Parking Spot Detection Using Deep Learning Computer Vision Applied to Satellite Imagery: Applications for Solar Carport Potential Estimation

Josh Neutel* Renee White* Peiyu Li*
Stanford University
{jneutel, reneedw, peiyu1}@stanford.edu

Abstract

Solar carports are an underutilized technology that offers multiple potential community benefits, including clean energy and shading for cars. We present a deep learning pipeline for quantifying the solar potential of parking lots using satellite imagery. At the core of our approach is a fine-tuned SegFormer model, a Transformer-based segmentation architecture, trained to identify usable parking block areas where solar canopies can be deployed. On the APKLOT dataset, our model achieves a strong performance in structured lot layouts, with a mean intersection over Union (mIoU) of 85% on test images and a parking block IoU of 79%, outperforming the previous baseline of 66%. We apply the model to nearly 17,000 satellite images of parking lots in Los Angeles, estimating an annual solar generation potential of 12.6 TWh. While this pipeline offers a scalable pipeline for solar potential assessment, further improvements could come from custom dataset development, refined irradiance modeling, and incorporating obstructions like trees and buildings. Code is available at: `https://github.com/Peiyu-Li-Tara/CS231-final-project-code`.

1. Introduction

Solar carports are a promising yet underutilized technology, offering multiple community benefits, including clean energy, dual use with electric vehicle charging, and shading for cars. Solar carports are much less widespread compared to rooftop solar, making them an interesting growth opportunity. However, solar carports are comparatively expensive, due to the costs of permitting and the metal structures used to hold up solar carport arrays. In this work, we develop a learning pipeline to estimate the energy potential of solar photovoltaics for parking lots, utilizing deep learning and computer vision applied to satellite imagery. We demonstrate how this pipeline/model can be used to quantify the energy potential for solar carports in a given region, illustrated here for Los Angeles, California.

This work comes at an opportune time in history for several reasons: 1) intensifying global warming continues to motivate the rapid expansion of clean energy, 2) satellite imagery is now highly accessible, using publicly available platforms such as OpenStreetMap (OSM) and Google Earth Engine (GEE), 3) the continued advancement of artificial intelligence – due to improvement in algorithms and access to computation – means deep learning & computer vision may be applied to satellite imagery to estimate solar carport potential at scale.

1.1. Problem Statement

We will focus primarily on parking lot segmentation. Specifically, we train a computer vision Transformer model that - given a satellite image of a parking lot - can segment the parking spot areas usable for solar carports. We then use the segmentation model in a larger pipeline to estimate the energy potential of solar carports in Los Angeles.

2. Related Work

In recent years, computer vision has been applied to satellite imagery to produce estimates of solar *rooftop* potential. Rooftops are an analogous problem to parking lots, providing a good reference point for our work. We have provided a non-exhaustive list of reference papers that estimate solar rooftop potential ([1], [2], [3], [4], [5]). To our knowledge, frameworks developed for rooftop solar have yet to be applied to parking lots.

Reviewing the literature, the following features are typically considered when estimating solar rooftop potential using computer vision, and can be considered for parking lots as well: 1) available area, considering both the perimeter and inner obstructions, 2) orientation of the site to the sun - south-facing is preferred, 3) shading from nearby buildings and trees, and 4) favorable local weather conditions such as strong solar irradiance. Several of these issues have been studied in previous work, most notably in [1], with issues 1-

*Equal contribution

3 being addressed using computer vision applied to satellite imagery.

A challenge unique to solar carports is parking spot segmentation, of relevance because solar carports are typically built where cars are parked [6]. Parking spot segmentation is a crucial first step in estimating solar carports because it is not the size, orientation, and obstruction of entire lots that are of practical importance, but rather that of individual parking spaces, motivating an ability to automatically locate parking blocks and quantify the profitability of each block independently. Given our focus on parking block segmentation using computer vision, the impact of orientation and shading from obstructions on solar potential is ignored for now, with this being a subject for future research.

Parking block segmentation has been attempted in prior studies for various applications, such as security [7] and predicting earnings [8], although studies often utilize surveillance and/or aerial imagery. Hurst-Tarrab et al., however, used satellite imagery for the task of parking block segmentation and released a hand-annotated dataset called APKLOT for this purpose [7]. Along with the release of said dataset, Hurst-Tarrab et al. utilized a convolutional neural network (CNN) based architecture pretrained on ImageNet to achieve 0.658 parking block IoU on the APKLOT test set. A contribution of our work is testing how modern Transformer vision models perform on the parking block segmentation task, a promising approach given progress in Transformer-based vision models since the original APKLOT publication in 2020.

3. Dataset

APKLOT is a satellite imagery dataset developed for the task of parking block segmentation [7]. The dataset was created for a much different purpose: to build models that could supplement surveillance footage for enhanced parking lot security. However, the dataset fits our task reasonably well. It includes high-resolution images of parking lots from diverse geographic regions, with annotated polygon masks over contiguous parking regions or blocks (not individual spots).

