CS230

Solar Digital: Mapping Solar Panels using Satellite Imagery

Ryan Dwyer
Department of Computer Science
Stanford University
rydwy01@stanford.edu

Andy Huynh
Department of Computer Science
Stanford University
avhuynh@stanford.edu

Yash Gupta
Department of Civil and
Environmental Engineering
Stanford University
yashg@stanford.edu

Abstract

Reducing energy production from carbon-emitting energy sources and moving to clean energy sources is vital in the fight against the global climate change crisis. A comprehensive database of the location of solar panels is important to assist analysts and policymakers in defining strategies for further expansion of solar energy. Building a large-scale solar project database through voluntary surveys and self-reports is often incomplete and outdated. We have utilised high-resolution satellite imagery to collect solar installation information by leveraging state-of-the-art deep learning models/techniques. Our work provides an efficient and scalable method for detecting solar panels, achieving an accuracy of **0.99** for classification and an IoU score of **0.81** for segmentation performance.

1 Introduction

As our world is increasingly confronted with the consequences of climate change, many countries attempt to mitigate these effects by rapidly expanding their renewable energy sources and rely less on fossil fuels. Although solar panel production continues to increase, the integration of renewable energy is losing momentum, and carbon emission reduction goals are falling short due to wavering and unsupportive policy frameworks. A comprehensive database of the location of solar panels is essential to assist analysts and policymakers in defining strategies for further expansion of solar energy. Moreover, reducing electricity production from carbon-emitting energy sources requires predicting both the energy demand and the energy supply from natural sources in order to optimise the demand-supply curve.

While all the natural energy sources have inherent uncertainty due to climate/weather conditions, quantifying solar energy prediction requires a complete database containing the accurate locations and size information of solar photovoltaics (PV) installations, especially of distributed rooftop/residential solar panels. Building a large-scale solar project database through voluntary surveys and self-reports is often incomplete and outdated. However, the high-resolution satellite imagery of a country/region, which is regularly updated (in general, annually), offers a rich data source for collecting solar installation information by leveraging state-of-the-art deep learning models/techniques. By combining satellite imagery and deep learning for accurate image classification and segmentation of solar panels, we can construct the solar installation database and utilise it to make better solar energy predictions.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

Moreover, this database can also help improve the accuracy of socioeconomic analyses on solar adoption, thus helping policymakers and solar companies make better strategies & more informed decisions.

2 Relevant Work

Currently, the only significant work in developing solar panel databases is done by the "DeepSolar" project team at Stanford University (Jiafan *et al.*, 2018)[1]. They conducted this study in 2018 to build a large-scale solar photovoltaics map of the United States. While they trained the "Google Inception V3" to classify images containing solar panels, they did not build any supervised learning segmentation model to compute solar panel area. Instead, segmentation results were obtained by setting a threshold to Class Activation Maps generated by aggregating feature maps learned through the convolutional layers. Our work majorly improves upon the segmentation task by building a supervised-learning model using state-of-the-art segmentation architectures and encoder backbones. We also tried to improve the classification accuracy by upgrading the outdated "Google Inception V3" model with some recent CNN architectures.

3 Dataset

Since the classification and segmentation model differs in the sample labels, we used distinct datasets for training and testing of both the models. However, all the images from the segmentation dataset were also used as positive images for the classification dataset. For classification model, our total dataset contains 21,520 unique data points, which consist of satellite images at a specified latitude/longitude coordinates. Out of this dataset, there are 16,140 negative images without a solar panel and 5,380 positive images with a solar panel. We further had 3,716 additional positive images from the segmentation dataset. The dataset was collected from several web resources[5][6] and from collaboration with DeepSolar (Jiafan *et al.* 2018)[1]. 312x312 pixel images were collected from Google Static Map API at zoom level 21. To train the classification model, we considered a training/dev/test split of 90%/5%/5%.

For the segmentation model, we collected a total of 3,716 satellite images along with their corresponding mask labels for segmentation training. For each mask, “0” indicates the background, while the target PV is recorded as "1". This dataset is generated by (Jiang *et al*., 2021) and made openly available on the Zenodo website[7]. To train the segmentation model, we considered a training/dev/test split of 90%/5%/5% for the available 3,716 total aerial images. All images for both models were resized to a size of 224x224 pixels for training and testing.

4 Methods

Our deep learning framework consists of a two-branch model using an image classifier in tandem with a semantic segmentation model. With this framework, when deployed, the model will first classify satellite images as "positive" or "negative" for solar panels. If it is labeled "positive", the model will perform segmentation on the image to determine the solar panel area.

