# Binary Classification : Solar Panel Identification in Satellite Images

## Nathan Inkawhich, Lisa Sapozhnikov, Yifan Li

Netids: nai2, ls258, yl506

Link: https://github.com/inkawhich/machine-learning-course/blob/master/Kaggle%20Competition%20Report.ipynb (https://github.com/inkawhich/machine-learning-course/blob/master/Kaggle%20Competition%20Report.ipynb)

## Abstract

As the solar industry grows and solar panels become more prevalent, it is valuable for many parties, including utilities companies and climate change activists, to have a greater volume of information about the distribution of solar power across regions. With the goal of automating the collection of such data, this work seeks to design a computer vision solution for detecting solar arrays in satellite images. In particular, we use a lightweight Convolutional Neural Network (CNN) as a binary classifier to determine if a given image contains a solar panel or not. In validation, the trained model greatly outperformed a random classification algorithm, with an average accuracy of 88.12% and average AUC of 0.952 across five folds of cross-validation. In testing, our model achieved an even higher accuracy of 93.45%. These results indicate that it is productive and valuable to apply a CNN to this problem, and the approach shows great potential for further improvement upon training with a larger dataset.

## Introduction

With the reality of climate change sinking in, the renewable energy sector of the United States is growing dramatically, with solar panels as a leading solution. In 2015, solar energy related employment had already overtaken oil, gas, and coal employment in the United States, and in 2016, more than 260,000 Americans were employed in the solar industry. As of 2017, the United States has over 50 gigawatts (GW) of installed photovoltaic capacity, with solar power technology generating over 52.1 terawatt-hours (TWh), or 1.29% of the US electricity, over the course of the year (Solar Energy Industries Association 2017).

Given this rapid growth, information about regional distributions of solar panel arrays is becoming very valuable to power grid designers who wish to optimize power line locations for maximum energy delivered to and from the photovoltaic cells. Neighborhood-level information on solar panel geography would enable for the development of predictive algorithms to allow power companies to foresee and develop to support future growth. Statistics on neighborhoods resisting the increasing reliance on solar energy would also provide climate change activist groups with a more tangible goal in dissipating information on the necessity of renewable energy.

Unfortunately, despite the rapid scaling and increasing interest, very little accessible and centralized data exists on the locations of photovoltaic installations. User surveys are an unreliable metric, and accessing the already disorganized national and state records requires building permit data and working with utilities companies to gain access to proprietary data under use agreements. Given these circumstances, we must look for other methods of data collection. To escape the limitations of the currently available national utility data, we turn to the combination of high-resolution satellite images and computer vision algorithms to automatically generate information about solar panel distribution across the United States. In this work specifically, we work on a subset of the problem which assumes that images have already been generated and regions of interest have been isolated.

Recent research has shown that machine learning algorithms for image classification are very powerful and yield close to, if not better than, human results on some datasets (Russakovsky et al. 2015). Most of the work in this area has focused on complex model design for very large scale datasets with many classes. In this work, we leverage the technology being developed by these researchers to answer a much simpler binary question: *Does a solar panel exist in a given satellite image?* To answer this question we train a Convolutional Neural Network (CNN) on a small set of labeled training data, then evaluate our method through validation and testing. As a result, we find that CNNs are in-fact an effective approach to this problem. Our initial results are promising, and in the following sections we will discuss our approach and findings in greater detail.

## Background

Previous research conducted with the goal of identifying solar panels in satellite images with the use of machine imaging validates this as a field of interest and shows how much further exploration there is to do in the field. A variety of supervised learning methods have been used successfully to classify satellite images as containing solar panels.

Collins et. al. (2015) and Malof et. al. (2016) both emphasized image preprocessing and feature extraction. Collins et. al. divided each image into regions and returned the probabilities that each region contained a solar panel. To alleviate some computational overhead, they used each grey-scaled region's foreground color to determine a pre-screening confidence metric and eliminate the lowest scoring 30% of regions. After this early elimination, additional features such as background color, extent, compactness, solidity, mean, variance, and Kurtosis of grey-scale pixels within each region were extracted as predictors. Feeding the predictors and observations into a trained support vector machine classifier yielded an accuracy rate of 94% with only 4 false detections.

Malof et. al. focused on classifying the entire image rather than segmenting it, transforming the original image into the feature image, a new representation where each pixel is now represented by an $M$-dimensional vector of feature values. The feature values consisted of the means, $\mu$, and variances, $\sigma^2$, computed in several 3x3 windows surrounding the pixel, for each channel of the original RGB image. This feature matrix was fed into a Random Forest Classifier which uses decision trees to assign confidence values to individual pixels. The resulting confidences were post-processed to remove false maximas and classify regions of high confidence as solar panels. Both methods yielded high success rates showing image pre-processing to be a valuable step in accurate classification.

Both Bezobrazov et. al. (2017) and Imamoglu et. al (2017) both relied upon CNNs as their model training technique. Bezobrazov et. al. fed the satellite images into the first convolution layer followed by a pooling layer. The last fully connected layer of the network has one neuron only, which

produces the probability of presence solar panels on image. This simple neural network only yielded an accuracy rate of 87% indicating that image preprocessing is invaluable in increasing model accuracy. Imamoglu et. al. 2017 modelled their CNN after the human vision circuit introducing a feedback mechanism in addition to the traditional CNN feed-forward data flow. With this feedback structure, it is possible to obtain richer feature representation before the decision layer by providing relatively low-level contextual features, yielding an true negative rate of 93% and true positive rate of 99%.

## Data

Each satellite image in our dataset is in the form of a $101x101$px RGB image. In total, the given training set contains 1,500 of these images. Below are some examples of the given training data. Fig. 1 shows three examples of images containing no solar panels and Fig. 2 shows three examples of images that contain solar panels. Note, the yellow annotations in Fig. 2 do not exist on the actual images but are shown here to highlight the presence of the solar panels.

*Figure 1.* Satellite Images Lacking Solar Arrays

*Figure 2.* Satellite Images Lacking Solar Arrays

From simply visualizing the data, it seems apparent that color is the greatest indication of the presence of a solar panel. The human eye identifies each solar panel by discerning a shape significantly darker than its surroundings. A challenge inherent in relying on color for identification is that not any splatch of black is actually a solar panel. A fundamental question a classifier must answer is how large and distinct in color must a dark patch of pixels be to be confidently identified as a solar panel?

On the other hand, features such as pixel geography and shape do not seem to be entirely indicative of solar panel identification. It will be crucial to not overtrain the model to expect a solar panel in a certain region or expect the solar panel to be a neat square or rectangle. These challenges may be overcome by training our model on images containing a variety of solar array shapes and on various rotations of the same image so that the model does not weigh solar panel location too highly. Overall, from visualizing the data, we see that this is not a trivial problem. The solar panels appear in different spatial locations, with different rotations, and in different colors and shapes. Our classifier must be robust to this type of variance which is something we must account for in the training process.

## Methods

The fundamental components of our method are a CNN implemented in the Caffe2 framework, a two step data augmentation technique when training, and a results averaging step when testing. In addition, we do employ data cleaning and preprocessing, all of which is described below. A flowchart of our method is also shown in Fig. 4 below.

### Data Cleaning/Equalization

The first step in our approach is to equalize the representation of the classes in the training data, as to not create a model that is bias towards one class. In the full training set there are 1500 total observations, 505 of which belong to Class 1 (solar panel present) and 995 of which belong to Class 0 (solar panel absent). To prevent an inflated true negative rate from the training data, we equalized our training set by arbitrarily selecting 505 Class 0 images and disregarding the remaining 495. This yields a training dataset consisting of 1010 total images, where each class makes up exactly half of the data.

### Preprocessing

To improve the speed and accuracy of our training model, our training images underwent several preprocessing steps before being input into our model. First, to reduce computational overhead, each 101x101 pixel image was center cropped, keeping the center 90x90 pixels. We found that this area was large enough so that even boundary-adjacent solar panels were identifiable. This method also significantly reduced model training time without a significant effect on our model's accuracy.

To improve generalizability and reduce the effect of overfitting to the extremely limited amount of training examples, we trained with two common augmentations of the data. First, each training image was flipped horizontally, and second, each image was rotated 90 degrees, effectively tripling our training observations. Each of these images was then normalized through an approximate subtraction of the mean (by subtracting 128 from each of the images which have features in the range [0,255]), which is another common data manipulation technique when working with CNNs.

### Model Architecture

The CNN model architecture used is an adaptation of the famous LeNet model, which was the first very successful CNN for the MNIST dataset (LeCun et al. 1998). The original LeNet model architecture is shown in Fig. 3. Since our data has significantly higher dimensionality than the grayscale 32x32px MNIST images the LeNet model was designed for, we adapt the model to accept 3 channels of input (for our RGB images), have a third convolutional and max pooling layer, and also introduce ReLU activations after each layer. However, we do not increase the number of filters in the convolutional layers, nor do we add additional fully connected layers.

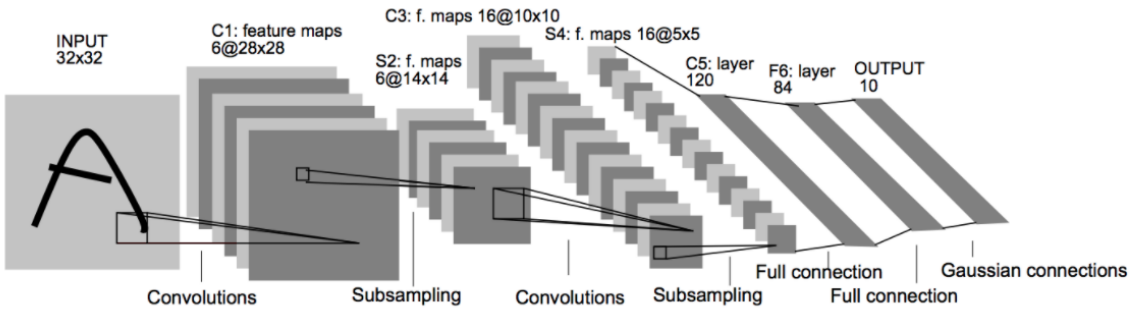


Figure 3. Visualization of the original LeNet CNN model for the MNIST dataset.

### Training

We trained our CNN on batches of size 150. Recall, our augmentation technique triples the amount of data, so each batch contains 50 unique original samples which we then apply our preprocessing to, to create a total of 150. The optimization algorithm used during backpropagation is vanilla Stochastic Gradient Descent (SGD) with a fixed learning rate of .01. In this work we trained the model for 45 epochs which was enough for the training accuracy to converge to 100% across most of the batches. Our training iterations (epochs) and learning rate was empirically chosen after trying different configurations, and was shown to give a favorable accuracy and training time tradeoff.

### Testing

This testing section refers to how we use the model once it has been trained and we are concerned with running inference. Similarly to how we handle the training data, each test image is cropped, mean-subtracted, horizontally flipped, and rotated. Given a single 101x101px image that we wish to classify, the following steps are taken. First, the image is center cropped to 90x90px then pushed through the classifier to get a prediction. Second, cropped image is flipped horizontally then pushed through the classifier to get another prediction. Next, the cropped image is rotated by 90 degrees and that is classified to get a third prediction. Finally, to obtain the final result, the output predictions from each of the three versions are averaged and the average prediction is returned as the final prediction. This process is visualized in Figure 4 below.

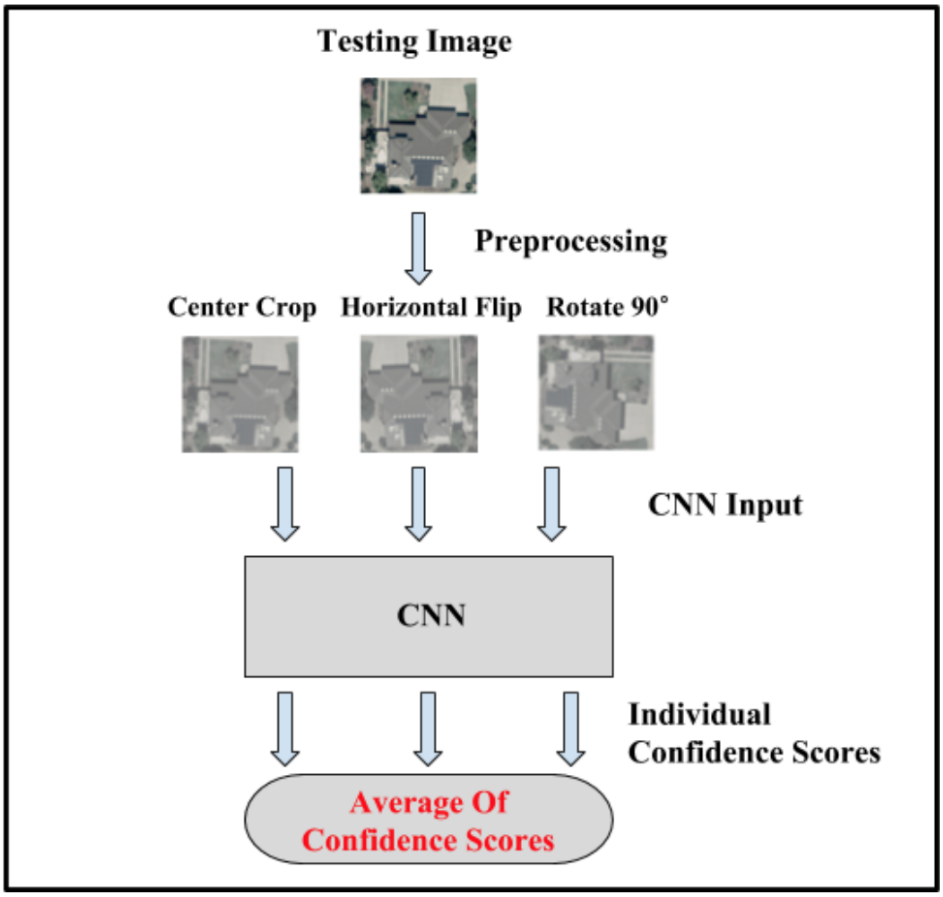


*Figure 4.* Testing Data Flow. Like all training images, all testing image are center-cropped, normalized and used to create a flipped copy and a rotated copy. All three copies are input to the CNN model which outputs three individual confidence scores to be averaged together to calculate the final confidence score of the satellite image.

# Results

Now, we will discuss how our model performed as compared to a baseline, how it performed over 5 splits of cross validation, and where/how our model failed. Before beginning detailed discussion of the CNN cross-validation results, we will first explain the baseline method.

### Baseline

For the baseline, we use a K-Nearest Neighbor (KNN) classifier with handcrafted features from the images. After representing each training image as a grayscale image and flatting the two-dimensional image into a vector, we extract 4 features: the mean, the median, the standard deviation, and the standard error of the mean. We then employ a simple validation approach where we used 80% of the training data for the train data (808 images), and the remaining 20% for the validation set (202 images). On this validation set, the resulting AUC was only around .63, which is better than random guessing but significantly lower than our CNN implementation. This baseline represents an extremely simple and naive approach for classification that performs better than guessing.

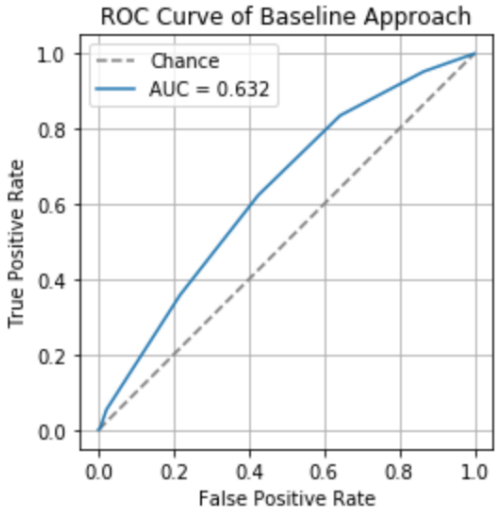


*Figure 5.* The ROC Curve of Simple Baseline KNN Approach

### CNN Results

We measured our model's generalization performance through 5-Folds Cross Validation. Below, Figures 6-10 show the per-fold results. For each fold, we display the training accuracy/loss plotted per iteration and the confusion matrix produced by testing on the held-out fold.

### Fold 1

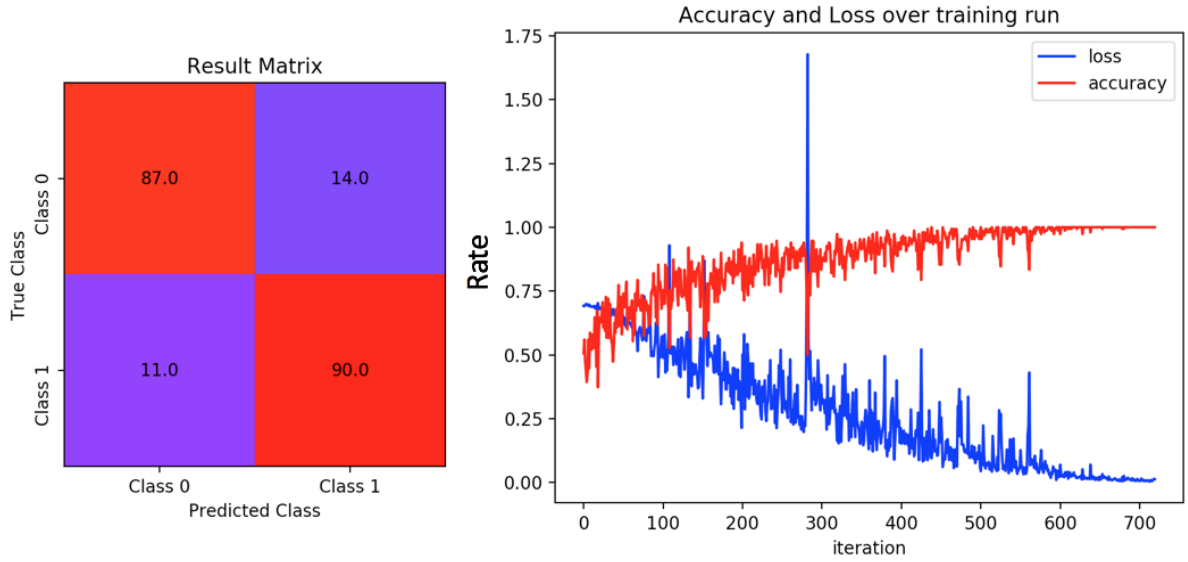


*Figure 6.* First round of 5-Folds Cross Validation.

**Fold 2**

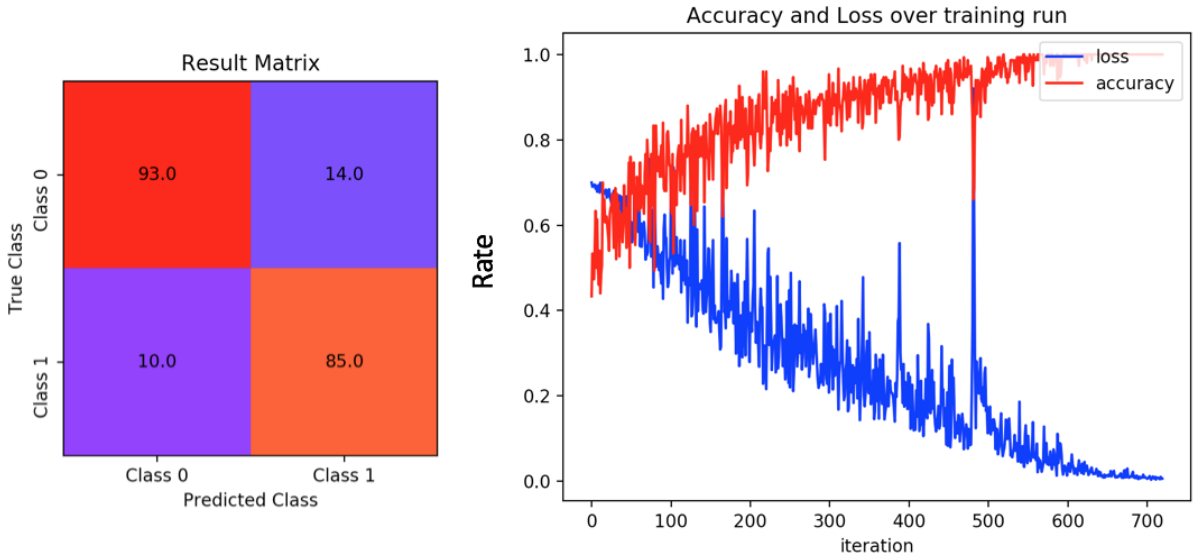


*Figure 7.* Second round of 5-Folds Cross Validation.

**Fold 3**

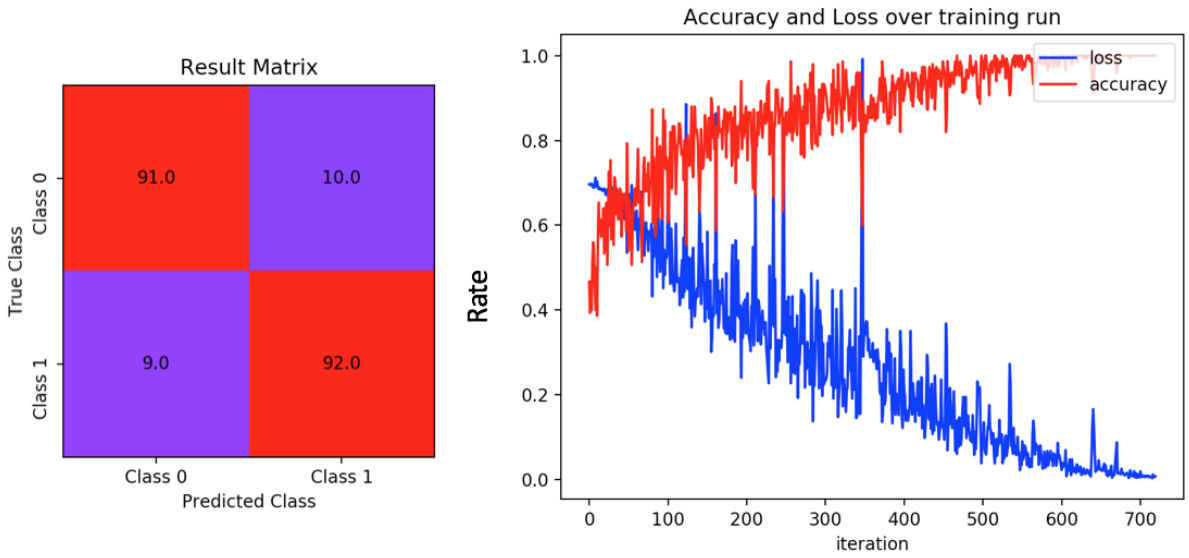


*Figure 8.* Third round of 5-Folds Cross Validation.