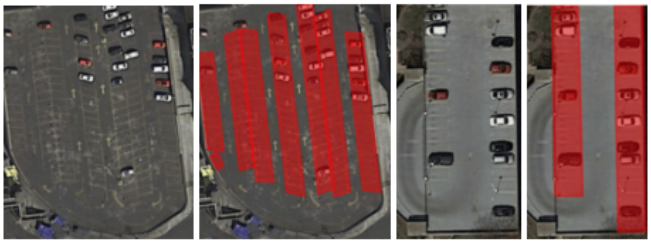


Figure 1: Examples from APKLOT

The original dataset comprises 400 satellite images in PNG format, collected at a zoom level of 14 cm/pixel from OSM. Each image is accompanied by a .json annotation file in Pascal VOC format. Only images of outdoor parking lots captured under clear daylight conditions and with standard demarcation are included. Images with significant occlusions, poor lighting, or non-standard parking layouts are excluded.

Given the small number of annotations (400), we applied data augmentation strategies to each image and polygon mask. We follow the same protocol as described by Hurst-Tarrab et al [7], who employ photometric and geometric transformations designed to simulate real-world satellite image variability. Augmentations were applied consistently to both images and annotations via the `imgaug` library, preserving mask integrity post-transformation. These augmentations were found to enhance the generalizability and robustness of the segmentation model [7].

- **Random Cropping**: Up to 50 pixels from the image borders, simulating satellite view shifts and framing variation.
- **Flipping**: Horizontal and vertical flips (each with 50% probability), ensuring symmetry robustness.
- **Rotation**: Uniform random rotation between $-45°$ and $+45°$, addressing variations in parking lot orientation relative to roads.

The result of the augmentations is a relatively large dataset: $40,000$ augmented samples derived from the original 400 training images, totaling $80,000$ images in total (one input image and one mask for each sample), and 80 GB. Given the size of the data, it was necessary to create our own custom Hugging Face dataset. The dataset is available at `jneutel/APKLOT/CS231N`. While Hurst-Tarrab et al [7] provide original annotations and a recipe for augmentation, they did not provide the full augmented dataset, and thus this is a contribution of our work.

The SegFormer model includes automatic preprocessing applied to each image, including resizing the input to 512×512 pixels using rescaling and padding as needed, as well as applying layer normalization to each image.

4. Methods

4.1. Larger Pipeline for Solar Estimation

First, we describe the larger pipeline for solar estimation.

1. Train a computer vision model that, given a satellite image of a parking lot, can automatically segment parking blocks.
2. Using OSM, identify the longitude and latitude coordinates of bounding boxes for all tagged parking lots in the region of interest. The perimeter areas of most major parking lots in populous cities are tagged in OSM.

3. Using GEE, programmatically pull satellite images for all parking lots identified in Step 2.

4. Apply the computer vision model to create segmentation masks of parking blocks for candidates identified in Step 3.

5. Cross-reference with solar irradiance data from the National Renewable Energy Laboratory (NREL) [9] to estimate the solar potential of each parking lot using the equations below.

$$\text{Solar Potential } [kW]_C = \text{Panel Efficiency} \cdot (\text{Solar Irradiance } [kW/m^2])_C \cdot (\sum_{b \in C} \text{Area}_b[m^2]) \quad (1)$$

where

$$\sum_{b \in C} \text{Area}_b[m^2] = \frac{\text{Square meters}[m^2]}{\text{Pixel}} \cdot (\sum_{p \in C} \text{mask}_C) \quad (2)$$

Here, the solar irradiance of a candidate parking lot C is estimated by plugging in its longitude and latitude coordinates into NREL's Python Web API [9]. The area available for solar panels is the sum of the areas of each parking block, which is equal to the pixel-wise sum of parking block segmentation masks multiplied by a conversion factor of square meters per pixel. Lots are pulled from GEE at the same scale (scale = 0.6), meaning the conversion from pixels to area is the same for all images at inference time ($0.36\ m^2$/ pixel).

4.2. SegFormer Model

Given the small size of the APKLOT dataset, we build upon an existing model called SegFormer that was developed by NVIDIA and pre-trained on a large dataset, fine-tuning the model on our domain-specific data. Code for fine-tuning was developed by us, but with aid from [10].

SegFormer comes in six variants (MiT-B0 to MiT-B5), each scaling in depth, width, and number of attention heads. MiT-B0 is the smallest and fastest, optimized for real-time use, while MiT-B5 is the heaviest in compute and parameters yet most accurate. In benchmark comparisons, SegFormer outperforms previous work, such as traditional CNN-based models and Transformer-based models, in both accuracy and computational efficiency [8]. SegFormer in large part owes its improved performance to a new encoder architecture called a Mix Transformer (MiT), which included several key innovations:

- Unlike prior Vision Transformer (ViT) approaches that only generate single-scale, low-resolution feature maps, MiTs provide high-resolution feature maps at four spatial resolutions. This hierarchy allows the model to retain fine-grained spatial information in early layers while capturing a global context in deeper layers. This approach mimics the effective receptive field (ERF) of CNNs and improves localization in segmentation tasks [8].
- Rather than using non-overlapping patch embeddings commonly used in ViTs, SegFormer employs overlapping patch merging, where small convolution kernels are used to embed patches. This overlap preserves the local neighborhood of information, which is important for precise segmentation boundaries such as those found between parking spots [8].
- To reduce the quadratic computational complexity of vanilla self-attention, SegFormer introduces sequence reduction attention. This technique reduces the spatial token count by applying a learned projection to the key/value tensors. Each encoder stage uses a different reduction ratio, enabling scalable attention while maintaining accuracy [8].
- Instead of relying on positional embeddings that require interpolation when input resolutions change, SegFormer introduces Mix-FNN, a combination of a 3×3 depthwise convolution and a standard multilayer perceptron (MLP) layer. Mix-FNN injects location information implicitly through padding, ensuring robust performance even when test image resolutions differ from training [8].

In addition to changes to the Transformer encoder architecture, SegFormer introduced a simple but effective decoder architecture that comprised of solely multilayer perceptrons (MLPs). This is in contrast to traditional segmentation models that use heavy decoders. The SegFormer decoder fuses the multi-scale feature maps from the MiT encoder by first aligning their channel dimensions through linear projections, then upsampling each to a common spatial resolution, and finally concatenating and projecting them into the output segmentation mask. The decoder design is simple but takes advantage of the rich, multi-level features and large ERF provided by the MiT encoder [8].

4.3. Evaluation Metric

We use intersection over union (IoU) as our primary evaluation metric, as done in the APKLOT paper for which we compare (our baseline), as well as other solar rooftop potential studies [1] and the SegFormer release paper [8]. IoU measures the fraction of overlap between the predicted and ground-truth segmentation regions [7]. We calculate IoU scores for the parking block and background areas separately, with the mean of these two scores referred to as the mIoU.

$$IoU_{\text{Class}} = \frac{\text{True Positive (\# Pixels)}}{\text{Prediction Area} \cup \text{Ground Truth Area (\# Pixels)}} \quad (3)$$

$$mIoU = \frac{IoU_{\text{Parking}} + IoU_{\text{Background}}}{2} \quad (4)$$

4.4. Baseline

We compare our performance to the original APKLOT publication, which used a CNN-based architecture pre-trained on ImageNet [7]. Hurst-Tarrab et al. achieves a 65.8% parking block IoU on the test set. A contribution of our work is testing how modern Transformer vision models perform on the parking block segmentation task, a potentially promising approach given the progress of Transformer vision models since the original APKLOT publication in 2020.

5. Experiments and Hyperparameter Selection

Experiments were conducted to set several hyperparameters, including the learning rate, the number of augmented images to use, and the number of training epochs. During experimentation, models were trained and evaluated using a $95\%/5\%$ split of the data. All models were trained on an A10 chip with 24 GB of VRAM, 30 vCPUs, 200 GB of RAM, and 1.4 TB of SSD storage using Lambda Labs.

First, we set the learning rate schedule. We performed separate linear searches on the learning rate and warmup steps and found that the prior had a noticeable impact on performance, while the latter had a small impact. This stage of experimentation was conducted using one training epoch and 25% of the training data, or a total of $10,000$ images ($9,500$ images for training and 500 images for evaluation). As shown in Figure 2, a learning rate of $5e-4$ results in the best performance and thus is used moving forward.

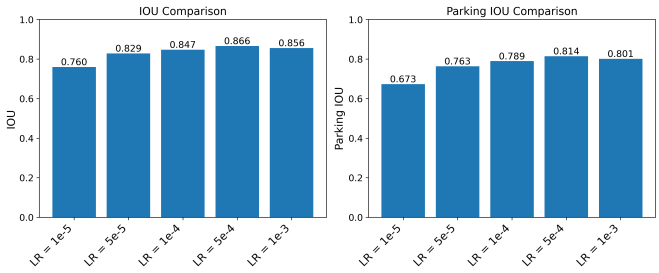


Figure 2: Results of learning-rate experiments

After selecting a reasonable learning rate, we scaled the size of the training data from 25% of the augmented dataset to 50%, 75%, and finally 100%. These tests helped ensure that hyperparameters selected using 25% of the data still performed well with additional data, particularly testing whether SegFormer began to overfit due to recognizing augmentations. As can be seen in Figure 3, additional data only improved performance on the evaluation set, and so moving forward we include all augmentations for a total of $40,000$ images.

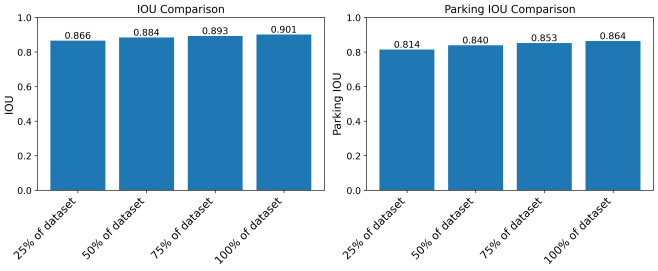


Figure 3: Results of data-scaling experiments

Finally, after fixing the learning rate schedule and the number of augmented images, we increased the number of training epochs to determine how many were needed before performance plateaued. As shown in Figure 4, performance stops improving at five epochs, which is used moving forward.

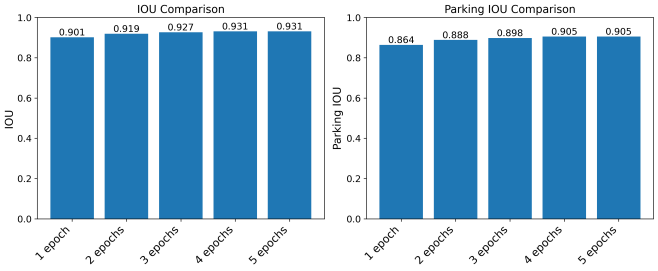


Figure 4: Results of epoch experiments

The final hyperparameters selected are summarized in Table 1. Note that we also tested the impact of weight decay, but its effect was small. This is likely because weight decay only affects the decoder, which has significantly fewer parameters than the encoder (frozen from SegFormer). Training and evaluation batch size were chosen to accommodate computational constraints. Finally, SegFormer model MiT B0 was chosen because performance was acceptable even with the lightweight model, however future work could explore the benefits of using larger models MiT B1-B5. The model outlined here scored a mIoU of 93.1% and parking block IoU of 90.5% when evaluated on the evaluation set.

6. Results and Discussion

6.1. Computer Vision Task

In Table 2, we compare the performance of our preferred SegFormer model with the performance reported in the original APKLOT publication, evaluated on the APKLOT test-set (a holdout of 100 images). As can be seen, our fine-tuned SegFormer model outperforms the APKLOT

Hyperparameter	Choice
SegFormer Model	MiT B0
Number of Images	40,000 (39,600 augmentations)
Learning Rate	5e-4
Learning Schedule	Linear warmup 10% of training steps
Number of Training Epochs	5
Weight Decay	0.01
Training Batch Size	8
Evaluation Batch Size	2

Table 1: Hyperparameters used in the final model.

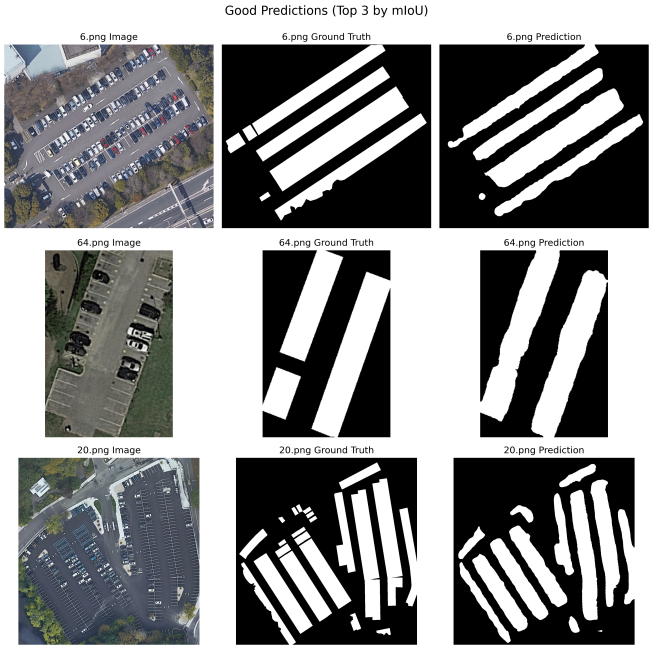


Figure 5: Top 3 well-performing segmentation results based on mIoU. From left to right: input satellite image, ground truth parking block mask, and the predicted mask from our best model.

model by about 12% on parking block IoU. We attribute improved performance to a change in the underlying model architecture, with Hurst-Tarrab et al. using a CNN-based architecture and us using a Transformer-based architecture.

Note that our fine-tuned SegFormer model performs worse on the test set (parking block IoU of 0.790 and mIoU of 0.856) than it does on the evaluation set (parking block IoU of 0.905 and mIoU of 0.930). This is likely because the evaluation set is made up of augmented images, meaning there may be examples in the evaluation set that are similar to examples seen in the training set. By comparison, the test

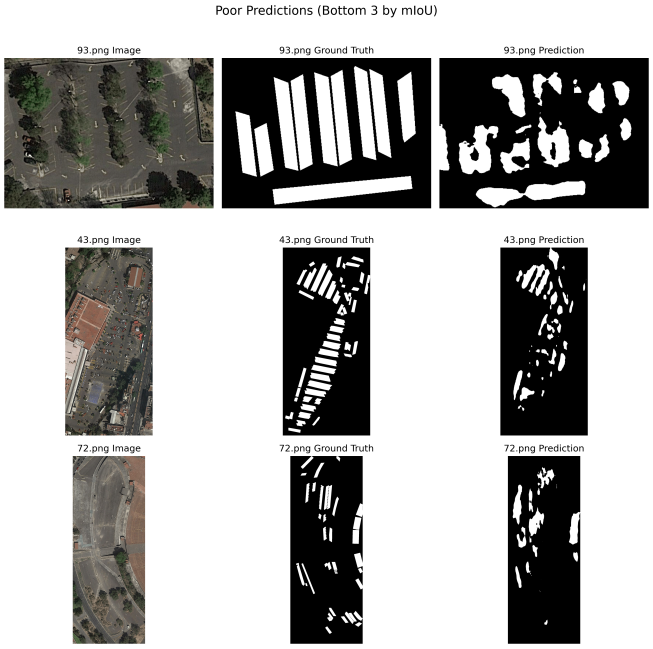


Figure 6: Bottom 3 poor-performing segmentation results based on mIoU. From left to right: input satellite image, ground truth parking block mask, and the predicted mask from our best model.

set is made up of entirely unseen examples.

Given the discrepancy in performance on the evaluation set and the test set, we ran an additional experiment where we trained the model on only the 400 original APKLOT images. As can be seen in Table 2, the SegFormer model performs surprisingly well in this case, with similar performance to the original APKLOT publication, and with about 10% worse performance compared to training SegFormer on the full augmented dataset. That performance is reasonable, even with relatively few examples, and shows the power of pre-training on very large datasets. This also demonstrates the ability of deep neural networks to transfer knowledge from one task to another by extracting key features of images. Overall, these results suggest that augmentations, while helpful, can only bring us so far in the parking block segmentation task. In addition, results motivate hand-labeling of additional data, as even a few hundred additional images may improve performance significantly.

Looking at some qualitative examples from the test set (Figure 5), our model demonstrates strong performance on images that feature clear, orthogonal parking layouts with high-contrast markings and minimal occlusion. In such cases, the regular geometry and visual consistency closely match patterns present in the training data, enabling the model to segment parking lots effectively.

In contrast, the model struggles in scenes with occlu-

Model	Parking Block IoU	Background IoU	mIoU
APKLOT CNN Baseline	0.667	Not Reported	Not Reported
Finetuned SegFormer on Augmented Dataset (40,000 images)	0.790	0.922	0.856
Finetuned SegFormer on Non-Augmented Dataset (400 images)	0.686	0.884	0.785

Table 2: Performance of our model compared to APKLOT baseline [7].

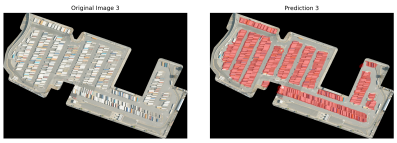
sions, irregular geometries, or underrepresented patterns. In the top right example in Figure 6, for example, parking lines are obscured by large tree shadows, irregular parking places, and overlapping tree branches, leading to both false negatives (missed parking place) and false positives (misclassified background). In the middle example of Figure 6, heterogeneous parking layouts differ significantly from the training distribution, while in the bottom left example, parking lines are curved, sparse, and difficult to identify, resulting in substantial irregularities in predicted mask shape. Overall, these failures demonstrate the following limitations of the current model: 1) sensitivity to occlusion and shadowing from obstructions (trees, buildings); 2) difficulty with curved, rotated, or non-grid-aligned layouts, and 3) lack of robustness to scenes with sparse or subtle parking lines. The reason for such limitations may be due to a lack of occluded and irregular examples in the training dataset, since a large portion of our training data are relatively uniform or grid-aligned structures, leading to inductive bias in our model to favor axis-aligned structures.

6.2. Estimating Solar Potential

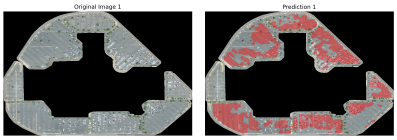
After finalizing the parking block segmentation model, we applied it to estimate the solar potential of all parking lots tagged in OSM for Los Angeles, California. There were a total of 16,941 inference images.