Our classification model uses a ResNet-34 backbone pre-trained on ImageNet. We then trained the model using transfer learning with our satellite imagery dataset. For our training loss of the classification model, we are using binary cross-entropy loss and we evaluate the model using binary accuracy, precision, and recall metrics.

$$\textbf{Binary Cross-entropy Loss: } L(y,p) = -(y\log(p) + (1-y)\log(1-p))$$

$$\textbf{Jaccard Loss: } L(U,V) = 1 - \frac{|U \cap V|}{|U \cup V|}$$

$$\textbf{Accuracy: } Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\textbf{Precision: } Precision = \frac{TP}{TP + FP}$$

$$\textbf{Recall: } Recall = \frac{TP}{TP + FN}$$

$$\textbf{IoU Score: } IoU(U, V) = \frac{|U \cap V|}{|U \cup V|}$$

$$\textbf{F Score: } F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN}$$

In order to quantify the area of solar panels, we have used semantic segmentation on the images that are classified as positive(i.e., containing solar panels). It is a binary segmentation model that assigns all the pixels either label "1" (solar panel) or "0" (non-solar panel). Our model uses U-Net architecture with ResNet-34 as the encoder backbone for segmenting solar panels in satellite images. The training loss for our model is a combination of "Binary cross-entropy loss" and "Jaccard loss". While the Jaccard loss tries to improve on the IoU score, the Binary cross-entropy calculates loss for each pixel and tries to improve on labelling each pixel correctly. To evaluate the performance of our model, we have used IoU and Fscore with a target to improve on the IoU score as much as possible.

5 Experiments

5.1 Classification Model

We used an existing GitHub repository[8] to train our model using transfer learning on a ResNet-34 backbone pretrained on ImageNet. We determined that RestNet-34 outperformed other backbones, such as VGG. Given the large dataset (21,520 images), we used a Keras DataLoader to load each batch (batch size = 32). Through hyperparameter fine-tuning, we determined that the classification model performed best with an Adam optimizer with a learning rate of 1e-04. The model was trained for 2 epochs and the best weights were saved using Keras callback API.

5.2 Segmentation Model

We referred to a GitHub repository[9] which allowed us to explore 4 different model architectures i.e., Unet, Linknet, FPN, and PSPNet for image segmentation. Firstly, we tested all the four architectures available using the same encoder backbone (VGG-16) and found that U-Net performed better than others. We also found that ResNet-34 as the encoder-backbone with a U-Net segmentation model outperformed other backbones such as VGG, EfficientNet, DenseNet and others. Hence, we decided to use ResNet-34 as the encoder-backbone with U-Net model (Figure 1) as our segmentation model architecture for training.

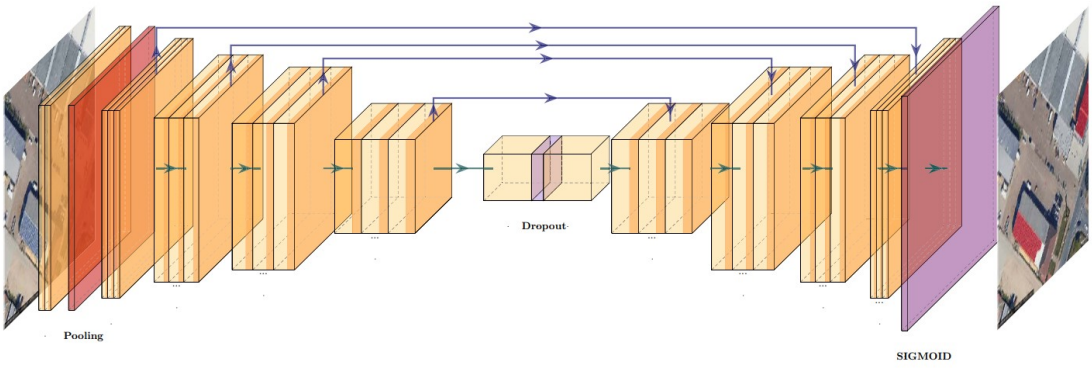

Figure 1: Segmentation Model Architecture

Due to a large amount of training data (3,716 images), we had to build a Keras DataLoader function that allowed us to load each batch (batch size = 16) lazily. We achieved the best performance for training the semantic segmentation model using a learning rate of 1e-04 and the Adam optimizer. We compiled the model by importing the encoder weights pre-trained on ImageNet while the decoder

weights are randomly initialized. Hence, we decided to train only randomly initialized decoder weights and not the properly trained encoder weights to avoid damaging them with huge gradients during the first few steps of training. After fine-tuning all the hyperparameters, we trained the model for 10 epochs and saved the best model using Keras callbacks API.

6 Results and Discussion

Because we are training a CNN model pre-trained on ImageNet with mini-batches of a large dataset, we saw that longer training resulted in overfitting of the model to the training set. Thus, we found that 2 epochs were enough for the classification model to reach maximum performance. The loss, accuracy, precision, and recall metrics for our final model are presented in Table 1. We achieved 0.99 accuracy on the training, validation, and test sets.

