Deep Learning for Photovoltaic Characterization

Adrian Manuel de Luis Garcia
University of Arkansas-Fayetteville

Follow this and additional works at: https://scholarworks.uark.edu/etd

Part of the Artificial Intelligence and Robotics Commons, OS and Networks Commons, and the Systems Architecture Commons

Citation

This Thesis is brought to you for free and open access by ScholarWorks@UARK. It has been accepted for inclusion in Graduate Theses and Dissertations by an authorized administrator of ScholarWorks@UARK. For more information, please contact scholar@uark.edu.
ABSTRACT

This thesis introduces a novel approach to Photovoltaic (PV) installation segmentation by proposing a new architecture to understand and identify PV modules from overhead imagery. Pivotal to this concept is the creation of a new Transformer-based network, S3Former, which focuses on small object characterization and modelling intra- and inter-object differentiation inside an image.

Accurate mapping of PV installations is pivotal for understanding their adoption and guiding energy policy decisions. Drawing insights from current Deep Learning methodologies for image segmentation and building upon State-of-the-Art (SOTA) techniques in solar cell mapping, this work puts forth S3Former with the following enhancements:

1. Contrary to popular implementations for PV segmentation, S3Former eliminates the need for a classifier network and focuses on learning strong representations at the segmentation network.

2. S3Former introduces a pioneering Transformer-based architecture featuring a Mask-Attention Transformer Decoder. The novel attention mechanism adeptly captures relationships within high-resolution features, accurately identifying minute solar cells and excelling at contextual understanding around PV installations to prevent mischaracterization of background elements with similar features.

3. Adds a Self-Supervised component to enhance feature extraction at the backbone level and create stronger representations for the down-stream segmentation task.

Validation of the proposed method is undertaken through extensive experiments on three annotated datasets—one from California and two from France. This diverse set of backgrounds and PV characteristics ensures the robustness of our method in addressing solar PV segmentation challenges. Benchmarking S3Former against current SOTA methods and popular networks for semantic segmentation reveals superior performance across widespread metrics.
In conclusion, this work presents a new pathway for accurately mapping current solar installations, contributing to a deeper understanding of solar energy extension. We hope that the methods and processes described in this work contribute to reducing the impact of PV installations on the grid and, ultimately, create a pipeline for automatically detecting solar cells in the future.
© 2023 by Adrian Manuel de Luis Garcia
All rights reserved
ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to the individuals whose contributions have been instrumental in making this thesis a reality.

First and foremost, I would like to thank my advisor and supervisor Ngan Le. Dr. Le’s support throughout my research at the University of Arkansas has made this thesis and all of my research this year possible. Her consistent motivation and mentorship have played a pivotal role in my academic success and personal development over the past two years. Dr. Le’s dedication to her students and her hard-working mentality serve as a true inspiration to everyone. Thank you, Dr. Le.

Secondly, I would like to thank everyone at the AICV lab. The presence of such brilliant and thoughtful individuals has greatly contributed to my personal growth. I extend special thanks to Minh Tran for his steadfast support and valuable advice during my research. His foundational work on semantic segmentation proved to be critical for my own research, and I can confidently assert that this thesis would not have been possible without his contributions.

I am also grateful to the University of Arkansas, Dr. Panda, and Dr. Di for affording me the opportunity to pursue my Master’s in Computer Science. Special thanks are due to Dr. Mantooth and Dr. McCann, who generously devoted their time and provided valuable critiques for my thesis.

Finally, heartfelt appreciation goes to all my family and friends for their unwavering support throughout this journey. The challenge of uprooting from Spain, my lifelong home, was undoubtedly a very difficult one. However, the support of newfound friends and the encouragement from my lifelong companions made this transition not only possible but also enriched the entire experience.
TABLE OF CONTENTS

Chapter I: Introduction and Background ................................................. 1
  1.1 Photovoltaic Energy ................................................................. 1
  1.2 Deep Learning .......................................................................... 3
  1.3 Deep Learning for PV analysis ................................................. 12
  1.4 Thesis Structure ...................................................................... 15
Chapter II: Methodology ....................................................................... 17
  2.1 SolarFormer Framework ............................................................ 17
  2.2 SolarFormer Architecture ......................................................... 18
  2.3 Transformer-Decoder Loss ......................................................... 22
  2.4 S3Former Framework ............................................................... 25
Chapter III: Experiment Results ............................................................. 28
  3.1 Remote Sensing Datasets ......................................................... 28
  3.2 Experiment Set-Up .................................................................. 30
  3.3 Augmentations ....................................................................... 30
  3.4 Metrics .................................................................................. 31
  3.5 Quantitative Analysis ............................................................... 32
  3.6 Qualitative Analysis ................................................................. 36
  3.7 Ablation Studies .................................................................... 41
Chapter IV: Conclusion and Future Work ................................................. 43
Bibliography ...................................................................................... 46
<table>
<thead>
<tr>
<th>Number</th>
<th>Illustration Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Examples of different PV modules currently available on the market. The % indicates the efficiency of each material (adopted from Clean Energy Review, 2023 n.d.).</td>
<td>2</td>
</tr>
<tr>
<td>1.2</td>
<td>Example of an MLP layer. We include the same nomenclature used on Equation 1.2. Typical activation functions are also listed (adopted from Different Activation Function, 2018 n.d.).</td>
<td>6</td>
</tr>
<tr>
<td>1.3</td>
<td>Example of an Convolution operation. Pictured here is the application of a convolution in a 2D-grid and how it translates to multiple channels (adopted from Altaf et al., 2019).</td>
<td>7</td>
</tr>
<tr>
<td>1.4</td>
<td>Equivalent circuit representing the three parameter model.</td>
<td>12</td>
</tr>
<tr>
<td>1.5</td>
<td>Examples of challenging characteristics of solar PV segmentation. Within a class, there is a large diversity in appearance: intra-class heterogeneity (red); some different classes share the similar appearance: inter-class homogeneity (green), solar PV are dense and small such that they are hardly identifiable (blue).</td>
<td>15</td>
</tr>
<tr>
<td>2.1</td>
<td>SolarFormer framework: a semantic segmentation network for solar installations. Top component represents our implementation, bottom is the SOTA approach.</td>
<td>17</td>
</tr>
<tr>
<td>2.2</td>
<td>Overall network architecture of our proposed SolarFormer which consists of three components i.e., Backbone, Pixel Decoder, and Mask-attention Transformer Decoder.</td>
<td>18</td>
</tr>
<tr>
<td>2.3</td>
<td>Mask-attention Transformer Decoder. This module represents the transformations described on Equation 2.4.</td>
<td>21</td>
</tr>
<tr>
<td>2.4</td>
<td>Hierarchical comparison of the S3Former framework, with the top component depicting our enhanced implementation, the middle segment representing SolarFormer, and the bottom illustrating the SOTA approach.</td>
<td>26</td>
</tr>
<tr>
<td>2.5</td>
<td>Self-distillation with no labels using Remote Sensing images. For simplicity, only one view per network is included.</td>
<td>27</td>
</tr>
<tr>
<td>3.1</td>
<td>Qualitative comparison on (a) IGN France, (b) GGE France, (c) USGS California. From top to bottom: Original RGB Image, Groundtruth, Upernet (Xiao et al., 2018) and DeepLabv3+ (L.-C. Chen et al., 2017) and our S3Former.</td>
<td>38</td>
</tr>
</tbody>
</table>
3.2 Extended qualitative comparison on (a) IGN France, (b) GGE France, (c) USGS California. From left to right: RGB Image, Ground-Truth, SolarFormer and our S3Former. We selected special cases highlighting the strength of both models and improvements of S3Former with respect to SolarFormer on: inter-class homogeneity (green), intra-class heterogeneity (red), and small-object identification (blue). We further highlight cases of missing annotations on the data (purple).

4.1 Number of recent publications in SSL and SSL learning in recent years (Y. Wang et al., 2022).
<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Datasets characteristics comparison. &quot;+ Samples&quot; indicate images containing solar panels, while &quot;- Samples&quot; lack solar arrays.</td>
<td>29</td>
</tr>
<tr>
<td>3.2</td>
<td>Performance comparison between our S3Former and SolarFormer with existing <strong>DL-based Solar PV profiling methods</strong> on GEE, France dataset.</td>
<td>33</td>
</tr>
<tr>
<td>3.3</td>
<td>Performance comparison between our S3Former and SolarFormer with existing <strong>DL-based Solar PV profiling methods</strong> on IGN, France dataset.</td>
<td>33</td>
</tr>
<tr>
<td>3.4</td>
<td>Performance comparison between our S3Former and SolarFormer with existing <strong>DL-based Solar PV profiling methods</strong> on USGS, California dataset.</td>
<td>33</td>
</tr>
<tr>
<td>3.5</td>
<td>Performance comparison between our S3Former and SolarFormer with existing <strong>DL-segmentation Networks</strong> on various <strong>backbones</strong> on GEE, France dataset.</td>
<td>35</td>
</tr>
<tr>
<td>3.6</td>
<td>Performance comparison between our S3Former and SolarFormer with existing <strong>DL-segmentation Networks</strong> on various <strong>backbones</strong> on USGS, California dataset.</td>
<td>36</td>
</tr>
<tr>
<td>3.7</td>
<td>Performance comparison between our S3Former and SolarFormer with existing <strong>DL-segmentation Networks</strong> on various <strong>backbones</strong> on IGN, France dataset.</td>
<td>37</td>
</tr>
<tr>
<td>3.8</td>
<td>Ablation study performed on GEE, France dataset on three different augmentations.</td>
<td>41</td>
</tr>
<tr>
<td>3.9</td>
<td>Ablation study performed on USGS, California dataset on three different augmentations.</td>
<td>41</td>
</tr>
<tr>
<td>3.10</td>
<td>Ablation study performed on IGN, France dataset on three different augmentations.</td>
<td>41</td>
</tr>
</tbody>
</table>
Chapter 1
INTRODUCTION AND BACKGROUND

As climate change effects intensify, the global imperative to shift towards sustainable energy sources becomes more pronounced. While the share of fossil fuels used for electricity generation decreases, renewable energy continues to increase. Photovoltaic (PV) energy has become one of the preferred options by users thanks to its reliability, easy installation and growing competitive market. The first chapter on this thesis, Section 1.1 will provide an overview of PV energy, detailing its generation process and exploring various PV installations types. In Section 1.2 we introduce Deep Learning and present a concise overview enumerating implementations for Multilayer Perceptrons (MLP), Convolutional Neural Networks (CNNs) and Transformers. Afterwards, Section 1.3 is aimed at merging both Deep Learning (DL) and PV characterization, offering a comprehensive analysis of DL methods applied to challenges in PV mapping and characterization. Lastly, Section 1.4 discusses the methods employed on my research as well as its contributions for PV installation segmentation. We believe the methods included on this work will help further the characterization of PV installations and contribute to the advancement of its analysis to combat climate change.

1.1 Photovoltaic Energy

Photovoltaic (PV) energy is generated by arrays of solar cells joined together. Solar cells can create electricity by converting sunlight into photocurrent and photovoltage (Dittrich, 2018) and are typically composed of semiconductor materials. The semiconductor materials employed originate from three distinct categories: 'III-V,' 'II-IV,' or 'IV.' Each designation corresponds to a specific amalgamation of elements derived from their respective groups on the periodic table (Green et al., 2021).