**Fold 4**

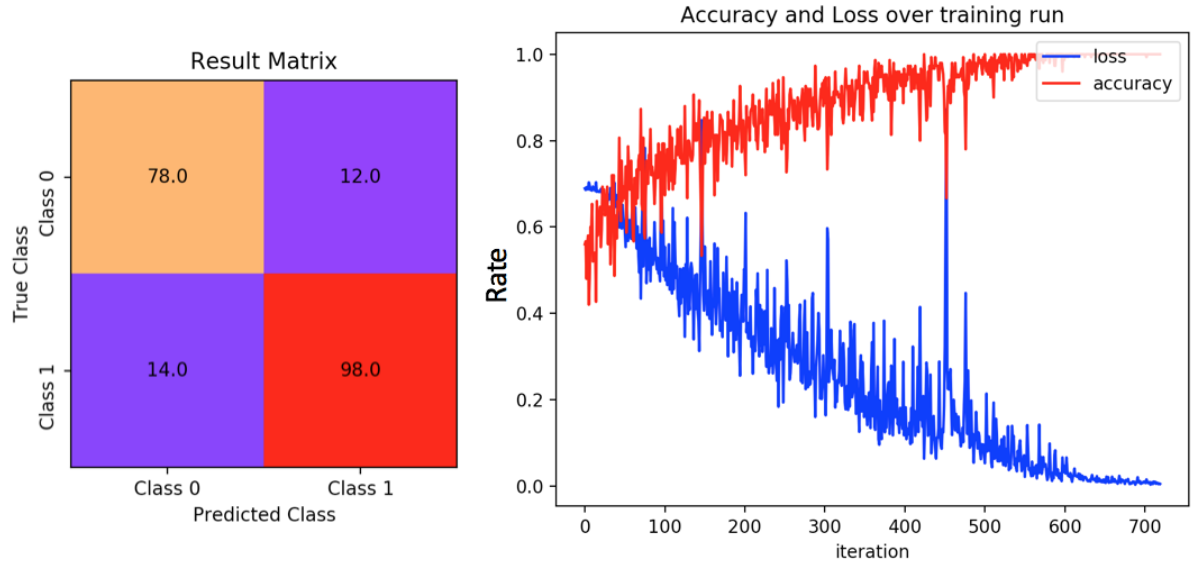


*Figure 9.* Fourth round of 5-Folds Cross Validation.

**Fold 5**

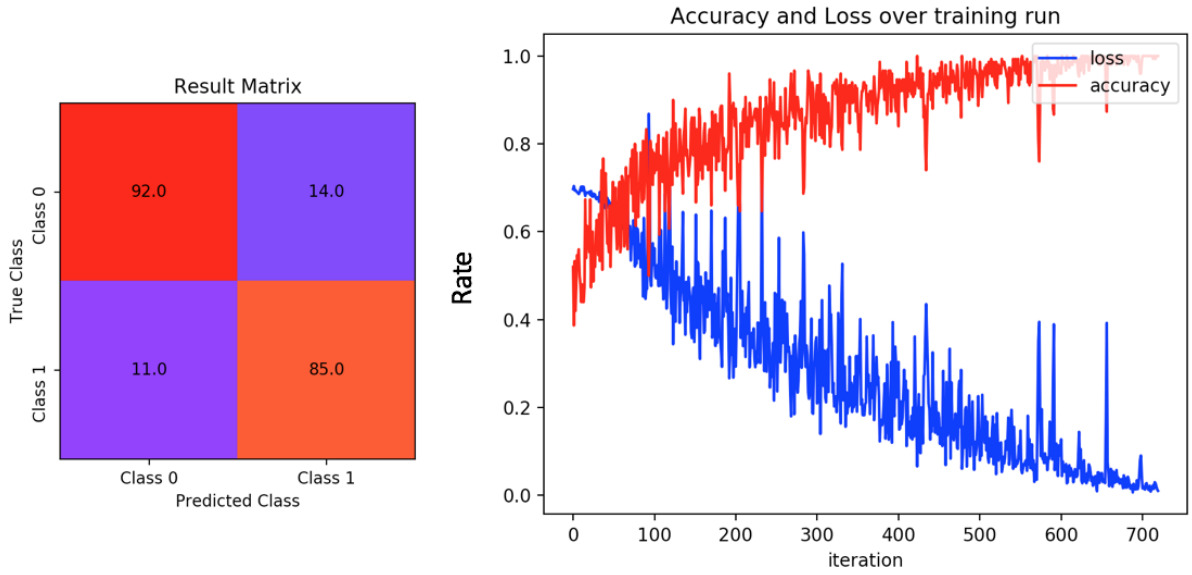


*Figure 10.* Fifth round of 5-Folds Cross Validation.

From Figures 6-10, a few trends are evident. First, each of the training accuracy/loss versus iteration plots show that over the 45 training epochs, the models each converged to nearly 100% training accuracy and close to 0 loss. Although this does not give an indication of generalization performance, it does give an indication that training is complete. Next, we see that the confusion matrices have strong diagonals, meaning the accuracy as measured on the fold is relatively high, and there is a balance in the number of false negatives and false positives.

| Test Fold | True Positive Rate | True Negative Rate | False Positive Rate | False Negative Rate | Accuracy |
|---|---|---|---|---|---|
| **1** | 89% | 86% | 14% | 11% | 87.62% |
| **2** | 89% | 87% | 13% | 11% | 88.11% |
| **3** | 91% | 90% | 10% | 9% | 90.59% |
| **4** | 87% | 86% | 13% | 13% | 87.12% |
| **5** | 88% | 86% | 13% | 12% | 87.62% |

*Table 1.* Comparing generalization results across folds.

From table 1, we immediately see that the accuracy as measured on each fold is fairly consistent, with an average of about 88.21%. This number is a good indication of generalization performance and shows our method is effective. The false positive rate is on-average slightly higher than the false negative rate indicating that our model is more likely to detect a solar panel when there isn't one than miss a solar panel present in an image.

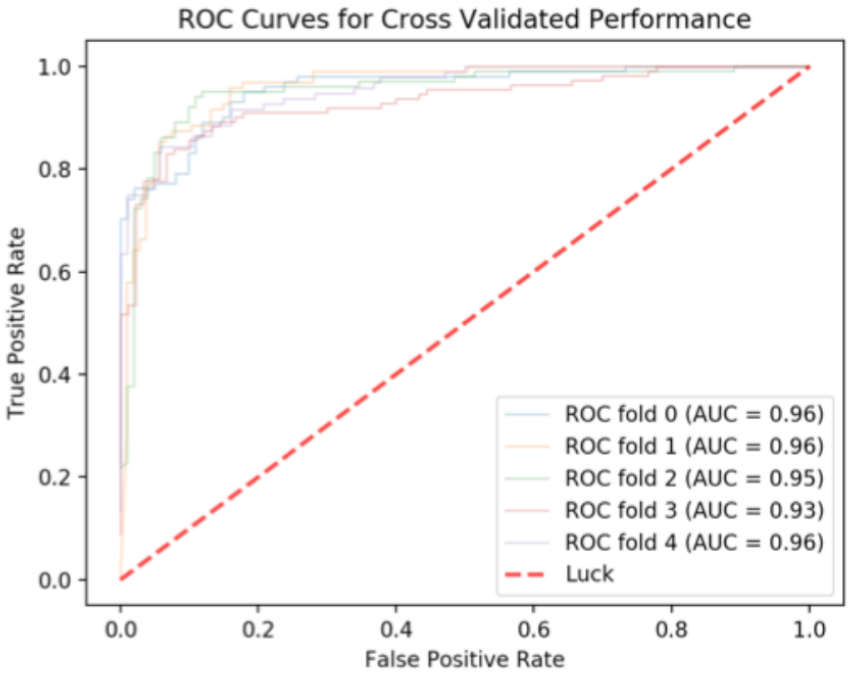


*Figure 11.* ROC curves for 5-Folds Cross Validation Generalization Testing.

Figure 11 shows the ROC curves for the model across all five validation folds. All of the curves display consistent AUC values that are close to 1 (recall, the ideal AUC equals 1). The average AUC is 0.952, which is considerably higher than the 0.632 as measured on the baseline approach, so we can conclude the CNN method is significantly better than the baseline KNN. Also, if we consider random guessing as a baseline, our 0.952 average AUC significantly outperforms the 0.5 AUC from random guessing.

Overall, from the analysis of the results, we can make some educated approximations of the generalization ability of the model and the expected test error. Validation shows that our model can expect about 88% accuracy (12% error), 0.95 AUC scores, and high recall ability when tested on images from the same domain as the training images. Fortunately, when we submitted the predictions for evaluation on the full Kaggle test dataset, our model outperformed expectations with a 93.45% top-1 accuracy on the public leaderboard.

**Failure Analysis**

*Figure 12.* Images correctly classified as Class 0 (true negative).

*Figure 13.* Images correctly classified as Class 1 (true positive).

*Figure 14.* Images incorrectly classified as Class 1 (false positive).

*Figure 15.* Images incorrectly classified as Class 0 (false negative).

Figures 12 - 15 aim to show the types of satellite images which tended to be correctly classified and the types of satellite images which were more likely to be misclassified. The classifier performed best on well lit images with high contrast. Solar panels were more accurately detected on lighter-colored roofs, which lighter-colored solar panels on darker roofs were often unnoticed. This indicates that the classifier underperforms with a more subtle pigment transition. A black-cement parking lot, palm tree, and shadow were all misclassified as solar arrays, indicating that our algorithm is not yet relying on many predictors other than color contrast. These errors could potentially be fixed by determining an RGB color range in which solar panels may be present, so that contrast between the solar panel and its surface is less heavily weighted. More importantly, training the model on a larger set of images will allow it to learn from more nuanced cases. With more training data, and model refinement, the CNN described above has a promising outlook.

## Conclusions

With the rise in popularity of solar energy and solar arrays, there is an increasing desire to understand the regional distributions of solar power consumption. Current methods of querying local governments and combing through survey results is cumbersome and hardly-feasible at scale. One interesting approach to collecting and centralizing the data for solar power distribution is to use machine learning algorithms with satellite imagery to identify the locations of solar arrays. Although this is a multi-facet problem that involves collection, processing, detection, classification, etc., the goal of this work is provide a way to reliably detect the presence of solar arrays in the processed imagery.

Specifically, our method involves training a CNN, as a binary-classifier, on a set of 1,010 labeled satellite images with the goal of ‘teaching’ the model to recognize images that contain solar arrays. We apply several data augmentation steps during training and testing, including flipping, cropping, and rotating to maximize generalization performance. We then evaluated the model performance across 5 folds of cross-validation to estimate the generalization performance. The validation performance yielded an average accuracy rate of 88.12% and average AUC of 0.952 across the 5-Folds. For comparison, the model greatly outperformed a random classification algorithm and a simple KNN classifier, indicating that it is productive and valuable to apply CNNs to this problem. When submitted to the island-in-the-sun Kaggle competition, the model achieved a 93.45% top-1 accuracy result.

The results of the analysis show that the method is promising, but there is definitely room for improvement. As future work, we look to expand the training set, find more effective augmentation techniques, test alternative score fusion methods when testing, and change the model architecture and hyperparameters to maximize the potential of the CNN. With improvements in these areas, it is not unreasonable to expect that the approach of classifying satellite imagery with CNNs will meet, if not surpass human accuracy for this task.

## Roles

**Nathan Inkawhich**: Model Design and Implementation, Cross Validation Generalization Analysis, Writing Contribution

**Lisa Sapozhnikov**: Literature Review and Background Research, Data Exploration and Visualization, Experimental Results Evaluation and Investigation, Writing Contribution

**Yifan Li**: Feature Extraction, Model Training and Testing, Writing Contribution

## References

K. Bradbury, L.M. Collins, J.M. Malof, and R.G. Newell (2016). Automatic Detection of Solar Photovoltaic Arrays in High Resolution Aerial Imagery. CoRR, abs/1607.06029.

A. Fujita, N. Imamoglu, M. Kimura, H. Miyamoto, and R. Nakamura (2017). Solar Power Plant Detection on Multi-Spectral Satellite Imagery using Weakly-Supervised CNN with Feedback Features and m-PCNN Fusion.

V. Golovko, S. Bezobrazov, A. Kroshchanka, A. Sachenko, M. Komar and A. Karachka, "Convolutional neural network based solar photovoltaic panel detection in satellite photos," 2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Bucharest, 2017, pp. 14-19.

Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov 1998. doi: 10.1109/5.726791

J. M. Malof, R. Hou, L. M. Collins, K. Bradbury and R. Newell, "Automatic solar photovoltaic panel detection in satellite imagery," 2015 International Conference on Renewable Energy Research and Applications (ICRERA), Palermo, 2015, pp. 1428-1431.

P. Pai (2017, October 25). Data Augmentation Techniques in CNN using Tensorflow. Retrieved February 10, 2017, from https://medium.com/ymedialabs-innovation/data-augmentation-techniques-in-cnn-using- (https://medium.com/ymedialabs-innovation/data-augmentation-techniques-in-cnn-using-) tensorflow-371ae43d5be9

O, Russakovsky*, J. Deng*, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg and L. Fei-Fei. (* = equal contribution) ImageNet Large Scale Visual Recognition Challenge. IJCV, 2015.

Solar Energy Industries Association. (2017, February 14). U.S. Solar Market Grows 95% in 2016, Smashes Records. Retrieved March 1, 2018, from https://www.seia.org/news/us-solar-market-grows-95-2016-smashes-records (https://www.seia.org/news/us-solar-market-grows-95-2016-smashes-records)