Qualitatively analyzing the performance of the fine-tuned SegFormer model on the Los Angeles inference data, the model generally does well on most parking lots, with many examples looking like Figure 7a, where the model successfully segments the parking blocks. That said, we see worse performance in cases where the inference data looks markedly different to data seen in the APKLOT dataset.

As a common example, lots in the APKLOT dataset are typically of a similar size, and parking lots are all reasonably visible/clear. In our inference dataset, all images are necessarily at the same scale, so solar potential can be estimated downstream. As consequence, some images are very large (for large lots) and some images are very small (for small lots). SegFormer processes these images as needed to meet the 512×512 input specification, zooming out for large images and padding smaller ones. The end result is the model encounters input variations that differ from the training distribution, i.e. lots that are very zoomed out (large lots) or lots that have excessive padding (small lots). Example segmentations for very large and small lots are shown in Figures 7b and 7c respectively. Another example occasionally seen in the inference dataset but not in the APKLOT dataset is parking lots with unclear parking space demarcation lines (Figure 7d), an additional case the model struggles with. As seen in Figures 7b-7d , when the model performs poorly on the inference dataset, it typically underestimates the correct parking block area.

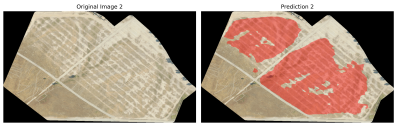


(a) Successful segmentation



(b) Large parking lot

(c) Small parking lot



(d) Unclear parking lines

Figure 7: Performance of fine-tuned SegFormer model on a few examples

Poor performance on large lots is particularly problematic, because we are most interested in large lots that likely have the highest solar potential. In future work, it may be possible to resolve this issue by splitting large lots into a collection of smaller images that are separately processed by the SegFormer model but then grouped together for the solar potential estimation. Poor performance on small lots

is not particularly critical, because very small lots should likely be filtered out of consideration for solar carports altogether. These issues are overall a result of a mismatch between the APKLOT dataset and our end application in certain edge cases, motivating development of a custom dataset that is tailored specifically to the task of estimating the energy potential of solar carports.

Summing all parking block areas and dividing by the total area of all parking lots, parking blocks are estimated to make up 15% of parking area in Los Angeles. Estimating this same fraction for each parking lot and then taking the mean across lots, parking blocks take up an average of 17% of area, with a standard deviation of 9%. These results are lower than those reported by Rudge et al. [6], who estimated this fraction by hand labeling 100 lots in Connecticut, finding that parking blocks make up 35% of parking lots. We understand the discrepancy in results to be due to the SegFormer model underestimating parking block areas for very large lots.

The variability in the fraction of area made up by parking blocks (9% standard deviation) is quite notable, with this result implying that the fraction of area made up by parking blocks can be quite variable between lots. This variability motivates the use of computer vision segmentation models to estimate the fraction of usable area uniquely for each lot, as done here, rather than using a general assumption that is applied to all lots, as done by Rudge et al. [6].

Estimating the solar potential for all parking lots using Equations 1-2, we find that Los Angeles alone has 12.6 TWh/year of solar carport potential. This is equivalent to about 4% of total California electricity usage [11]. It is also enough energy to fully charge a Tesla Model 3 Long Range about 130 million times [12], or enough to power about 2 million average California homes for a year [13]. These results are possibly an underestimate, given the model currently undercounts parking block areas for the largest lots. On the other hand, these results do not consider the impact of orientation on solar irradiance, as the NREL API assumes maximum solar irradiance at all time steps (i.e., it assumes solar panels are placed on a dual-axis tracker that can track the sun on its path through the sky). Overall, these results suggest large impact potential for solar carports, and at least motivate further work on this subject to make these estimations more precise.

7. Future Work

7.1. Larger Model

While the MiT B0 model used here performs well on the APKLOT test set, there may be additional room for improvement by using larger SegFormer models MiT B1-B5. Having additional parameters may make the model more expressive and thus performant, however, it might also cause the model to overfit to training data. Ultimately, the optimal model checkpoint should be determined through experimentation, as in Section 5.

7.2. Custom Dataset

APKLOT was designed for parking block segmentation, not necessarily for solar carport potential estimation. As such, the APKLOT dataset does not include certain edge cases that may be encountered when pulling images from OSM and GEE for solar potential estimation. For example, when estimating solar potential, images must be pulled at the same scale (meter per pixel), which may result in images being very large or small for very large and small lots. In addition, there may be cases where parking lots are pulled without clear parking space line demarcations, which should be accounted for in a solar-carport-specific dataset. Finally, Hurst-Tarrab et al. occasionally segments parking blocks even when there is a clear obstruction (e.g., a tree), whereas a solar-carport-specific dataset may not segment in this case [7]. Given proof of concept on the APKLOT dataset, a reasonable next step would be to hand-label a dataset catered specifically to the task of solar carport energy potential estimation. In addition to catering the dataset toward our specific use case, it would be preferable to expand the size of the base dataset from 400 images, to reduce reliance on augmentation.

7.3. Orientation

Above, we use NREL's Average Direct Normal Irradiance [9], which takes as input the longitude and latitude coordinates of a parking lot and returns the average daily irradiance (mean over hours and days of the year), assuming the solar panels are able to track the sun throughout the day. Solar tracking is only possible using a dual-axis tracker, a relatively expensive option. In all likelihood, solar carport panels would be stationary, and thus their orientation would need to be taken into account.

Previous work estimating solar rooftop potential has used computer vision to determine the orientation of segmentation areas [1]. However, given that GEE images are pulled north-facing, it may be possible to determine the orientation of segmentations by running Principal Component Analysis on predicted masks and then calculating the angle of each parking block relative to due North. Once orientation is determined, solar irradiance can be estimated more accurately using Eq. 5-7 below [1].

$$\text{Solar Irradiance } [kW/m^2] = I_{\text{beam}} + I_{\text{diffuse}} \tag{5}$$

$$\begin{aligned} I_{\text{beam}} = \text{Direct Normal Irradiance} \\ \cdot (\cos(\alpha) \cdot \sin(\beta) \cdot \cos(\psi - \theta) \\ \sin(\alpha) \cdot \cos(\beta)) \end{aligned} \tag{6}$$

$$I_{\text{diffuse}} = \text{Diffuse Horizontal Irradiance} \cdot \frac{1 + \cos(\beta)}{2} \quad (7)$$

Solar irradiance consists of two components: direct irradiance (I_{beam}) and diffuse irradiance from the sky ($I_{diffuse}$) [1]. I_{beam} is affected by the solar elevation angle (α), surface tilt (β), solar azimuth angle (ψ), and surface azimuth angle (θ), whereas $I_{diffuse}$ is only affected by tilt. Here, solar elevation and azimuth angle (α and ψ) are dependent on location (longitude/latitude) and the time of day, surface tilt (β) is commonly $5°-10°$, and the surface azimuth angle (θ) is the orientation from due North. In other words, given the longitude latitude coordinates of a parking lot, all of these parameters are known except for the surface azimuth angle (θ), which is defined by the orientation angle of each parking block.

7.4. Obstructions

In addition to orientation, shading from obstructions such as trees and buildings should be considered to make solar potential estimates more accurate. Computer vision models can be layered on top of the segmentation model shown here to identify building and tree obstructions that can cast shade and thus reduce solar output. The level of detail may vary, from simply identifying trees or buildings that may be problematic, to developing full hourly shading models that incorporate an understanding of obstruction size, height, angle, and distance to parking blocks.

7.5. Scaling

Due to time and storage constraints we limited inference to Los Angeles, California. Future work could include scaling this same process to other major cities in California and across the US.

8. Conclusion

In this work, we demonstrate proof-of-concept for a learning and modeling pipeline that can estimate the solar potential of parking lots using satellite imagery and computer vision. The computer vision task was as follows: given a satellite image of a parking lot, segment the parking blocks (continuous regions where cars can park). To do so, we fine tune SegFormer, a pre-trained segmentation model developed by NVIDIA with a Transformer architecture [8]. We achieve a test-set mIoU of 0.856 and parking block IoU of 0.790, compared to the previously published APKLOT baseline that scored a parking-block IoU of 0.667 [7]. We then applied our fine-tuned SegFormer model to nearly $17,000$ satellite images of parking lots in Los Angeles pulled using GEE and OSM APIs, and estimated potential for 12.6 TWh/year of solar production. While in this work we demonstrate a larger modeling pipeline and establish motivation for continued research in this area, there are several ways the solar potential estimation could be improved, most notably: 1) developing a custom and larger dataset tailored specifically for the task of solar carport potential estimation, 2) updating solar irradiance estimations to account for orientation, and 3) developing a separate computer vision model that can account for shading from obstructions like trees and buildings.

9. Contributions

Peiyu developed and performed data augmentation, created visualization and hyperparameter experiment code, ran training experiments, and tested the model on the APKLOT test dataset, producing relevant figures. Josh defined the project scope and designed the overall model pipeline, helped develop OSM and GEE code to pull inference images, established our first working SegFormer model, and ran some fine-tuning experiments. Renee aided in the pipeline design, researched and chose SegFormer as our base pre-trained model, ran training experiments, lead development and execution of steps 3-5 of the pipeline, produced experiment figures and model tables, and trained the model on the original APKLOT training dataset (400 images). All project partners contributed significantly to the writing the final report, with Josh leading Introduction, Related Work, and discussion of Results, Peiyu leading sections detailing the SegFormer architecture and APKLOT dataset, and Renee editing and formatting all sections.

Libraries Used:

Library	Version
Numpy [14]	1.26.4
Pillow [15]	11.2.1
tqdm [16]	4.67.1
PyTorch [17]	2.7.0+cu126
transformers [18]	4.51.3
datasets [19]	3.6.0
evaluate [20]	0.4.3
torchmetrics [21]	1.7.1
imgaug [22]	0.4.0
matplotlib [23]	3.10.3
pandas [24]	2.2.3
scikit-learn [25]	1.6.1
seaborn [26]	0.13.2

References

[1] Stephen Lee, Srinivasan Iyengar, Menghong Feng, Prashant Shenoy, and Subhransu Maji. Deeproof: A data-driven approach for solar potential estimation

using rooftop imagery. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, pages 2105–2113, New York, NY, USA, 2019. Association for Computing Machinery.

[2] Qingyu Li, Sebastian Krapf, Lichao Mou, Yilei Shi, and Xiao Xiang Zhu. Deep learning-based framework for city-scale rooftop solar potential estimation by considering roof superstructures. *Applied Energy*, 374:123839, 2024.

[3] Zhixin Zhang, Teng Zhong, Min Chen, Zixuan Zhou, Yijie Wang, and Kai Zhang. Estimation of rooftop solar potential using publicly available geodata and deeping learning. In *Applied Energy Symposium 2020: Low Carbon Cities and Urban Energy Systems*, 2020.

[4] Daniel de Barros Soares, François Andrieux, Bastien Hell, Julien Lenhardt, Jordi Badosa, et al. Predicting the solar potential of rooftops using image segmentation and structured data. *NIPS Proceedings*, 2021.

[5] Wenbo Cui, Xiangang Peng, Jinhao Yang, Haoliang Yuan, and Loi Lei Lai. Evaluation of rooftop photovoltaic power generation potential based on deep learning and high-definition map image. *Energies*, 16(18), 2023.

[6] Kieren Rudge. The potential for community solar in connecticut: A geospatial analysis of solar canopy siting on parking lots. *Solar Energy*, 230:635–644, 2021.

[7] Nisim Hurst-Tarrab, Leonardo Chang, Miguel Gonzalez-Mendoza, and Neil Hernandez-Gress. Robust parking block segmentation from a surveillance camera perspective. *Applied Sciences*, 10(15):5364, 2020.

[8] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers, 2021.

[9] National Renewable Energy Laboratory. Solar resource data api. `https://developer.nrel.gov/docs/solar/solar-resource-v1/`, 2025. Accessed: 2025-06-03.

[10] Nicolas Patry. Fine-tune segformer for semantic segmentation. `https://huggingface.co/blog/fine-tune-segformer`, 2022. Accessed: 2025-06-03.

[11] California Energy Commission. 2022 total system electric generation. `https://www.energy.ca.gov/data-reports/energy-almanac/california-electricity-data/2022-total-system-electric-generation`, 2022. Accessed: 2025-06-04.

[12] EnergySage. How much does it cost to charge a tesla?, 2024.

[13] Kelly Bedrich and Rebecca Bridges. Average electricity bill, usage and price per kwh by state (june 2025), 2025.

[14] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[15] Alex Clark. Pillow (pil fork) documentation, 2015.

[16] Casper da Costa-Luis, Stephen Karl Larroque, Kyle Altendorf, Hadrien Mary, richardsheridan, Mikhail Korobov, Noam Yorav-Raphael, Ivan Ivanov, Marcel Bargull, Nishant Rodrigues, Shawn, Mikhail Dektyarev, Michał Górny, mjstevens777, Matthew D. Pagel, Martin Zugnoni, JC, CrazyPython, Charles Newey, Antony Lee, pgajdos, Todd, Staffan Malmgren, redbug312, Orivej Desh, Nikolay Nechaev, Mike Boyle, Max Nordlund, MapleCCC, and Jack McCracken. tqdm: A fast, extensible progress bar for python and cli, November 2024.

[17] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[18] Hugging Face. Hugging face transformers. GitHub repository, 2018. State-of-the-art natural language processing library.

[19] Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu,

Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[20] Hugging Face. evaluate: A library for easily evaluating machine learning models and datasets. `https://github.com/huggingface/evaluate`, 2022. Accessed: 2025-06-03.

[21] Nicki Skafte Detlefsen, Jiri Borovec, Justus Schock, Ananya Harsh, Teddy Koker, Luca Di Liello, Daniel Stancl, Changsheng Quan, Maxim Grechkin, and William Falcon. Torchmetrics - measuring reproducibility in pytorch. *Journal of Open Source Software*, 7(71):4101, 2022. `https://github.com/Lightning-AI/torchmetrics`.

[22] Alexander B. Jung. imgaug. `https://github.com/aleju/imgaug`, 2018. [Online; accessed 30-Oct-2018].

[23] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[24] The pandas development team. pandas-dev/pandas: Pandas, February 2020.

[25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[26] Michael L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021.