Table 1: Classification Model Performance

	Loss	Accuracy	Precision	Recall
Training	0.02	0.99	0.99	0.98
Validation	0.06	0.99	0.98	0.96
Test	0.05	0.99	0.98	0.96

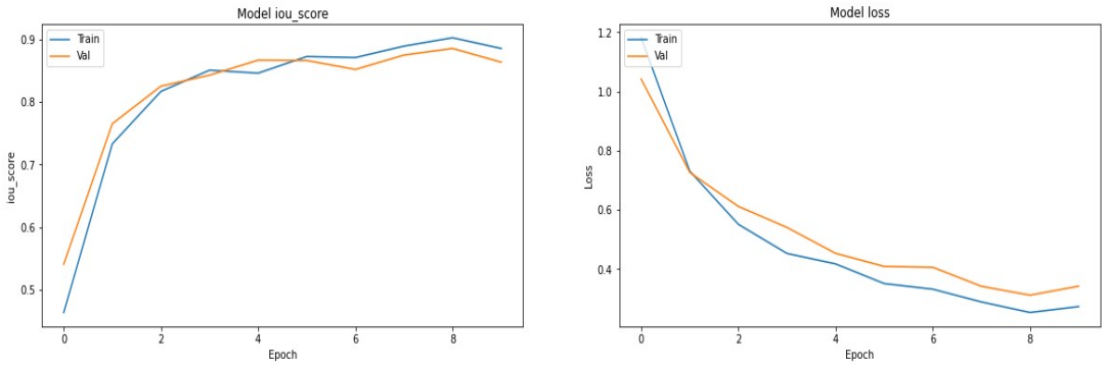

Figure 2: Segmentation Model Training Plots

The semantic segmentation model training plots of IoU scores and loss are presented in Figure 2. We achieved an IoU score of 0.88 on the training set, 0.86 on the validation set and 0.81 on the test set. Appendix 9.1 shows the predicted mask for some samples from the test set while the overall performance of the model is tabulated in Table 1.

Table 2: Segmentation Model Performance

	Loss	IoU Score	F1 Score
Training	0.27	0.88	0.94
Validation	0.34	0.86	0.92
Test	0.40	0.81	0.85

Finally, once we obtain the predicted mask layer for segmented solar panels, its area can be calculated using the Mercator Projection after resizing the mask to the size of the original image. Mercator Projection accounts for the differences in length per pixel depending on the zoom and distance from the equator; thus, the length per pixel is given by:

$$\textbf{meters per pixels} = \frac{156543.03392 * cos(latitude * \frac{\pi}{180})}{2^{zoom}}$$

The total area of solar panels is then computed by multiplying the number of pixels labeled as '1' in the mask layer by the area per pixel value.

7 Conclusions

There is a heightened urgency to solve the climate change crisis by expanding solar energy. Our work solves this challenge by leveraging deep learning methods and implementing a two-branch model using an ResNet-34 classifier with a ResNet-34 encoder-backbone U-Net semantic segmentation model. We achieved an accuracy of **0.99** for classification and an IoU score of **0.81** for segmentation performance. However, this model can have some potentially negative impacts, such as surveillance and investigation of compliance with solar energy mandates, which would disparately impact low-income communities.

8 Contributions

- **Yash Gupta:** Configured Jupyter notebook on AWS instance with GPU access; Tested different model architectures with several encoder-backbones to find the best one; Trained the U-Net (ResNet-34 encoder backbone) segmentation model along with the fine-tuning of its hyperparameters, contributed to submitables (i.e., milestone, final report,etc.).
- **Andy Huynh:** Wrangled and exported complete latitude/longitude data for classification dataset from multiple sources, Collected satellite images using Google Static Map API, Experimented to choose classification model backbone and refine hyperparameters, contributed to submitables (i.e., milestone, final report,etc.).
- **Ryan Dwyer:** Training image augmentation code development, classification model research/implementation, Deep Solar labeled dataset negotiation, contributed to submitables (i.e., milestone, final report,etc.).

References

[1] Jiafan Yu, Zhecheng Wang, Arun Majumdar, Ram Rajagopal, DeepSolar: A Machine Learning Framework to Efficiently Construct a Solar Deployment Database in the United States, Joule, Volume 2, Issue 12, 2018, Pages 2605-2617, ISSN 2542-4351, https://doi.org/10.1016/j.joule.2018.11.021.

[2] Jiang, H., Yao, L., Lu, N., Qin, J., Liu, T., Liu, Y., and Zhou, C.: Multi-resolution dataset for photovoltaic panel segmentation from satellite and aerial imagery, Earth Syst. Sci. Data, 13, 5389–5401, https://doi.org/10.5194/essd-13-5389-2021, 2021.

[3] Wang, Z., Wang, E. & Zhu, Y. Image segmentation evaluation: a survey of methods. Artif Intell Rev 53, 5637–5674 (2020). https://doi.org/10.1007/s10462-020-09830-9

[4] Maxwell, A.E.; Warner, T.A.; Guillén, L.A. Accuracy Assessment in Convolutional Neural Network-Based Deep Learning Remote Sensing Studies—Part 1: Literature Review. Remote Sens. 2021, 13, 2450. https://doi.org/10.3390/rs13132450

[5] `https://zenodo.org/record/5842519`

[6] `https://datasets.wri.org/dataset/globalpowerplantdatabase`

[7] `https://zenodo.org/record/5171712`

[8] `https://github.com/qubvel/classification_models`

[9] `https://github.com/qubvel/segmentation_models`

[10] `https://github.com/albumentations-team/albumentations#list-of-augmentations`

9 Appendix

9.1 Segmentation model's prediction on a few test samples: