Filtered to Contain 10% White Pixels

To start, we began by filtering the dataset to contain only image subregions where more than 10% of the image pixels were white. This resulted in a vast improvement in performance over the prior implementation in which a single white pixel was sufficient for an image to be included in the dataset.

5 Epochs

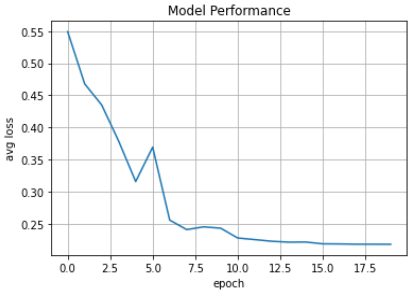
We started by training a network using images containing 10% white pixels over 5 epochs, as a sort of "proof of concept". The loss function results are as appears below:



As can be seen in this very rudimentary training effort, the loss function value decreases rapidly. Additionally, the output images were fairly close to the target images.

20 Epochs

We then trained the network over 20 epochs to see what potential improvements could be made in performance through additional training:



Again, the network converges quickly, albeit with an anomalous increase in loss-function value at one point in the training process. The loss function was able to converge to a lower value than when only trained over 5 epochs, and reached a steady-state of around 0.2.

Visually, the output of the network trained over 20 epochs with this filtered dataset was quite strong. We present a few illustrative examples of disparate compositions.

The image on the left is the input, the center image is the target image, and the image on the right is the output.

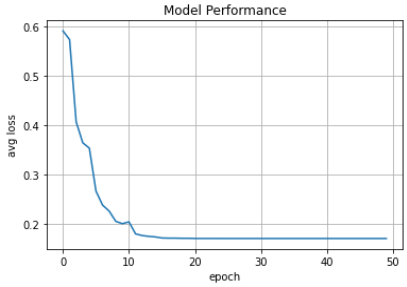
Looking at the images above, it is clear that the network performs quite well in identifying the PV panels. There are some spurious non-zero pixel values resulting from background objects, however the PV panels are the dominant feature in the output image. Another validation result from this 20-epoch training appears below. This image has greater complexity with regards to the PV panel geometry.

Input (left), target (center), output (right) for a network trained over 20 epochs with images containing at least 10% white pixels.

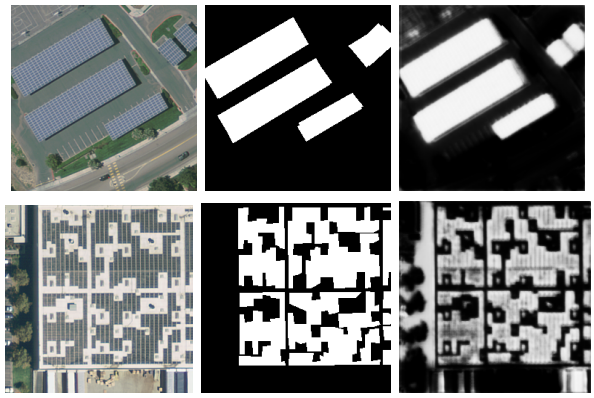
The PV panel geometry in this case has greater complexity than the previously discussed example. Again, there are clearly some background features that erroneously appear as non-zero pixels in the output image, however in general, the performance is still quite strong.

50 Epochs

We now consider the performance training the network over 50 epochs, with a threshold of 10% white pixels:



It is evident that this loss function value converges to a lower steady-state value then when the network was only trained over 20 epochs. In this case, the average loss value converges to about 0.15.



Input image (left), target image (center), output image (right) for a network trained over 50 epochs with images containing at least 10% of pixels with values equal to 1.

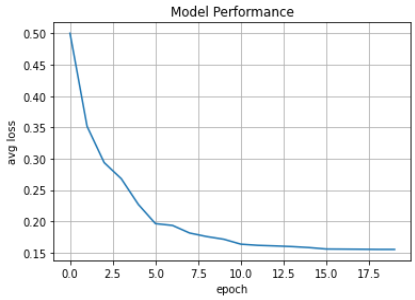
These two sets of images contain very different configurations and geometries of PV panels. In the top row, it is clear that the PV panels are identified with very few erroneous pixel values in the image background. In the bottom row, some background pixels are erroneously identified as belonging to the PV panel class. In general, however, the appearance of the output after training over 50 epochs, rather than over 20 epochs, is improved and appears to have fewer non-zero background pixel values.

4.2. Images Filtered to Contain 5% White Pixels

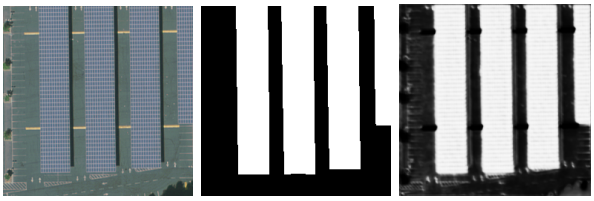
We then decreased the threshold for the percentage of white pixels for an image to be included in the dataset to 5%. We felt that this would be more inclusive of different PV panel array sizes and densities, yielding a potentially more versatile solution. We also felt that this would increase the size of the dataset somewhat by filtering out fewer images. Finally, we had some concern as to whether this might present a challenge as images would be more imbalanced and dominated by non-PV pixels.

20 Epochs

We first began by training the network over 20 epochs:



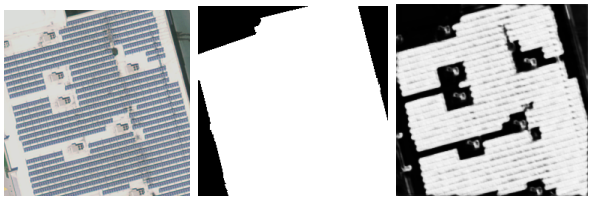
Trained with this data over 20 epochs, the network converges to an average loss-function value comparable to, or slightly lower, than the resulting average loss function value achieved when training a network with images containing 10% white pixels (fewer images, less imbalance data) over 50 epochs. We now include some representative results.



Input image (left), target image (center), output image (right) for a network trained over 20 epochs with images containing at least 5% of pixels with values equal to 1.

The above image has a geometrically simple PV panel installation. The result has relatively little erroneous noise in the background, and clearly identifies the PV panels.

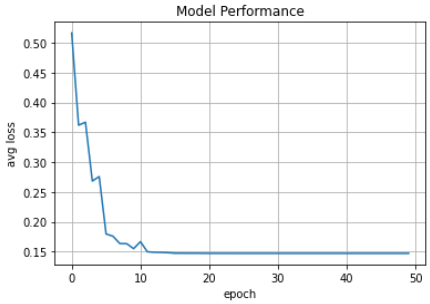
In the image below, the panels installed have a much more complex geometry. The output is clearly able to identify the PV panels, even when the target image greatly simplified geometry:



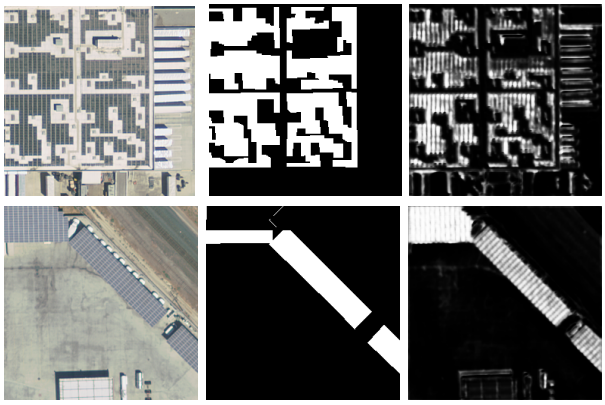
Input image (left), target image (center), output image (right) for a network trained over 20 epochs with images containing at least 5% of pixels with values equal to 1.

50 Epochs

We now train the network with images containing at least 5% white pixels over 50 epochs:



As can be observed in the plot above, the network converges to an average loss value of .15. The performance of this network is also strong in the resulting output, as depicted in the example cases below. It is clear, however, that some pixels are incorrectly labeled as belonging to a PV panel, particularly in the top row.



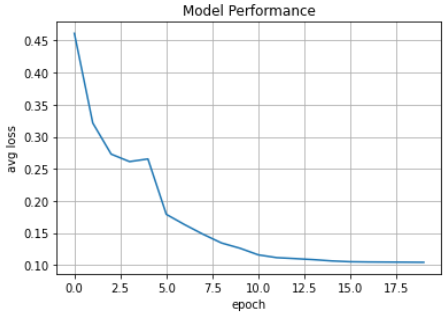
Input images (left), target images (center), output images (right) for a network trained over 50 epochs with images with at least 5% of pixels with values equal to 1.

4.3. Images Filtered to Contain 5% White Pixels with Added Rotation

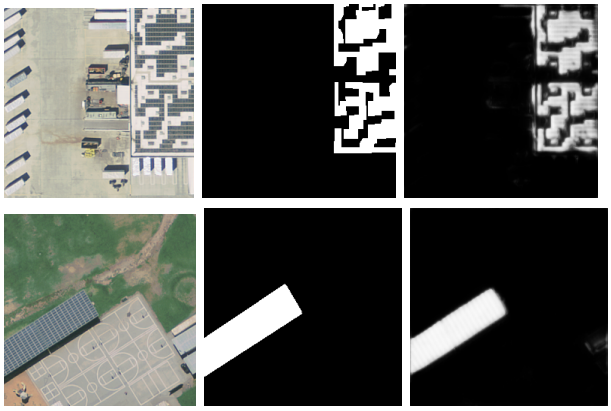
We now consider the performance of our network when trained with augmented data. As described previously, the images have been rotated by 90 and 180 degrees, resulting in a greater variety and number of images (3x as many).

20 Epochs

We first present the average loss function value after training the network over 20 epochs, followed by representative samples of the network output.



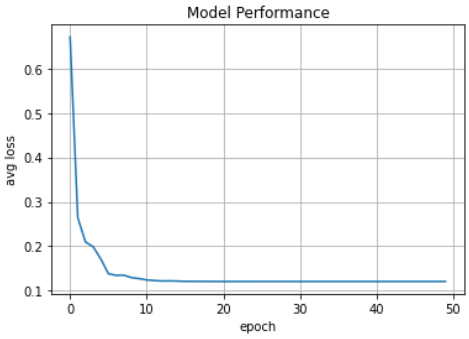
The output of the network when trained with the augmented dataset is clearly an improvement over the previously discussed results. In particular, we consider the appearance of what seem to be several trucks. In the previous example, the apparent trucks in the input image were incorrectly flagged as PV panels. In this case, with the rotated images added to the dataset, they were correctly *not* identified as PV panels (*first row*).



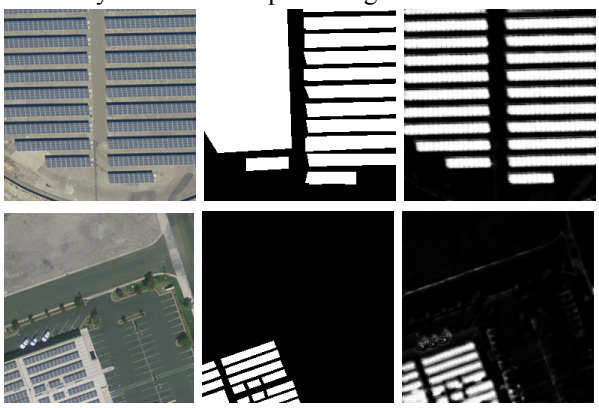
Input (left), target (center), and output (right) for a network trained over 20 epochs with an augmented dataset with images containing at least 5% of pixels with values equal to 1.

50 Epochs

We again observe a strong result in the rapid decrease and low steady-state value of the loss function when the network is trained over 50 epochs using augmented data.



Note again an instance where the mask greatly simplified the panel composition; like before, the model was able to accurately determine the panels regardless.

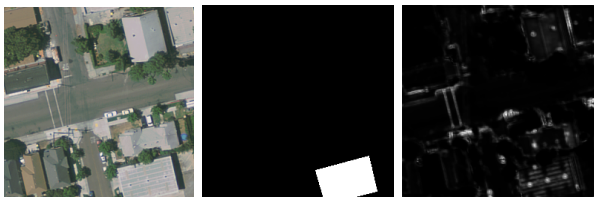


Input (left), target (center), and output images (right) for a network trained over 50 epochs with an augmented dataset composed of images with at least 5% of pixels with values equal to 1.

In addition to strong performance with regards to identifying the PV panels in the above images, there is also a lack of incorrect background pixel values.

4.4. Outliers

While the performance was generally very strong, it is important to note that in each data and training configuration, there were some validation results that appeared as outliers, with generally poor performance in identifying PV arrays in the image. We provide an example below from the validation set for the network trained over 50 epochs using augmented data with a 5% threshold for white pixel values.



Input (left), target (center), and output (right) found using a network trained over 50 epochs with an augmented dataset of rotated and non-rotated images with at least 5% of pixels with values equal to 1.

In this image there is a single PV panel array in the lower-right corner. Unfortunately the network fails to detect this in the input image, and instead the output is populated with spurious non-zero pixels in the image background. This is surprising given the generally strong performance of the network based on the loss function value and other output images. It seems to be that, given the shade of color of the panel, the model is having difficulty distinguishing between it and neighboring roofs.

Another outlier in performance that can be observed in the other examples provided in the results section is the erroneous presence of non-zero pixel values where there is not a PV array. This may potentially be attributed to the variation in appearance of PV panels (lighter or darker, larger or smaller size based on installation configuration) as well as the appearance of neighboring objects such as cars or parking lots. We do note several cases, however, where the network successfully discriminates between a similar looking object and a PV panel, however this is a potential challenge in the success of the network that could be mitigated perhaps through a larger set of training data, augmented data to ensure robustness to variability in color, or the addition of semantic labeling for neighboring objects to have greater scene context.

5. Next Steps

There are a number of opportunities for expansion in this model and approach. As mentioned previously we had initially tested the model with batch normalization in place after each convolution layer. Owing to time constraints, we were unable to sufficiently test this feature with our filtered, augmented data set, though it remains an area to explore in subsequent iterations. In a similar vein, we did not explore the contribution that dropout may have on either the encoding or decoding phase of UNet – combined with batch normalization, it may be quite effective.

Contextual applications of our model may also lend themselves to expansion. We may, for example, leverage semantic segmentation to identify characteristics of regions *surrounding* existing photovoltaic arrays that may support establishing a definition of an “ideal” installation site. The definition of an optimal site may vary based on the context in which it exists – for example, in an urban environment, perhaps photovoltaic arrays are best installed on rooftops, whereas in rural environments, they should be installed in open fields. These site-specific definitions of optimal installation location characteristics can be developed through application of detection of already-deployed photovoltaic arrays in conjunction with semantic labeling of their surroundings.

6. Conclusion

Through this project we have demonstrated the ability to

detect photovoltaic arrays in satellite imagery using a relatively small selection of an open source dataset. Our approach was to modify the U-Net architecture originally described by Ronneberger et al. [6] to account for identical input/output dimensions and single-class classification. The results we obtained were surprisingly robust: The model had no problems delineating the borders of the panels in our filtered examples, both before and after image augmentation through 90- and 180-degree rotation, at least once we accounted for filtering out images that were particularly sparse. Accounting for this relative sparsity and training the model further to identify even the smallest rooftop panels would be an interesting area of further research. Building on this research with such training would go a long way to providing researchers and policymakers with the tools needed to assess large-scale renewable energy deployments using both satellite and other forms of aerial imagery.

7. References

[1] K. Bradbury, R. Saboo, T. Johnson, J. Malof, A. Devarajan, W. Zhang, et al., "Full Collection: Distributed Solar Photovoltaic Array Location and Extent Data Set for Remote Sensing Object Identification," *figshare*, https://doi.org/10.6084/m9.figshare.c.3255643.v4, 2016.

[2] "New dataset developed at Duke will benefit solar energy growth," *Duke Energy Initiative*, https://energy.duke.edu/news/making-solar-count-new-dataset-developed-duke-will-benefit-solar-energy-growth, 2016.

[3] X. Yuan, J. Shi, and L. Gu, "A review of deep learning methods for semantic segmentation of remote sensing imagery," *Expert Systems with Applications*, vol. 169, 2021.

[4] B. De Brabandere, D. Neven, L. Van Gool, "Semantic Instance Segmentation with a Discriminative Loss Function", arXiv:1708.02551 [cs.CV], 2017.

[5] K. He, G. Gkioxari, P. Dollár, R. Girshick, "Mask R-CNN", arXiv:1703.06870 [cs.CV], 2017.

[6] O. Ronneberger, P. Fischer, T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation", arXiv:1505.04597 [cs.CV], 2015.

[7] X. Hou, B. Wang, W. Hu, L. Yin, H. Wu, "SolarNet: A Deep Learning Framework to Map Solar Power Plants in China From Satellite Imagery", arXiv:1912.03685 [cs.CV], 2019.

[8] M. Cordts, M. Omran, S. Ramos, T. Rehfeld et al., "The Cityscapes Dataset for Semantic Urban Scene Understanding", arXiv:1604.01685 [cs.CV], 2016.

[9] C. Bartolome, Y. Zhang, A. Ramaswami, "DeepCell: Automating cell nuclei detection with neural networks", *CS230: Stanford University*, 2018.

[10] S. Du, "Understanding Focal Loss for Pixel-Level Classification in Convolutional Neural Networks", *Medium*, https://medium.com/swlh/understanding-focal-loss-for-pixel-level-classification-in-convolutional-neural-networks-720f19f431b, 2020.

[11] D. Rolnick, P. L. Donti, L. H. Kaack, K. Kochanski, A. Lacoste, K. Sankaran, A. S. Ross, N. Milojevic-Dupont, N. Jaques, A. Waldman-Brown, A. Luccioni, T. Maharaj, E. D. Sherwin, S. K. Mukkavilli, K. P. Kording, C. Gomes, A. Y. Ng, D. Hassabis, J. C. Platt, F. Creutzig, J. Chayes, Y. Bengio, "Tackling Climate Change with Machine Learning," arXiv:1906.05433v2 [cs.CY], 2019.

[12] J. J. Vega Diaz, M. Vlaminck, D. Lefkaditis, S. A. Orjeula Vargas, H. Luong, "Solar Panel Detection within Complex Backgrounds Using Thermal Images Acquired by UAVs," *Sensors*, 20, vol. 21, 2020.

Libraries:

- Matplotlib: https://github.com/matplotlib/matplotlib
- NumPy: https://github.com/numpy/numpy
- OpenCV: https://github.com/opencv/opencv
- Pandas: https://github.com/pandas-dev/pandas
- PyTorch: https://github.com/pytorch/pytorch