Due to the enormous availability and reliability of such materials, improved cell manufacturing (Peters et al., 2019), competitive market, and increasing energy costs (Lemay, Wagner, and Rand, 2023) PV energy has become one of the preferred options for renewable energy sources. In 2022, the U.S. Energy Information Administration reported a 440% increase in small-scale PV installation
capacity, reaching 39.5 GW from 7.3 GW in 2014 (EIA, 2022 n.d.). Some of the use cases for PV installations include water pumping, electrodialysis (Maka and Alabid, 2022), zero-energy building (Kylili and Fokaides, 2014), etc. It is therefore anticipated that PV will play a significant role in the long-term world electricity-generation as a sustainable alternative to fossil fuels.

**Photovoltaic Installations**

Photovoltaic (PV) systems are heavily weather-dependent, with their energy yield tied to humidity variations (Kazem and Chaichan, 2015), wind speed (Cludius et al., 2014), solar irradiance and other factors such as materials occluding the solar cells (Silvestre et al., 2018; M. Chen et al., 2011) or small cracks (Dhimish, Holmes, Mather, et al., 2018; Dhimish, Holmes, Dales, et al., 2017). Tilt and azimuth angles are modifiable parameters that greatly affect the total annual energy production of a solar module. Both angles impact the total solar energy absorbed by a PV cell. Tilt is the angle of the PV installation from the horizontal plane for a fixed mounting (Alsadi and Nassar, 2017) and the azimuth angle is the angle of the PV modules relative to the direction due south, i.e. $0^\circ$ is south, $+90^\circ$ is west and $-90^\circ$ is east (Sidek et al., 2017).

Authors have proposed different methods for calculating the efficiency of a PV module, which is typically expressed as a percentage. The efficiency is defined as:

$$\text{Efficiency(%) } = \frac{P_{\text{max}}}{A_{\text{area}} \times A_{\text{irr}}} \times 100$$

where $A_{\text{area}}$ is the area of the solar module, $P_{\text{max}}$ is the Maximum Power of the PV installation and $A_{\text{irr}}$ is the solar irradiance at 1000 W/m$^2$.
recommending tilt and azimuth angles, either by geographical location (Le Roux, 2016; Mondol, Yohanis, and Norton, 2007), weather conditions (Cludius et al., 2014; Armstrong and Hurley, 2010), or implementing sun-tracking systems (Antonanzas et al., 2018).

Correctly characterizing the azimuth and tilt angles of PV installations is pivotal for estimating the PV production of solar cells. Researchers have leveraged these angles to assess the viability of fixed rooftop PV systems within specific areas, as highlighted by (Hong et al., 2017). For instance, neighboring rooftop PV modules may exhibit entirely distinct azimuth and tilt angles owing to variations in roof orientations. This variability presents a considerable challenge when estimating the overall PV output within a given area. The diversity in installation configurations underscores the complexity associated with predicting solar cell production and emphasizes the need for robust methodologies that can accommodate such intricacies in order to enhance the accuracy of PV output estimations.

1.2 Deep Learning

Machine Learning (ML) has been successfully integrated into applications performing statistical analysis on non-linear data. Deep Learning (DL) is an ML-based algorithm capable of learning complex representations of data through the use of linear and non-linear neural network layers. To do so, raw data is fed to the network to automatically discover a representation suitable for a given task, which includes detection, classification, segmentation, etc. As data flows through different layers, the non-linear transformations generate more abstract representations. For classification tasks, for example, the higher layers amplify aspects of the data for discrimination and avoid learning irrelevant features.

One of the mainstream approaches to creating these architectures is supervised learning. For this approach, we have an input set of observations $X = \{ x^{(n)} \}_{n=1}^{D}$ paired with its corresponding labels $Y = \{ y^{(n)} \}_{n=1}^{D}$, which forms a dataset $D = \{ (x^{(n)}, y^{(n)}) \}_{n=1}^{D}$. We later can further divide this set in two folds: train and test. We will use the train fold to learn representations of the input observations in our network, while the test fold is an unseen distribution used for validating our model. We can quantify the distance between the empirical distributions using a loss function such as:
\[
J = \frac{1}{N} \sum_{i=1}^{N} L(y^{(i)} - \hat{y}^{(i)})
\]

where \( \hat{y} \) represents the output from a DL model. We can express \( \hat{y} \) in terms of the input observations \( x \) using function \( g \). \( L \) is the corresponding loss function, which measures the similarity between the predictions \( \hat{y} \) and the Ground Truth (GT) values \( y \). Our goal is empirical risk minimization, seeking function \( g \) that best fits the distribution of the training data. To achieve this, we may expand \( g \) as:

\[
\hat{y} = f_l(z_l)
\]

\[
z_l = W_{l-1}a_{l-1}
\]

\[
a_{l-1} = f_{l-1}(z_{l-1})
\]

... \[
z_0 = W_0x
\]

where \( \hat{y} \) is the output of the model, \( l \) denotes the neural network layer at which each component is located, \( f \) is the non-linear activation function and \( z \) is the result of the product between the linear weights \( W \) and the input \( a \). This expression can be expanded all the way to input observation \( x \). This process is commonly referred as forward propagation, and includes all operations transforming the input data all the way to the prediction and finishes when a loss value is calculated.

However, this first step doesn’t allow our model to learn. To create a more accurate representation of \( y \) from \( x \) we need to fine-tune the value of the weights \( W \) at each layer. We can do so by applying a numerical method called Gradient Descent (GD). The objective of this function is to iteratively minimize \( L \) until a converge value is reached. We can denote this operation as:
\[ W_l \leftarrow W_l - \eta \nabla L(W_l) \]
\[ \nabla L(W_l) = \begin{bmatrix} \frac{\delta L}{\delta W_{l1}} \\ \vdots \\ \frac{\delta L}{\delta W_{lm}} \end{bmatrix} \] (1.3)

where each of the weights at layer \( l \), \( W_l \) are updated after each iteration by a step learning rate \( \eta \). \( \nabla L(W_l) \) represents the partial derivative of the loss \( L \) with respect of the weights \( W \) at layer \( l \).

We can expand this expression through all the layers by applying the chain rule all the way to the input. This process is usually referred to as \textit{backward pass}, and allows our model to create more accurate representations of the distribution of our data. This process can be repeated as many times as necessary to reach a convergence value. The data may also be shuffled to avoid cyclic patterns.

For a classification problem, we consider that the label \( y \) is drawn from a categorical distribution that follows class probabilities \( \pi_1, \ldots, \pi_k \), i.e., \( y_i \sim \text{Dog}(\pi_i) \), which can be represented by the output of the softmax function. The softmax function takes a vector \( z \in \mathbb{R}^k \) with \( k \) values as its input and normalizes it into a probability distribution consisting of \( k \) probabilities. We apply this function to \( \hat{y} \) to model the distribution of \( y \), i.e., \( \hat{y} = \text{softmax}(\hat{y} = f_l(z_l)) \). The softmax function is defined as follows for \( k \geq 1 \):

\[
\text{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)} \quad \text{for} \quad i = 1, \ldots, k
\] (1.4)

In the following subsections, we will discuss different implementations of popular Deep Learning architectures.

**Multilayer Perceptron**

The Multilayer Perceptron (MLP) is a type fully-connected feedforward artificial neural network (ANN). It is one of the bases for Deep Neural Networks (DNN) and Deep Learning. The output of an MLP is usually a combination of a linear and non-linear transformations from the different layers of the network. MLP neuron blocks are represented by:
Figure 1.2: Example of an MLP layer. We include the same nomenclature used on Equation 1.2. Typical activation functions are also listed (adopted from Different Activation Function, 2018 n.d.).

\[
\text{MLP}(a_{l+1}) = f_{l+1}(\sum_{i=1}^{M} w_{il}a_{li} + w_{i0}b_{i})
\]  

(1.5)

Where \(\text{MLP}(a_{l+1})\) is the output of neuron \(a\) at layer \(l+1\), \(f\) is the non-linear activation function, \(W_i\) are the linear transformation weights from layer \(l\) to \(l+1\), \(a_i\) is the output from the previous layer and \(b_i\) is the bias term added. Combined with the backpropagation algorithm described on Section 1.2, MLPs can learn complex representations and non-linear models. Some of the most common activation functions include ReLU and its variations, tanh, sigmoid and Maxout.

**Convolutional Neural Networks**

Convolutional Neural Networks (CNNs) are a type of feedforward neural network proficient in feature extraction through convolution operations. They enhance the MLP architecture by addressing three pivotal aspects:

1. **Local-Connections**: Neurons exhibit selective connections to the preceding layer, diminishing the overall parameter count in the network.
2. **Weight sharing**: Clusters of connections share identical weights, further curtailing the total number of parameters.

3. **Down-sampling dimensionality reduction**: Pooling layers retain information while down-sampling the input size of data.

These properties allow CNNs to be a valuable resource for feature extraction. Their success stems from a design that enables the automatic and adaptive learning of spatial hierarchies of features. This particular trait proves exceptionally beneficial in image processing, where the spatial relationships among pixels encapsulate crucial information regarding the image’s features. The expression of a 2D image convolution can be formulated as follows:

$$y_{i,j}^{(l)} = \sum_{u=1}^{k_h} \sum_{s=1}^{k_w} H_{u,s} x_{(i+u),(j+s)}^{(l-1)} + b$$

(1.6)

where $H_{u,s}$ represents the convolution kernel for the spatial dimension from layer $l-1$ to $y_{i,j}^{(l)}$, $b$ is the bias term and $x$ represents the input from layer $l-1$. This operation can be seen on Figure 1.3, where the kernel influences the shape of the output feature map. Adjusting the kernel parameters...
such as size or stride influences the perceived area and changes the density and characteristics of the extracted features. We can further diminish this output by using a pooling layer. Authors have explored the applications of different convolutions such as deformable (Dai et al., 2017), atrous (L.-C. Chen et al., 2017), separable (Chollet, 2017) and multi-dimensional (C. Li, Zhou, and Yao, 2022).

**RNN**

Recurrent Neural Networks (RNNs) are another popular Neural Network. It employs a sequential data architecture and feeds the output from the previous step to the current stage. This architectural design accommodates input sequences of arbitrary lengths and incorporates a "memory" mechanism, retaining information from prior states. In the context of a current state represented by $h_t$ and an output state denoted as $y_t$, the recursive calculation of an RNN is articulated as follows:

$$
  h_t = \sigma(W_{xx}h_{t-1} + W_{hx}x_t + b_{th}) 
$$

(1.7)

$$
  y_t = \sigma(W_{yx}h_t + b_{ty}) 
$$

(1.8)

where $W_{xx}, W_{hx}, W_{yx}, b_{th}, b_{ty}$ are coefficients shared temporally and $\sigma$ is the activation function. RNNs architecture has excelled at predicting the next words in a sentence or image to text translation as demonstrated by (Lipton, Berkowitz, and Elkan, 2015). Despite being adept at capturing long-term dependencies, RNNs traditionally grapple with data sequences that are temporally distant. Included below are some architectures aiming to fix these issues:

**LSTM:** long short-term memory (LSTM) uses special hidden units to store long-term information. First proposed by (Hochreiter and Schmidhuber, 1997), the information flow within a cell is governed by three gates: (i) a 'Forget Gate' ($f_t$) that discerns the unimportant information from the preceding cell, (ii) an 'Input Gate' ($i_t$) regulating the information to be incorporated into the current cell, and (iii) an 'Output Gate' ($o_t$) overseeing the output. The expressions for these gates
can be articulated as follows:

\[ i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \]
\[ f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \]
\[ o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \]

\[ \tilde{c}_t = \tanh(W_c[h_{t-1}, x_t] + b_c) \]
\[ c_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t \]
\[ h_t = o_t \cdot \tanh(c_t) \]  (1.10)

As seen on Equation 1.10, the state of the current cell \( c_t \) is aware of which information it has to forget from the previous cell \( f_t \) and which information it has to aggregate from \( i_t \) and the candidate hidden state \( \tilde{c}_t \). The updated cell \( h_t \) value is then leveraged from the current state \( c_t \) and the output gate \( o_t \). This structure adjusts long-term and short-term information for an optimised solution. Bidirectional LSTM add more complexity to the network by allowing data to flow in two directions by adding a LSTM layer. This extension are trained to predict positive and negative time directions and can increase the performance of traditional LSTM for sequence classification (Siami-Namini, Tavakoli, and Namin, 2019).

**GRU**: Gated Recurrent Units (GRUs). Similarly to LSTM, GRUs constitute a recurrent network that incorporates gated mechanisms to regulate the information flow between cells, as proposed by (Cho et al., 2014). The updated architecture eliminated the output gate and replaced the input and forget gates in favor of update and reset gates. GRUs focus on learning dependencies from long-range relationships rather than forgetting information from earlier segments of the sequence. The denotation for this architecture is expressed as:

\[ z_t = \sigma(W_z[h_{t-1}, x_t] + b_z) \]
\[ r_t = \sigma(W_r[h_{t-1}, x_t] + b_r) \]  (1.11)
\[ h_t = \tanh(Wx_t + r_t \cdot h_{t-1}) \]

\[ h_t = z_t \cdot h_{t-1} + (1 - z_t) \cdot \tilde{h}_t \] (1.12)

where the updated cell \( h_t \) values are calculated performing less operations than LTSM. Although performance-wise GRU might not outperform LTSM except for smaller and less frequent datasets (Chung et al., 2014), they are faster to compute and more memory efficient while maintaining comparable performance.

**Transformer**

Transformers (Vaswani et al., 2017) have recently attracted significant interest in the research community. Initially implemented for Natural Language Processing (NLP) tasks, the Transformer incorporates a self-attention mechanism to enhance learning on long-range relationships inside the data. The Transformer architecture exhibits adaptability for parallelization, facilitating expedited training across multiple nodes, particularly beneficial for handling extensive datasets. The foundational structure adheres to a standard encoder-decoder network configuration, with the combination of a multi-head self-attention mechanism, a positional encoding and a position-wise feed-forward networks.

The multi-head attention mechanism is based on:

\[
\text{Attention}(Q, K, V) = \text{softmax} \left( \frac{QK^\top}{\sqrt{d_k}} \right) V 
\] (1.13)

where \( Q, K \) and \( V \) represent sets of queries, keys and values packed together and \( d_k \) is the dimension of \( Q \) and \( K \) vectors. Instead of implementing a single attention function, learned linear projections from \( d_k \), \( d_k \) and \( d_v \) dimensions show better performance. The query, key, and value tensors are created by linearly projecting the input with learnable weights of \( W^Q \), \( W^K \), and \( W^V \in \mathbb{R}^{d \times d} \) and \( d \) is the feature dimension divided on \( n \) equal blocks. The attention function can then be applied on parallel on each set. This multi-attention architecture allows for the network to attend to the information from representation sub-spaces at multiple positions as:
MultiHead(Q, K, V) = Concat(head₁,...,headₙ)Wₒ

headᵢ = Masked-MultiHead(QWᵢ,Qᵢ,KWᵢ,Kᵢ,VWᵢ,Vᵢ)

Masked-MultiHeadᵢ = softmax \left( \frac{QᵢKᵢᵀ}{\sqrt{d/h}} + M \right) Vᵢ \tag{1.14}

where Wᵢ,Q ∈ \mathbb{R}^{d \times dk}, Wᵢ,K ∈ \mathbb{R}^{d \times dk}, Wᵢ,V ∈ \mathbb{R}^{d \times dv} are the projection matrices for each attention head and Wₒ ∈ \mathbb{R}^{nd_v \times d} is the final projection matrix for all the heads. The multi-head-attention is applied on both the encoder and the decoder. On the encoder, the keys, values and queries are gathered from the previous layer of the encoder itself and attends to all positions. On the other hand, the decoder prevents leftwards information flow by masking out values in the softmax function corresponding to illegal connections.

The Feed Forward Network (FFN) is a simple linear transformation including a ReLU activation function. It is mathematically expressed as:

\[
\text{FFN}(x) = \text{ReLU}(xW₁ + b₁)W₂ + b₂ \tag{1.15}
\]

Here, W₁ and W₂ denote the weight matrices, and b₁ and b₂ are the bias vectors. In practice, the linear transformation has a size defined as four times the transformer dimension and then resized to the original size, allowing the model to learn more complex representations.

Building upon the success of the Transformer in addressing Natural Language Processing (NLP) tasks, (Dosovitskiy, Beyer, et al., 2021) adopted the attention mechanism and applied it to Computer Vision (CV) problems in the form of the Vision Transformer (ViT). In a domain where Convolutional Neural Networks (CNNs) have traditionally held the state-of-the-art (SOTA) position, ViT introduced a unique self-attention mechanism capable of capturing information from lower levels, enhancing the model’s ability to grasp a broader global context. Images are transformed to 1D linear embeddings and divided by patches, where the model learns powerful relationship between them. However, ViTs are notorious for requiring bigger datasets for an effective training, a drawback that
may hinder its results with limited amount of data. This approach has already been implemented for a variety of tasks such as image recognition (Touvron et al., 2021; Joo et al., 2023), video captioning (Yamazaki, Truong, et al., 2022; Yamazaki, K. Vo, et al., 2023), action localization (K. Vo, Yamazaki, et al., 2022; K. Vo, Truong, et al., 2023), aerial imaging (Kasmi, Dubus, et al., 2022) object detection (Sun et al., 2021; Tran et al., 2022), image segmentation (Tran et al., 2022) and medical imaging (Nguyen et al., 2023) proving their capacity to synthesise global information.

1.3 Deep Learning for PV analysis

Existing Conventional Methods

Traditionally, mathematical techniques to forecast PV energy has been divided into three categories: physical, statistical and hybrid.

Physical models describe the conversion processes to electricity in PV models. These approaches usually rely on parameters drawn from the datasheet of the electrical circuit modelling the output of the PV installation. The conversion from sunlight to electricity inside a PV cell can be represented as a current generator using:

\[
I = I_{PV} - I_D = I_{PV} - I_0(e^{\frac{V}{nV_T}} - 1)
\]

Figure 1.4: Equivalent circuit representing the three parameter model.

where the three parameters characterizing this model are \(I_{PV}\), \(I_0\) and \(n\) representing the light-generated current, the reverse saturation current of the PN junction and the diode ideality factor.
respectively. Commonly recognized as three-parameter models (Dolara, Leva, and Manzolini, 2015), these formulations provide a concise representation of the photovoltaic system. Some authors opt for equivalent circuit models, expanding the complexity by incorporating up to five parameters (Celik and Acikgoz, 2007). This heightened complexity aims to capture additional nuances in the system, offering a more comprehensive depiction of the photovoltaic behavior.

Although accurate, implementations with the previous methodology tend to add historical data to produce better representations of the real performance of a solar cell. Weather data from service stations is the preferred method for such analysis, where the employed models use a database of parameters including the power output of the PV installation for comparison. Linking both physical and statistical implementations, these methodology falls under the hybrid category (Ogliari et al., 2017).

Statistical approaches focusing on individual sources have heavily dominated PV prediction literature. Focusing on the production of a single PV plant over a period a time, the solar power output is treated as an dependent variable from a set of predictors. An example of this process is a Machine Learning (ML) Support Vector Machine (SVM) with PV measurements and weather data introduced for accurate short-term PV power prediction by (Abuella and Chowdhury, 2017). Using the datasets from the Global Energy Forecasting Competition (GEFCom, 2014), the SVM produced impressive results at forecasting the PV output over a monthly basis, even compared with an ANN model. Forecast of this nature can be applied to an ensemble of solar plans in a region to keep accurate metrics for grid operators to balance demand and supply for an area. However, these studies are inefficient for small-scale operations, where the installation is "invisible" for the grid operator. With the rise of solar PV and its growing market, the effect of these installations on the energy market continues to rise, leading to the introduction of Computer Vision to properly characterize them.

**Existing Convolutional Methods**

In recent times, the mapping of high-fidelity solar installations has become increasingly important. This uptick in interest is a direct result of the rapid adoption of photovoltaic (PV) energy for small
installations. The decentralized nature of those PV modules coupled with the lack of measurements necessary to apply the analysis discussed on Section 1.3 led to researchers focusing on solutions applicable on a greater scale.

The preliminary efforts in this field focused on binary classification of images, determining the presence or absence of PV installations (Jordan M Malof et al., 2016; Jordan M. Malof, Collins, and Bradbury, 2017). However, since these classifications don’t include the coordinates of the PV systems, they don’t allow for further downstream applications. DeepSolar (Yu et al., 2018) bridged that gap by utilizing a Convolutional Neural Network (CNN) as a dual-stage network which includes classification and segmentation strategy coupled with precise socioeconomic analysis, marking a significant milestone in solar PV profiling. The success of this endeavor paved the way for refining the characterization network. This was achieved by integrating state-of-the-art (SOTA) CNN methods into classifier and segmentation architectures. Meanwhile, certain researchers opted to concentrate solely on segmentation such as (Zhuang, Z. Zhang, and L. Wang, 2020; Zech and Ranalli, 2020). Furthermore, these methodologies are adaptable for generating PV capacity estimates, with 3D-PV-Locator(Mayer et al., 2022), (Kasmi, Dubus, et al., 2022) being one of the first works to explore this approach. Such assessments leverage the dimensions and geographical data of the segmented installations to further analyse the impact of PV energy.

**Challenges of PV analysis**

Although current approaches for PV segmentation have proven remarkably successful at creating an accurate map of installations throughout regions such as USA (Yu et al., 2018) or France (Kasmi, Dubus, et al., 2022), the two-stage framework consisting of a classifier and segmentor networks employed in both works has two major limitations: (i) They employ a two-stage framework, consisting of separate classifier and segmentor networks. This approach heavily relies on the classifier network, leading to suboptimal learning in the segmentation network. Essentially, they adapt existing DL frameworks designed for natural images to solar PV imagery, neglecting the unique challenges posed by the latter. (ii) These existing DL frameworks are originally designed for natural images and do not adequately address the specific challenges encountered in solar PV
imagery. As depicted in Fig. 1.5, these challenges include intra-class heterogeneity, inter-class homogeneity, and the identification of small objects.

1.4 Thesis Structure

Although the current SOTA methods have achieved impressive results at denoting the size of solar modules using overhead imagery, there is still room to improve. In order to accurately analyze the impact of solar installations, we need to properly mask them. Leveraging the advances of the Transformer architecture, S3Former will improve on the segmentation framework of the current literature implementations and demonstrate more robust results.

The remainder of this thesis is aimed at discussing the methodology, data and results obtained throughout my research. The main contribution from my work is to introduce a new method, S3Former, to characterize PV modules. Our results improve those obtained by the traditional CNN approaches for this task and we validate our findings through extensive benchmarking on comparable scenarios.

The thesis chapters are divided as:

1. **Chapter 2**: Introduces the datasets analysed by S3Former. Additionally, we delve into
the description of S3Former, elucidating its novel design aimed at enhancing small object identification and addressing both inter-class and intra-class dependencies.

2. **Chapter 3**: Results and implementation are discussed in depth. The experiments conducted and scenarios tested are meticulously presented, offering insights into cases where S3Former outperforms the state-of-the-art (SOTA) models using equivalent setups. To provide a well-rounded assessment, quantitative values derived from commonly used metrics are incorporated to quantify and contextualize our results. Additionally, details such as augmentations employed to enhance feature extraction and the loss function implemented are elucidated in this section.

3. **Chapter 4**: Recap of the work presented on this thesis and further improvements are discussed.

The work described on this thesis can be found on the following research:


2.1 SolarFormer Framework

Figure 2.1 shows the difference between the SOTA approach to characterizing PV installations vs our proposed architecture. SolarFormer eliminates the necessity for a classification network in the processing pipeline, prioritizing the development of a robust segmentation network. The rationale behind this design choice can be summarized as follows:

1. Better segmentation learning. Since we are removing the classification network, the segmentation will see both images with and without solar modules during training. By introducing this change we are allowing the model to learn stronger representations of solar panels. Given that classification networks are inherently fallible, erroneously identified positive images supplied to the segmentation network are less likely to be inaccurately masked in SolarFormer compared to the SOTA implementation.

2. Reducing complexity. Less steps to compute the mask leads to simpler networks. This also involves eliminating training both components separately.

Figure 2.1: SolarFormer framework: a semantic segmentation network for solar installations. Top component represents our implementation, bottom is the SOTA approach.
Figure 2.2: Overall network architecture of our proposed SolarFormer which consists of three components i.e., Backbone, Pixel Decoder, and Mask-attention Transformer Decoder.

2.2 SolarFormer Architecture

This section provides a detailed description of our transformer-based framework S3Former. Both SolarFormer and S3Former share the same overall architecture barring the addition of the Self-Supervised module for feature extraction for the later. Both frameworks are specifically tailored for the segmentation of PV installations. As illustrated in Fig.2.2, our network comprises three distinct components: Network Backbone, Pixel Decoder, Mask-attention Transformer Decoder. Given an RGB image with dimensions $H \times W \times 3$, the network produces a corresponding segmentation mask of dimensions $H \times W$.

Network Backbone

The first block of our architecture is represented by the Backbone networks, which extract features from the input image of dimensions $H \times W \times 3$ to be further processed by the downstream architecture. Backbone networks are crucial in network design, and various implementations can be found in the literature. VGG (Simonyan and Zisserman, 2014), for instance, was developed as a deep CNN suitable for image classification, boasting a depth of up to 19 layers. However, its straightforward
design, primarily consisting of pooling and fully connected layers, paved the way for other CNN backbones to rise to prominence. CNN backbones include ShuffleNet (X. Zhang et al., 2018), Inception (Szegedy et al., 2015), DenseNet (Huang et al., 2017), EfficientNet (Tan and Le, 2019), etc. One of the most popular backbone networks are Residual Networks (ResNets) (K. He et al., 2016). This family of networks is based on blocks of convolution and pooling layers with skipped connections and recurrent units between them followed by batch normalization’s. The ResNet family has a variety of implementations: ResNet-50, ResNet-34, ResNet-101, etc. and has been widely used for image classification and semantic segmentation tasks. For our model, we will be using ResNet pretrained on ImageNet as the backbone for our implementation and testing the performance of two of its most popular varieties: ResNet-50 and ResNet-101.

By employing ResNet as our proposed model’s backbone, an input image $I \in \mathbb{R}^{H \times W \times 3}$ is transformed into several multi-scale feature maps $F$. In our model, we produce feature maps at multiple resolutions. Specifically, we generate four such feature maps denoted as $F_1 \in \mathbb{R}^{C_{F_1} \times \frac{H}{4} \times \frac{W}{4}}$, $F_2 \in \mathbb{R}^{C_{F_2} \times \frac{H}{8} \times \frac{W}{8}}$, $F_3 \in \mathbb{R}^{C_{F_3} \times \frac{H}{16} \times \frac{W}{16}}$ and $F_4 \in \mathbb{R}^{C_{F_4} \times \frac{H}{32} \times \frac{W}{32}}$, where $C_{F_i}$ represents the number of channels, and $H \times W$ is the size of the input image.

**Pixel Decoder**

The Pixel Decoder module serves as an intermediary between the Backbone Network and the Mask-Attention Transformer Decoder. It consists of two main components: the Multi-Scale Transformer Encoder and the Per-Pixel Embedding Module. The first component produces enhanced embedding features $D_1, D_2, D_3, D_4$ whereas the later component generates per-pixel embeddings for the image, denoted as $E_{\text{pixel}}$.

**Multi-Scale Transformer Encoder**

This module takes the last four feature maps $F_1, F_2, F_3, F_4$ generated by the backbone network and processes them hierarchically. These feature maps are ordered from low to high resolution and flattened using an embedding projection to achieve a consistent channel size $C_e$, obtaining $S_i \in \mathbb{R}^{H_i \times W_i \times C_e}$ where $i = 1, 2, 3, 4$ and $C_e$ is equal to the size of $C_{F_1}$. The flattened embeddings are
then merged into $S \in \mathbb{R}^{K \times C_e}$, where $K = \sum_{i \in \{4,3,2,1\}} H_i \cdot W_i$.

Since the embedding features $S$ are flattened out of their original spatial shapes and scale level, they are supplemented with a learnable encodings $L$ and $P$ of shape $S, P, L \in \mathbb{R}^{K \times C_e}$. $L$ is a positional encoding providing spatial information about the original location of each feature within the image. The second encoding, $P$, retains information about the different scales for the transformer encoder.

The three encodings are then passed through a Multi-Scale Transformer Encoder (Dosovitskiy, Beyer, Kolesnikov, et al., 2021) to produce learned features from the input sequence, taking the embedded features and generating encoded features that capture the relationships between the elements. We can denote this operation as:

$$E = \text{Multi-scaleTransformerEncoder}(S, P, L) \quad (2.1)$$

where the correlated feature embeddings $E_i \in \mathbb{R}^{H_i \cdot W_i \times C_e}$ are then divided into groups based on the multi-scale level and restored to their original spatial shape with a fixed channel size $C_e$ as $D_1 \in \mathbb{R}^{C_e \times \frac{H}{4} \times \frac{W}{4}}, D_2 \in \mathbb{R}^{C_e \times \frac{H}{8} \times \frac{W}{8}}, D_3 \in \mathbb{R}^{C_e \times \frac{H}{16} \times \frac{W}{16}}$ and $D_4 \in \mathbb{R}^{C_e \times \frac{H}{32} \times \frac{W}{32}}$.

**Per-Pixel Embedding Module**

The second stage of the Pixel Decoder computes the pixel embeddings $E_{\text{pixel}}$ using the first feature layer $D_1$ outputted from the Multi-Scale Encoder. We can calculate this process as:

$$E_{\text{pixel}} = \text{EmbeddingModule}(D_1) \quad (2.2)$$

The goal of this process is to scale the $D_1 \in \mathbb{R}^{C_e \times \frac{H}{4} \times \frac{W}{4}}$ to the original spatial shape $H \times W$ of the image and creating $E_{\text{pixel}} \in \mathbb{R}^{C_e \times H \times W}$. To do so, two 2x2 transposed convolutional layers with a stride of 2 are applied sequentially to upscale $D_1$. Each pixel in the $E_{\text{pixel}}$ represents both the semantic and mask classification of the corresponding pixel on the original image.
Figure 2.3: Mask-attention Transformer Decoder. This module represents the transformations described on Equation 2.4

**Mask-attention Transformer Decoder**

**Mask Predictor**

To predict the segmented masks of possible instances in every image, per-pixel embeddings $E_{pixel}$ are utilised. The prediction process involves learning $N$ per-segment query embeddings $Q \in \mathbb{R}^{N \times C_e}$, where $N$ represent the features of the maximum amount of possible instances in the image. Each $Q$ then correlates with every pixel feature in $E_{pixel}$ and determines if the pixel belongs to the corresponding instance. The predicted instance segmentation can be derived as:

$$M = \text{MaskPredictor}(Q, E_{pixel})$$  \hspace{1cm} (2.3)$$

where $M \in \mathbb{R}^{N \times H \times W}$ is calculated simply performing a dot product followed by a sigmoid activation.
Mask-attention Transformer Decoder

This module decodes $N$ per-segment query embeddings $Q \in \mathbb{R}^{N \times C_q}$ from the encoded feature maps $D_1, D_2, D_3, D_4$. Each of the per-segment embeddings represents instances of the image by applying attention to the image features. The decoding procedure is done recurrently on different steps, where each step is associated to a layer from the encoded feature maps. At the first step $l = 0$ we are processing the encoded feature layer with the lowest resolution $D_4$, and we recurrently process each layer all the way to the highest resolution $D_1$. At each of these layers, the query $Q_{l+1}$ is decoded from the previous layer’s $Q_l$ and its corresponding encoded feature maps.

A predicted mask $M_l$ is subsequently computed using the current $Q_l$ query and the per-pixel embeddings $E_{\text{pixel}}$ and interpolated to the same size as $D_{4-l}$. $M$ is then used as an attention mechanism that focuses on the salient parts of the features maps and allows for the query embeddings to attend to the features that are most important to the instances being decoded. Specifically, the attention mask is applied to $D_{4-l}$ during the decoding process to selectively attend to certain areas of the feature maps most relevant to the instance decoded. Each step can be formulated as:

$$M_l = f_p(Q_l, E_{\text{pixel}})$$

$$Q_{l+1} = f_t(Q_l, D_{4-l}, M_l)$$  \hspace{1cm} (2.4)

Segmentation Mask Prediction

The last step of the model generates the final segmentation masks $M_{\text{final}} \in \mathbb{R}^{N \times H \times W}$. To do so, the query embeddings from the decoder at the last step $l = 3$, $Q_{l=3}$ and the per-pixel embeddings $E_{\text{pixel}}$ are utilized to compute $M_{\text{final}}$.

2.3 Transformer-Decoder Loss

Typically, training large models requires large quantities of memory, especially when datasets contain high-resolution imagery. PointRend (Kirillov et al., 2020) circumvents this problem by proposing a reduced training for segmentation models selecting $K$ points from the mask and
computing loss calculations on the selected subset. This methodology can lead to a great reduction of model complexity while retaining great performance. The new training strategy is applied for both losses calculated: a matching loss for the cost matrix for bipartite matching and the final loss between the predicted and ground truth masks. The final loss is the combination of both losses:

$$L_{\text{final}} = L_{\text{mask}} + \lambda_{\text{cls}} L_{\text{cls}}$$  \hspace{1cm} (2.5)

where $\lambda_{\text{cls}} = 2.0$. We further expand on this equation on the sub-sections below.

**Classification Loss**

Also referred as the matching loss, it refers to the region-based loss first described on PointRend (Carion et al., 2020). To reach this value, we need to partition an image into $N$ regions represented with binary masks $m_i \in [0, 1]^{H \times W}$, where $i = 1, 2, \ldots, N$ and associating each region with a distribution over $P$ categories. To perform mask classification, the desired output is a set of $N$ probability-mask pairs, i.e., $z = (c_i, m_i)$. It must be noted that $N$ and $P$ may take different values, leading to the inclusion of a "no value" $\emptyset$ label in the $P$ set.

To train a model using this loss, we need to define a $\sigma$ rule between the set of predictions $\hat{z}$ and ground-truth $z^{gt} = \{c^{gt}_i, m^{gt}_i\}$ where $m^{gt}_i \in [0, 1]^{H \times W}$, $c^{gt}_i \in \{1, 2, \ldots, K\}$ represent both the classes and masks at the $i^{th}$ ground-truth segment and $|z^{gt}| = N^{gt}$ is the size of the set. If the size of $N^{gt}$ differs from $N$, we pad the ground truth set with a "no value" $\emptyset$ label to permit one-to-one matching. For semantic segmentation, we can use bipartite matching to generate our cost matrices:

$$\hat{\sigma} = \arg\min \sum_{i=1}^{N} L_{\text{match}}(z^{gt}_i, \hat{z}_{\sigma(i)})$$  \hspace{1cm} (2.6)

where $L_{\text{match}}(z^{gt}_i, \hat{z}_{\sigma(i)})$ is a pair-wise matching cost between the ground truth $z^{gt}_i$ and its corresponding prediction $\hat{z}$ at index $\sigma(i)$. We can optimize this operation by computing the Hungarian algorithm as (Stewart, Andriluka, and Ng, 2016). We can express the final loss by using the
ground-truth $z_i$ and the set of predictions $\hat{z}_i$ to compute the value of $\mathcal{L}_{final}$ as a classification loss composed of the cross-entropy classification loss and a binary mask loss as:

$$
\mathcal{L}_{final} = \sum_{j=1}^{N} \left[ -\lambda_{cls} \log(p_{\sigma(j)}) c_{j}^{gt} + \mathbb{1}_{c_{j}^{gt} \neq \emptyset} \mathcal{L}_{mask}(m_{\sigma(j)}, m_{j}^{gt}) \right]
$$

(2.7)

where we select the optimal assignments computed from $\sigma$ to compute the pairs of predictions and ground-truth values. We can further express this loss in terms of the variables defined on Section 2.2, where $N$ is the number of queries and $P \in \{0, 1\}$. To optimize memory usage, we uniformly sample $K$ points and compute the matching process on the subset matrix. We define $\mathcal{L}_{mask}$ on the section below.

**Mask Loss**

This term is calculated between the predicted and ground truths masks and corresponds to the per-pixel classification loss on the matched masks. $K$ points are sampled from different pairs of prediction using *importance sampling* introduced on PointRend (Kirillov et al., 2020). Firstly, $nK$ candidate points ($n > 1$) are randomly generated using an uniform distribution. From the pool of possible points $\beta K$ where $\beta \in [0, 1]$ are selected based on uncertain coarse predictions interpolated from predicted values in all $nK$. The top most uncertain $\beta K$ points are then selected, while the remaining $(1 - \beta)$ points are selected from the uniform distribution.

Once the matching points have been selected, a dice loss and a binary cross-entropy loss $\text{BCE}$ is applied. We express the binary cross-entropy for two classes as:

$$
\text{BCE} = -\frac{1}{K} \sum_{i=1}^{K} y_i \log(\sigma(\hat{y}_i)) + (1 - y_i) \log(1 - \sigma(\hat{y}_i))
$$

(2.8)

where $\hat{y}$ and $y$ are the predicted and Ground-Truth point matrix’s respectively and $\sigma$ is the sigmoid function. This loss is widely implemented for classification and segmentation tasks as it penalises

1The sigmoid function is modelled after $f(x) = \frac{1}{1+e^{-x}}$ and has a characteristic S shape.
wrong label assignments. To handle the class imbalances present for semantic segmentation tasks, the dice loss is calculated as:

\[
DICE = 1 - \frac{1 + 2 \sum_{i=1}^{K} y_i \hat{y}_i}{1 + \sum_{i=1}^{K} (y_i + \hat{y}_i)}
\]  \tag{2.9}

The final loss for the mask prediction is then leveraged from both terms by applying weights to both values:

\[
\mathcal{L}_{mask} = \lambda_{ce} BCE + \lambda_{dice} DICE
\]  \tag{2.10}

For our implementation, we have set $\lambda_{ce} = 5.0$ and $\lambda_{dice} = 5.0$.

### 2.4 S3Former Framework

Building on the architecture of SolarFormer, S3Former improves the methodology by introducing Self-Supervised Learning (SSL). Figure 2.4 presents the new architecture, with SSL added as a new component on the characterization pipeline. The rationale for integrating SSL stems from the challenges faced by SolarFormer in training a generalized model with the available data. As expounded in Section 3.1, the existing data proves insufficient for a Transformer-based model, given their high parameter count—a well-recognized limitation in such implementations.

To address this constraint, we propose pre-training the backbone network through a pre-task. The primary objective of this pre-training phase is to derive optimized weights for the network, thereby enhancing performance in subsequent tasks. This optimization is achieved by leveraging SSL, a technique elaborated upon in the forthcoming sections.

**Self-Supervised Learning**

Self-supervised learning (SSL) is a technique aimed at learning feature representations without a labelled dataset, contrary to supervised pre-training which requires manual data. This area has been growing rapidly recently (Y. Wang et al., 2022), and has started to get implemented for Computer
Figure 2.4: Hierarchical comparison of the S3Former framework, with the top component depicting our enhanced implementation, the middle segment representing SolarFormer, and the bottom illustrating the SOTA approach.

Vision tasks, where the network is trained in an unsupervised manner by learning a pretext task during pre-training and then transferring the learnt knowledge to a target model to fine-tune it on a down-stream task (Noroozi et al., 2018). Depending on the context of the pretext tasks, we can subdivide them into spatial (Doersch, Gupta, and Efros, 2015), spectral (Larsson, Maire, and Shakhnarovich, 2016), temporal (Misra, Zitnick, and Hebert, 2016) or other types of context (Jenni and Favaro, 2018). For our implementation, the backbone is trained on a pre-text task aimed at enhancing the spatial representations of our data. The improved features extracted are then used for the downstream PV installation segmentation task.

Self-Supervised Learning module

Using ResNet-50 and ResNet-101 as the baselines for our network backbones, we will also analyse the effect of unsupervised pre-training on the feature maps extracted. Our implementation for this task is based on DINO (Caron et al., 2021a), which leverages the use of self-supervised learning (SSL) with knowledge distillation (Hinton, Vinyals, and Dean, 2015) to train a teacher and student
Figure 2.5: Self-distillation with no labels using Remote Sensing images. For simplicity, only one view per network is included.

networks. Figure 2.5 illustrates this process. The teacher and student networks use different sets of parameters, named $g_{\theta_t}$ and $g_{\theta_s}$ respectively, using the same base model. An image is then randomly transformed $n_t$ and $n_s$ times, producing different views that are passed onto each network. This transformations may include color jiterring, vertical or horizontal flips or any other augmentations changing the original image. The output of each network is normalized with a temperature softmax over the feature dimension. A cross-entropy loss measures the similarity of the features generated and a stop-gradient (sg) operator progagates the gradients only through the student. A exponential moving average (ema) updates $g_{\theta_t}$ using $g_{\theta_s}$. Once training is finalized, the weights from the teacher network are transferred to S3Former as the backbone. In contrast, SolarFormer implements the base ResNet model pre-trained on ImageNet.
Chapter 3

EXPERIMENT RESULTS

3.1 Remote Sensing Datasets

For decades, aerial photography served as the primary source of commercial high-resolution imagery, including orthoimagery devoid of spatial distortion. Since 1999 with the launch of the Ikonos satellite, emerging satellites capable of high-resolution imagery have been deployed to produce panchromatic imagery with average revisiting times of 3 days. Remote Sensing (RS) images offer an overhead perspective of Earth’s surface, defining the level of detail as Ground Sample Distance (GSD). With both the high spatial and high temporal resolution images available, project monitoring and assessing our environment and resources have multiplied throughout the years. Object detection and classification are the cornerstone of such assessments. Such analysis include climate change analysis (O’neill et al., 2013), natural disaster monitoring (Said et al., 2019), urban planning (B et al., 2020; Effat and Hegazy, 2013), and land cover analysis (J. Wang et al., 2021).

Machine Learning techniques, concretely those related to scene categorization and object detection, provide a solution for generating insights automatically from aerial imagery. Supervised object detection is a common approach for understanding satellite imagery. The development of these methods requires of labelled data including the different instances present of each images, i.e roads, cars, etc. However, a significant challenge lies in crafting suitable datasets. The complications arise from (i) the extensive need for annotated images, a process that’s both time-consuming and meticulous, (ii) the potential sensitivity of the data, making it unsuitable for public distribution, and (iii) the size of solar panels, with most installations ranging from 1 to 2 meters in size, necessitating resolutions of at least 50 cm. Fortunately, organizations like IGN and USGS have been proactive, offering freely accessible aerial images. Despite this support, researchers often find themselves in a bind — they either annotate a limited selection of aerial imagery such as Hyperion-Solar-Net (Parhar et al., 2022), keep the collected data unreleased or organize crowd sourced annotating
Table 3.1: Datasets characteristics comparison. "+ Samples" indicate images containing solar panels, while "- Samples" lack solar arrays.

<table>
<thead>
<tr>
<th>Fold</th>
<th>Solar PV Datasets division</th>
<th>GGE, France</th>
<th>USGS, California</th>
<th>IGN, France</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>- Samples</td>
<td>+ Samples</td>
<td>- Samples</td>
</tr>
<tr>
<td>Train</td>
<td></td>
<td>9312</td>
<td>7968</td>
<td>11100</td>
</tr>
<tr>
<td>Test</td>
<td></td>
<td>3104</td>
<td>2656</td>
<td>3700</td>
</tr>
<tr>
<td>Validation</td>
<td></td>
<td>3104</td>
<td>2656</td>
<td>3700</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>28,807</td>
<td></td>
<td>37,660</td>
</tr>
</tbody>
</table>

campaigns such as IGN-France (Kasmi, Saint-Drenan, et al., 2023), USGS, California (Bradbury et al., 2016).

**California, USGS**

This aerial imagery, sourced from the United States Geological Survey (USGS), is part of their continued initiative to map various U.S. regions using detailed imagery, focusing specifically on California. The majority of the solar arrays captured in this dataset were installed post-2012, and the images were acquired after 2013. Four cities were selected: Fresno, Modesto, Stockton and Oxnard. Out of all the cities, Modesto has the smallest amount of data due to hosting the smallest density of solar arrays. In total, this dataset boasts 601 TIF images, each with dimensions of 5000x5000 pixels and a resolution finer than 30 cm/pixel. Each image represents a 2.25 square kilometer area. For consistency across datasets, we extracted 400x400 patches from these images, resulting in a total of 37,660 images, of which 50.87% feature PV installations. Annotations are provided by (Bradbury et al., 2016).

**France, IGN**

This large high-resolution aerial imagery dataset was gathered from the publicly available Institut géographique national (IGN) website in France. Tiles were downloaded with an original size of 25000x25000 covering an area of 25 square kilometers. Similar to the process followed with the California dataset, these images were annotated through a crowd-sourcing campaign (Kasmi, Saint-Drenan, et al., 2023). The dataset contains 17,325 thumbnails sized 400-by-400 at a resolution of 20 cm/pixel with a share of 44.34 % containing solar installations.
France, Google Earth Engine

While this dataset’s aerial imagery overlaps with the regions covered in the IGN, France dataset, the images were specifically derived from Google Earth Engine. Boasting a GSD of 10 cm/pixel, this resolution is the most refined of all datasets used in our experiments. Images were extracted automatically at a zoom level of 20 using the ground-truth from metadata gathered in (Kasmi, Saint-Drenan, et al., 2023). The same crowd-sourcing initiative was employed for annotations. This dataset includes 28,807 thumbnail images, each measuring 400x400 pixels, with a ratio of 46.11% depicting solar installations versus those that don’t.

3.2 Experiment Set-Up

We can subdivide each segment by the different components trained for S3Former:

**S3Former:** Our implementation for the segmentation task is based on Mask2former (Cheng et al., 2022). All the code has been developed using Python. We trained S3Former and SolarFormer using two RTX 8000 GPUs on an Ubuntu-based machine. We used batches of 16 images per iteration and totalled 40 epochs per dataset. We employed an AdamW (Loshchilov and Hutter, 2017) optimizer with a learning rate of $1 \times 10^{-4}$ and a weight decay of 0.05 for all backbones. A learning rate multiplier of 0.1 is applied for the backbone and the learning rate decays at 0.9 and 0.95 fractions of the total number of training steps by a factor of 10.

**SSL:** The SSL pre-task has been implemented using DINO (Caron et al., 2021b). Each backbone is trained using the same resources and local machine as S3Former. We use batches of 32 images for 100 epochs per dataset. We decided to use an SGD (Ruder, 2016) optimizer with a learning rate of 0.01 with a cosine scheduler down to $1 \times 10^{-6}$ and a weight decay of $1 \times 10^{-4}$.

3.3 Augmentations

Similarly to the previous section, we can apply different sets of augmentations depending on the task:

**S3Former:** During training, all images are cropped from their original size of 400×400 to a slightly smaller random cutout of 380 × 380. Cropped images are then randomly flipped and saturation,
brightness, contrast and hue transformations are further applied to avoid favoring certain features, like rooftop colors, to affect the performance of our model. Once the augmentations have been applied, images are then normalized using ImageNet values (i.e., a mean of \((0.485, 0.456, 0.406)\) and a standard deviation of \((0.229, 0.224, 0.225)\)). For testing, we are only normalizing input images and avoiding other transformations.

**SSL:** Our training methodology deviates from the implementation proposed by the authors of DINO. Rather than focusing on subsets of views generated through crops, we adopt a different approach. Specifically, we apply two distinct sets of augmentations, rotating the orientation, swapping the channels, or introducing random Gaussian noise to the original image. Subsequently, both modified views are fed into the student and teacher networks. We apply the same ImageNet normalization values as S3Former. We can loosely verify the performance of our features by comparing the KNN values obtained with those reported on the original work, but the most accurate validation consists on training and testing the model with the same methodology as described on *S3Former*.

All the transformations applied are based on the torchvision package implementation.

### 3.4 Metrics

To assess the efficacy of S3Former, we will focus on the analysis of the labels given to each Predicted mask \(P \in \mathbb{R}^{H \times W}\) and compute the difference to its corresponding Ground Truth \(GT \in \mathbb{R}^{H \times W}\). Each pixel has a value \(l \in [0, 1]\) assigned on both masks. These calculations hinge on the comprehension of four key values: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). In the context of PV segmentation, we can formulate these values as:
where all values are computed with respect to the "PV module" class ($l = 1$). We can further combine the previous values to calculate three standard metrics employed in semantic segmentation evaluations: Intersection over Union (IoU), F1 score, and Accuracy. We can express them as:

$$\text{IoU} = \frac{TP}{TP + FN + FP}$$  \hspace{1cm} (3.2)$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP}$$  \hspace{1cm} (3.3)$$

$$\text{F1-score} = \frac{2TP}{2TP + FN + FP}$$  \hspace{1cm} (3.4)$$

Normally, semantic segmentation results also include the mean of Equations 3.2, 3.3 and 3.4 calculated over all existing labels $l$. However, the "background" class carries no statistical significance for the analysis of PV energy and, in general, all models achieve near perfect scores in this metric. Utilizing such values for our analysis could lead to an overestimation of the performance of our model, so authors in this field tend to avoid reporting them.

### 3.5 Quantitative Analysis

We evaluate our S3Former in terms of both network performance and network design efficiency. The **bold** and *ITALIC UNDERLINE* represent the best and second best performances in all the tables presented in this work. **i. Network Performance:** Our S3Former and SolarFormer are
Table 3.2: Performance comparison between our S3Former and SolarFormer with existing DL-based Solar PV profiling methods on GEE, France dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>IoU</th>
<th>F1-score</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zech et al., (Zech and Ranalli, 2020)</td>
<td>68.59</td>
<td>81.40</td>
<td>77.79</td>
</tr>
<tr>
<td>3D-PV-Locator (Mayer et al., 2022)</td>
<td>80.70</td>
<td>89.30</td>
<td>90.70</td>
</tr>
<tr>
<td>Zhuang, et al., (Zhuang, Z. Zhang, and L. Wang, 2020)</td>
<td>76.60</td>
<td>86.69</td>
<td>84.20</td>
</tr>
<tr>
<td>Hyperion-Solar-Net (Parhar et al., 2022)</td>
<td><strong>81.49</strong></td>
<td>88.46</td>
<td>89.80</td>
</tr>
<tr>
<td><strong>Our SolarFormer</strong></td>
<td>79.42</td>
<td>88.53</td>
<td><strong>94.18</strong></td>
</tr>
<tr>
<td><strong>Our S3Former</strong></td>
<td>79.68</td>
<td>88.69</td>
<td><strong>92.91</strong></td>
</tr>
</tbody>
</table>

Table 3.3: Performance comparison between our S3Former and SolarFormer with existing DL-based Solar PV profiling methods on IGN, France dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>IoU</th>
<th>F1-score</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zech et al., (Zech and Ranalli, 2020)</td>
<td>38.60</td>
<td>55.69</td>
<td>45.19</td>
</tr>
<tr>
<td>3D-PV-Locator (Mayer et al., 2022)</td>
<td>53.10</td>
<td>69.40</td>
<td>66.40</td>
</tr>
<tr>
<td>Hyperion-Solar-Net (Parhar et al., 2022)</td>
<td><strong>58.80</strong></td>
<td>74.06</td>
<td><strong>81.67</strong></td>
</tr>
<tr>
<td><strong>Our SolarFormer</strong></td>
<td>56.41</td>
<td>71.13</td>
<td>79.66</td>
</tr>
<tr>
<td><strong>Our S3Former</strong></td>
<td><strong>59.22</strong></td>
<td><strong>74.39</strong></td>
<td><strong>82.96</strong></td>
</tr>
</tbody>
</table>

Table 3.4: Performance comparison between our S3Former and SolarFormer with existing DL-based Solar PV profiling methods on USGS, California dataset.

<table>
<thead>
<tr>
<th>Methods</th>
<th>IoU</th>
<th>F1-score</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zech et al., (Zech and Ranalli, 2020)</td>
<td>69.80</td>
<td>73.29</td>
<td>82.20</td>
</tr>
<tr>
<td>3D-PV-Locator (Mayer et al., 2022)</td>
<td>80.60</td>
<td>89.30</td>
<td>87.40</td>
</tr>
<tr>
<td>Zhuang, et al., (Zhuang, Z. Zhang, and L. Wang, 2020)</td>
<td>84.39</td>
<td>91.60</td>
<td>90.89</td>
</tr>
<tr>
<td>Hyperion-Solar-Net (Parhar et al., 2022)</td>
<td>83.04</td>
<td>90.73</td>
<td>90.74</td>
</tr>
<tr>
<td><strong>Our SolarFormer</strong></td>
<td><strong>88.78</strong></td>
<td><strong>94.05</strong></td>
<td><strong>94.39</strong></td>
</tr>
<tr>
<td><strong>Our S3Former</strong></td>
<td><strong>89.05</strong></td>
<td><strong>94.21</strong></td>
<td><strong>94.33</strong></td>
</tr>
</tbody>
</table>

compared with current state-of-the-art (SOTA) DL-based Solar PV profiling methods, as detailed in Tables 3.2, 3.3 and 3.4.  

ii. Network Design Efficiency: We compare our S3Former and SolarFormer against other prevalent DL-based segmentation models employing various backbone networks (i.e. ResNet-50 and ResNet-101). This comparison is presented in Tables 3.5, 3.7 and 3.6. These quantitative assessments were conducted across all three datasets.

i. Network Performance
Tables 3.2, 3.3 and 3.4 summarize the performance of our S3Former against current SOTA methods and SolarFormer for solar profiling. For all experiments, the networks implemented by the authors of each work have been fine-tuned using ImageNet weights on each dataset tested. Adhering to the methodology initially presented by each author ensures a fair and consistent basis for comparison across experiments. It is essential to acknowledge that the reported results in this paper may differ from those presented in the original papers. This discrepancy arises from the use of the entire test fold for our final results, as opposed to a subset of images, specifically the ones referred to as "+ Samples" on Table 3.1. This modification in the evaluation setup ensures a more comprehensive and robust assessment, encompassing the entirety of the test dataset rather than a selected subset, thereby providing a more holistic perspective on the model’s performance. The difference between SolarFormer and S3Former are further discussed on the next section, so the result discussion will only reflect the values of our methods vs those achieved in SOTA. Our segmentation model outperforms Zech et al., (Zech and Ranalli, 2020), 3D-PV-Locator (Mayer et al., 2022), Zhuang, et al., (Zhuang, Z. Zhang, and L. Wang, 2020) and Hyperion-Solar-Net (Parhar et al., 2022) on all three datasets. For instance, when analyzing the USGS California dataset in Table 3.4, our model exhibits substantial improvements, including an impressive 4.66% increase in IoU, a notable 2.61% enhancement in F1-score, and a substantial boost of 3.56% in Accuracy compared to the second-best CNN method Zhuang, et al., (Zhuang, Z. Zhang, and L. Wang, 2020). Similarly, on IGN France in Table 3.3, our model showcases remarkable advancements, with gains of 0.42%, 0.33%, and 1.29% in IoU, F1-score, and Accuracy, respectively, when compared to the runner-up method, Hyperion-Solar-Net (Parhar et al., 2022).

ii. Network Design Efficiency Tables 3.5, 3.6 and 3.7 present performance comparisons between our S3Former, SolarFormer and the SOTA methods. Overall, our S3Former achieves its highest performance using ResNet-50, while the second-best performance is delivered by our SolarFormer and S3Former implemented with ResNet-101.

GGE, France. Our S3Former and SolarFormer demonstrate significant improvements over CNN-based models. Specifically, when benchmarked on the ResNet-101 backbone, SolarFormer gains
Table 3.5: Performance comparison between our S3Former and SolarFormer with existing DL-segmentation Networks on various backbones on GEE, France dataset.

<table>
<thead>
<tr>
<th>Backbones</th>
<th>Networks</th>
<th>IoU</th>
<th>F1-score</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet-50</td>
<td>FCN (Long, Shelhamer, and Darrell, 2015)</td>
<td>74.10</td>
<td>85.12</td>
<td>86.59</td>
</tr>
<tr>
<td></td>
<td>UNet (Ronneberger, Fischer, and Brox, 2015)</td>
<td>76.60</td>
<td>86.69</td>
<td>84.20</td>
</tr>
<tr>
<td></td>
<td>PSPNet (Zhao et al., 2017)</td>
<td>77.79</td>
<td>87.50</td>
<td>85.50</td>
</tr>
<tr>
<td></td>
<td>UperNet (Xiao et al., 2018)</td>
<td>79.40</td>
<td>88.49</td>
<td>89.80</td>
</tr>
<tr>
<td></td>
<td>Mask2Former (Cheng et al., 2022)</td>
<td>74.14</td>
<td>85.15</td>
<td>90.73</td>
</tr>
<tr>
<td></td>
<td><strong>Our SolarFormer</strong></td>
<td>77.65</td>
<td>87.42</td>
<td>93.08</td>
</tr>
<tr>
<td></td>
<td><strong>Our S3Former</strong></td>
<td><strong>79.56</strong></td>
<td><strong>88.61</strong></td>
<td><strong>93.20</strong></td>
</tr>
<tr>
<td>ResNet-101</td>
<td>FCN (Long, Shelhamer, and Darrell, 2015)</td>
<td>73.20</td>
<td>84.53</td>
<td>87.14</td>
</tr>
<tr>
<td></td>
<td>UNet (Ronneberger, Fischer, and Brox, 2015)</td>
<td>68.59</td>
<td>81.40</td>
<td>77.79</td>
</tr>
<tr>
<td></td>
<td>PSPNet (Zhao et al., 2017)</td>
<td>78.29</td>
<td>86.29</td>
<td>87.80</td>
</tr>
<tr>
<td></td>
<td>UperNet (Xiao et al., 2018)</td>
<td>79.19</td>
<td>88.40</td>
<td>90.10</td>
</tr>
<tr>
<td></td>
<td>Mask2Former (Cheng et al., 2022)</td>
<td>77.03</td>
<td>87.02</td>
<td>92.39</td>
</tr>
<tr>
<td></td>
<td><strong>Our SolarFormer</strong></td>
<td>79.42</td>
<td>88.53</td>
<td><strong>94.18</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Our S3Former</strong></td>
<td><strong>79.68</strong></td>
<td><strong>88.69</strong></td>
<td><strong>92.91</strong></td>
</tr>
</tbody>
</table>

0.23% IoU and 0.13% F1-score on UperNet (Xiao et al., 2018) and improves the Accuracy result of Mask2former (Cheng et al., 2022) by 1.79%. On the ResNet-50 backbone, S3Former outperforms the second best method, UperNet (Cheng et al., 2022), by 0.16% on IoU and 0.12% on F1-score and improves Mask2former (Cheng et al., 2022) by 2.47% on Accuracy. S3Former also outperforms SolarFormer in all measured metrics. Although the high precision of the 10 cm/pixel GSD imagery likely aids S3Former and SolarFormer in discerning finer image details, the granularity of the images prevents greater performance gains as shown on the other datasets.

**USGS, California.** Again, S3Former and SolarFormer displays notable gains over both CNN-based models and the Mask2Former (Cheng et al., 2022) on both ResNet backbones. On the ResNet-101 backbone, when compared to the top performing CNN-based model UperNet (Xiao et al., 2018), SolarFormer gains 4.99% on IoU, 4.75% on F1-score and 3.19% on Accuracy. The difference with Mask2former (Cheng et al., 2022) is still remarkable at 1.8%, 1.02% and 0.29%. SolarFormer and S3Former exhibit very similar scores, with all scores boasting a small difference of less than 0.05%. For ResNet-50, SolarFormer gains 2.89% on IoU and 1.67% on F1-score while S3Former improves 3.10% on Accuracy with respect to UperNet (Xiao et al., 2018). Despite a high GSD, California’s
Table 3.6: Performance comparison between our S3Former and SolarFormer with existing DL-segmentation Networks on various backbones on USGS, California dataset.

<table>
<thead>
<tr>
<th>Backbones</th>
<th>Networks</th>
<th>IoU</th>
<th>F1-score</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet-50</td>
<td>FCN (Long, Shelhamer, and Darrell, 2015)</td>
<td>63.02</td>
<td>77.32</td>
<td>75.55</td>
</tr>
<tr>
<td></td>
<td>UNet (Ronneberger, Fischer, and Brox, 2015)</td>
<td>84.39</td>
<td>91.60</td>
<td>90.89</td>
</tr>
<tr>
<td></td>
<td>PSPNet (Zhao et al., 2017)</td>
<td>77.30</td>
<td>87.19</td>
<td>86.10</td>
</tr>
<tr>
<td></td>
<td>UperNet (Xiao et al., 2018)</td>
<td>84.50</td>
<td>91.60</td>
<td>90.49</td>
</tr>
<tr>
<td></td>
<td>Mask2Former (Cheng et al., 2022)</td>
<td>85.33</td>
<td>92.08</td>
<td>92.80</td>
</tr>
<tr>
<td></td>
<td><strong>Our SolarFormer</strong></td>
<td>87.39</td>
<td>93.27</td>
<td>92.76</td>
</tr>
<tr>
<td></td>
<td><strong>Our S3Former</strong></td>
<td><strong>89.05</strong></td>
<td><strong>94.21</strong></td>
<td><strong>94.33</strong></td>
</tr>
<tr>
<td>ResNet-101</td>
<td>FCN (Long, Shelhamer, and Darrell, 2015)</td>
<td>61.83</td>
<td>76.41</td>
<td>73.55</td>
</tr>
<tr>
<td></td>
<td>UNet (Ronneberger, Fischer, and Brox, 2015)</td>
<td>69.80</td>
<td>73.29</td>
<td>82.20</td>
</tr>
<tr>
<td></td>
<td>PSPNet (Zhao et al., 2017)</td>
<td>76.70</td>
<td>86.79</td>
<td>85.50</td>
</tr>
<tr>
<td></td>
<td>UperNet (Xiao et al., 2018)</td>
<td>83.79</td>
<td>89.30</td>
<td>91.20</td>
</tr>
<tr>
<td></td>
<td>Mask2Former (Cheng et al., 2022)</td>
<td>86.98</td>
<td>93.03</td>
<td>94.10</td>
</tr>
<tr>
<td></td>
<td><strong>Our SolarFormer</strong></td>
<td><strong>88.78</strong></td>
<td><strong>94.05</strong></td>
<td><strong>94.39</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Our S3Former</strong></td>
<td><strong>88.74</strong></td>
<td><strong>94.03</strong></td>
<td><strong>94.36</strong></td>
</tr>
</tbody>
</table>

Vast aerial imagery and reduced granularity seem to enhance the model’s feature recognition.

**IGN, France.** Despite its lower resolution, S3Former and SolarFormer exhibits robust scores against other models on the ResNet backbones. The dataset’s relatively modest performance can be attributed to its lower resolution and limited availability of imagery. Similar to the GGE France dataset, the granularity in the imagery could impact the model’s efficiency in detecting installations. Notably, when benchmarked on the ResNet-101 backbone, S3Former showcases improvements of 4.21% on IoU, 2.54% on UperNet (Xiao et al., 2018), and 13.14% on Accuracy in comparison to Mask2Former (Cheng et al., 2022). On the ResNet-50 backbone, S3Former improves the results of Mask2Former (Cheng et al., 2022) by 5.13% and 4.18% on IoU and F1-score, respectively, while also surpassing the results obtained by SolarFormer.

### 3.6 Qualitative Analysis

**Comparison: CNN-based methods**

In this section, we provide a qualitative comparison of our Transformer-based methods against widely-used baseline models: DeepLabV3+ (L.-C. Chen et al., 2017) and UperNet (Xiao et al., 2018). We’ll highlight the advancements of our model in addressing challenges previously outlined.
Table 3.7: Performance comparison between our S3Former and SolarFormer with existing DL-segmentation Networks on various backbones on IGN, France dataset.

<table>
<thead>
<tr>
<th>Backbones</th>
<th>Networks</th>
<th>IoU</th>
<th>F1-score</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet-50</td>
<td>FCN (Long, Shelhamer, and Darrell, 2015)</td>
<td>45.23</td>
<td>62.29</td>
<td>59.97</td>
</tr>
<tr>
<td></td>
<td>UNet (Ronneberger, Fischer, and Brox, 2015)</td>
<td>48.50</td>
<td>65.30</td>
<td>59.60</td>
</tr>
<tr>
<td></td>
<td>PSPNet (Zhao et al., 2017)</td>
<td>50.90</td>
<td>67.50</td>
<td>62.90</td>
</tr>
<tr>
<td></td>
<td>UperNet (Xiao et al., 2018)</td>
<td>52.89</td>
<td>69.19</td>
<td>65.49</td>
</tr>
<tr>
<td></td>
<td>Mask2Former (Cheng et al., 2022)</td>
<td>54.09</td>
<td>70.21</td>
<td>87.63</td>
</tr>
<tr>
<td></td>
<td><strong>Our SolarFormer</strong></td>
<td>56.06</td>
<td>71.84</td>
<td>82.25</td>
</tr>
<tr>
<td></td>
<td><strong>Our S3Former</strong></td>
<td>59.22</td>
<td>74.39</td>
<td>82.96</td>
</tr>
<tr>
<td>ResNet-101</td>
<td>FCN (Long, Shelhamer, and Darrell, 2015)</td>
<td>45.52</td>
<td>62.55</td>
<td>62.43</td>
</tr>
<tr>
<td></td>
<td>UNet (Ronneberger, Fischer, and Brox, 2015)</td>
<td>38.60</td>
<td>55.69</td>
<td>45.19</td>
</tr>
<tr>
<td></td>
<td>PSPNet (Zhao et al., 2017)</td>
<td>48.80</td>
<td>65.60</td>
<td>59.10</td>
</tr>
<tr>
<td></td>
<td>UperNet (Xiao et al., 2018)</td>
<td>52.20</td>
<td>68.59</td>
<td>65.10</td>
</tr>
<tr>
<td></td>
<td>Mask2Former (Cheng et al., 2022)</td>
<td>49.34</td>
<td>66.08</td>
<td>66.52</td>
</tr>
<tr>
<td></td>
<td><strong>Our SolarFormer</strong></td>
<td>56.41</td>
<td>71.13</td>
<td>79.66</td>
</tr>
<tr>
<td></td>
<td><strong>Our S3Former</strong></td>
<td>58.69</td>
<td>73.89</td>
<td>82.87</td>
</tr>
</tbody>
</table>

on Figure 1.5.

**Small objects:** Fig. 3.1 highlights S3Former’s proficiency in identifying small objects, even as diminutive as solar cells measuring $10 \times 5$ pixels, underscoring its potential in high-resolution aerial imagery. Specifically, in the USGS, California example (Figure 3.1 c)), S3Former excels in identifying a small PV module situated at the top-left of the image, a task where the compared methods fail. Figure 3.1 a) further reinforces this observation by illustrating the enhancements made by S3Former in accurately segmenting the target PV installations.

**Intra-class heterogeneity:** Figs.3.1 a), b), c) illustrate the diverse PV installations even within a single rooftop. The variety in shapes, colors, and orientations poses challenges for deep learning. Notably, S3Former accurately maps varied color cells in Fig. 3.1 a) and identifies accurate boundary in Fig. 3.1 b).

**Inter-class heterogeneity:** Figs.3.1 b), c) depict challenges due to visual similarities between PV installations and other elements. In Fig. 3.1 b), while both DeepLabV3+ and UperNet falter due to the rooftop’s similar tone to PV installations, S3Former accurately masks it. In Fig. 3.1 c),
Figure 3.1: Qualitative comparison on (a) IGN France, (b) GGE France, (c) USGS California. From top to bottom: Original RGB Image, Groundtruth, Upernet (Xiao et al., 2018) and DeepLabv3+ (L.-C. Chen et al., 2017) and our S3Former.
DeepLabV3+ erroneously misidentifies a pool reflection as a solar cell. Pools and windows are prone to create reflections under certain solar conditions which, due its similar tone and shape with PV modules, are mistaken by segmentation networks.

*Overall performance:* Fig. 3.1 exhibits S3Former’s superior ability to accurately depict PV installations, outperforming baseline models in clarity and precision. It consistently identifies minute panels throughout the image and consistently offers robust results with minimal background confusion.

**Comparison: SolarFormer**

In this section, we conduct a qualitative comparison between S3Former and SolarFormer to validate the enhancements introduced by the SSL pre-training. We will revisit the challenges discussed in the previous section, as highlighted in Figure 3.2.

*Small objects:* Fig.3.2 showcases the improvements of S3Former and SSL. Fig. 3.2 a) and c) showcase sharper lines and more accurate boundaries on solar modules produced by S3Former. This behavior is reaffirmed in Panel b), where S3Former effectively masks a small indent at the bottom of the solar installation.

*Intra-class heterogeneity:* Figs.3.2 a), b), c) illustrate observations from Figure 3.1. Both models excel at masking a diverse array of shapes, colors, and orientations of solar cells.

*Inter-class heterogeneity:* Figs.3.2 b), depicts a case where CNN models traditionally struggle: an object situated next to a solar installation with a similar size and color. Both models accurately discard this element as part of the solar mask. However, Figs.3.2 a) and c) contain instances of S3Former outperforming SolarFormer in identifying erroneous elements and eliminating them from the result mask.

*Overall performance:* Fig. 3.2 exhibits S3Former’s superior capability to delineate small installations and enhance the model’s proficiency in distinguishing solar cells from background objects. This reinforces the robust features extracted by the SSL pre-task in S3Former.
Figure 3.2: Extended qualitative comparison on (a) IGN France, (b) GGE France, (c) USGS California. From left to right: RGB Image, Ground-Truth, SolarFormer and our S3Former. We selected special cases highlighting the strength of both models and improvements of S3Former with respect to SolarFormer on: inter-class homogeneity (green), intra-class heterogeneity (red), and small-object identification (blue). We further highlight cases of missing annotations on the data (purple).
Table 3.8: Ablation study performed on GEE, France dataset on three different augmentations

<table>
<thead>
<tr>
<th>Backbone</th>
<th>SSL</th>
<th>GGE, France</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Color</td>
<td>Gaussian</td>
</tr>
<tr>
<td>ResNet-50</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td></td>
<td>×</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>√</td>
<td>×</td>
</tr>
<tr>
<td></td>
<td>√</td>
<td>√</td>
</tr>
</tbody>
</table>

Table 3.9: Ablation study performed on USGS, California dataset on three different augmentations

<table>
<thead>
<tr>
<th>Backbone</th>
<th>SSL</th>
<th>USGS, California</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Color</td>
<td>Gaussian</td>
</tr>
<tr>
<td>ResNet-50</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td></td>
<td>×</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>√</td>
<td>×</td>
</tr>
<tr>
<td></td>
<td>√</td>
<td>√</td>
</tr>
</tbody>
</table>

Table 3.10: Ablation study performed on IGN, France dataset on three different augmentations

<table>
<thead>
<tr>
<th>Backbone</th>
<th>SSL</th>
<th>IGN, France</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Color</td>
<td>Gaussian</td>
</tr>
<tr>
<td>ResNet-50</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td></td>
<td>×</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>√</td>
<td>×</td>
</tr>
<tr>
<td></td>
<td>√</td>
<td>√</td>
</tr>
</tbody>
</table>

3.7 Ablation Studies

In the following section, we delve into an analysis of the diverse augmentations applied to the SSL module of S3Former. The objective of this investigation is to substantiate the impact of the varied transformations implemented during the training of the ResNet-50 backbone for high-resolution feature extraction. This study systematically scrutinizes the effect of each augmentation by isolating it and conducting experiments analogous to those detailed in the preceding sections. The consistency of improvements achieved by ResNet-50 is underscored in Tables 3.8, 3.9, and 3.10 across all scrutinized datasets. Noteworthy is the uniformity observed in the values for IoU, F1-score, and Accuracy, consistently surpassing the performance of the baseline SolarFormer model, especially on the GGE, France dataset. USGS, California results demonstrate that less augmentations
can also lead to great improvements, showcasing its best results when only Gaussian Blur is applied.
Chapter 4

CONCLUSION AND FUTURE WORK

In this research, we introduced S3Former, an innovative Transformer-based model for solar PV profiling. Leveraging the Transformer architecture, S3Former adeptly learns high-resolution features and attend to the finer details while retaining complex relationships between them. This not only empowers S3Former with a comprehensive context but also lets it extract rich visual cues, as underscored by the robust results achieved. Thus, S3Former is able to address the solar PV challenges including low-resolution, inter-/intra-class homogeneity.

We introduce Self-Supervised learning to capture strong relationships between the features extracted for a pretext task. We then transfer this network to the backbone of our model to obtain more explicit information for the semantic segmentation task. We validate the efficacy of this approach by conducting ablation studies as well as showcasing the improvements of the new architecture vs our previous model, SolarFormer.

We benchmarked S3Former against current SOTA Solar-PV profiling methods and evaluated its performance alongside various CNN-based and Transformer-based segmentation networks using both ResNet-50 and ResNet-101 backbones. Our comprehensive quantitative analysis confirms that S3Former surpasses the results of other prevalent segmentation networks. Furthermore, our qualitative findings underscore its enhancements over prior methodologies.

Included below are some of the future directions that could be further investigated:

**Improving network architecture:** In a fast-moving field like Computer Vision, novel ground-breaking ideas are the norm. Our research centers on two technologies that have recently captured the attention of the literature, yet there remains ample room for refinement. As illustrated in this study, SSL has showcased remarkable capabilities in extracting robust relationships among image features. However, as evidenced by Figure 4.1, the trajectory of SSL and SSL for Remote Sensing applications is on the upswing. Looking ahead, delving into more representations fine-tuned for semantic segmentation, as exemplified by the work of DINOv2 (Oquab et al., 2023), holds promise
Figure 4.1: Number of recent publications in SSL and SSL learning in recent years (Y. Wang et al., 2022).

for further enhancing results.

Regarding our segmentation network there are multiple models available to perform the same downstream task. Despite the recent dominance of Transformer-based architectures coupled with attention mechanisms in the literature, recent works such as URUR (Ji et al., 2023) demonstrate that CNN networks remain a competitive solution for aerial imagery segmentation. Mask2Former (Cheng et al., 2022) remains a topic of interest, with works such as DFPQ (H. He et al., 2023) proposing a query design focused on providing fine-grained positional priors to localize the target segments and improving the cross-attention mechanism. Integrating these novel concepts into S3Former could potentially yield improved representations for the semantic segmentation of solar installations.

**Integration for downstream characterization task:** The primary objective of this study is to provide a comprehensive overview of the advancements made by S3Former for the semantic segmentation task for solar installations. However, it is important to contextualize it as another element meant to enhance our understanding of the ongoing expansion of solar energy. Much like the insightful analyses conducted by researchers such as (Kasmi, Dubus, et al., 2022), the ultimate aim is to integrate our work into a pipeline geared towards generating real-time solar
power estimations for a designated geographical area.

The creation of an exhaustive map detailing the distribution of small-scale solar modules is key for facilitating such analyses. This data can be interpolated and compared with publicly available solar energy power reports to validate power output predictions. Furthermore, mirroring the methodology outlined in Section 1.3 by incorporating environmental measurements is also an interesting concept worth exploring.

In pursuit of this objective, there is a proposal to compile a dataset specifically tailored for analyzing the power generation efficiency of solar panels. While previous authors (Mehta et al., 2018) have suggested data in this format, their oversight in utilizing overhead imagery has limited the applicability of their findings to aerial imagery, the predominant format for capturing images of solar modules.


Dosovitskiy, Alexey, Lucas Beyer, et al. (2021). “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: ICLR.


INDEX

Symbols
3D-PV-Locator, 34
A
Accuracy, 32
C
Computer Vision (CV), 11
Convolutional Neural Networks (CNN), 6
D
Deep Learning (DL), 3
DeepSolar, 14
DINO, 18
F
F1-score, 32
G
Gated Recurrent Units (GRU), 9
Ground Sampling Distance (GSD), 28
H
Hyperion-Solar-Net, 34
I
Intersection over Union (IoU), 32
L
Long Short-Term Memory (LSTM), 8
M
Mask Loss, 24
Matching Loss, 23
Multilayer Perceptron (MLP), 5
P
PointRend, 22
PV Energy, 1
R
Recurrent Neural Networks (RNN), 8
Remote Sensing (RS), 28
Residual Networks (ResNet), 18
S
S3Former, 25
Self-Supervised Learning (SSL), 25
softmax function, 5
SolarFormer, 18
T
Transformer, 10
Transformer-Decoder Loss, 22
V
Vision Transformer (ViT), 11