Lacuna Solar Survey Challenge: Counting Photovoltaic and Solar Panels from Aerial Imagery

Hugo Veríssimo
Complements of Machine Learning 24/25
University of Aveiro
Aveiro, Portugal
hugoverissimo@ua.pt

João Cardoso
Complements of Machine Learning 24/25
University of Aveiro
Aveiro, Portugal
joaopcardoso@ua.pt

***Abstract*—abstact**

Keywords: R, Object Detection

I. Introduction

Access to electricity and warm water has been a basic necessity for a long time in the developed world. In developing countries, the lack of a centralized distribution system makes this access harder for everyone. To improve on this gap, it is essential for governments, non-governmental organizations, and energy suppliers to understand how solar photovoltaic and solar thermal panels (for electricity and water heating, respectively) are distributed throughout the territory to ensure proper planning and effective policy making. In this context, a challenge was created by the Lacuna Fund and associate entities in the Zindi platform to develop a machine learning model capable of accurately detecting and counting the number of solar thermal (solar from here on) and photovoltaic panels in drone and satellite imagery.

In the present work, we have developed different approaches to the problem at hand, where we need to count the two types of panels separately: taking a regression type approach, where images are analysed as a whole and the target number of panels are provided per image; by identifying the panels through object detection and counting them afterwards; and by applying segmentation models to identify the regions of interest, analysing them and counting the panels. The different approaches were developed so that a single model for each type would be able to tackle the task of identifying and counting the different panels.

II. State of The Art

The task of detection of solar panels serves many purposes, with the strongest focus of the works here documented being on the detection and localization of solar panels in large areas (countries). This type of assessment allows for proper planning of infrastructure, and to adapt current maintenance through peak output of dense areas with solar panels.

In the work of Malof *et al.* the authors developed an automatic detection model of solar arrays, specific for photovoltaic panels. The model developed had computational efficiency at the forefront, with the purpose of being deployed nation wide in the United States, hence a multi-stage approach consisting of: pixel-wise feature extraction; Random Forest (RF) Classifier; post-processing to improve the pixel-wise classification accuracy for the object detection phase; finalizing with the object detection, via thresholding the finalized confidence map. This algorithm presents superior performance to standalone RF, and was well remarked as an initial assessment methodology for large areas.

The *DeepSolar* model consists allowed to create a nearly complete contiguous solar panel installation map, and consists of two parts, with different purposes: firstly, via transfer learning the convolutional neural network (CNN) classifier is trained on labeled imagery that merely indicates the presence or absence of panels (though the volume of data is considerable, at almost 400 000 images just for this task). The model is then enhanced by adding an additional CNN branch directly connected to the intermediate layers to add segmentation capabilities to the model. The difference is that this approach allows the model to be trained on the same dataset, to ”greedily” extract the relevant features associated with solar panels, learning in a semi-supervised manner. The model was able to achieved the best result of 93.7 % precision and 90.5 % recall in non-residential areas.

Different models have been proposed for panel detection through image segmentation, using Mask R-CNN He, 2020, TernausNet, based on U-Net Kausika 2021, MobileNet with U-Net backbone Wari, 2021, with different approaches to data preparation and application. These models consistently score higher than 90 % in terms of precision, at the cost of higher computation times compared to the examples previously mentioned (also for very different task sizes). A couple notable aspects across publications: the size of datasets (often in the tens of thousands of images), with strong strategies for collecting and/or preprocessing, ensuring that the models have the best possible starting point to learn and develop; and the focus on a specific type of panel or problem topic, not trying to solve multiple problems with a single tool.

For the purpose of this work, we have developed three approaches to the problem, to assess their strengths and merits, which will be detailed in the next subsections.

A. Image-based Regression

Considering that every image is labelled in terms of amount of photovoltaic and thermal solar panels, the conditions for training considering a regression approach are available. For this, deep neural networks (convolutional neural networks, CNN) were considered for their excellent capabilities in extracting spatial features, but each with a distinguishing feature.

ResNet (Residual Networks) was introduced in 2015 by Microsoft, and is notorious for dealing with vanishing gradient problem by skipping connections between neuron layers. This allows the model to grow and have several layers (over 100), making it capable of learning a larger variety of features from the first to the deeper layers, without suffering from performance degradation.

DenseNet follows on the footsteps of ResNet, and was introduced in 2016 by Huang *et al.*, whereby the idea of skipping connections is taken further, and every layer prior to a given layer is connected to it. This avoids the chain multiplication problem (which leads to vanishing gradients), while reusing previously learned features, retaining relevant information throughout the deepness of the CNN. This architecture also has the benefit of using less parameters than CNN in general (and ResNet as well).

EfficientNet (EffNet), the most recent of this group was introduced by Google AI in 2019, and departs from the direct implementation of the aforementioned methods, dealing with vanishing gradients given its intrinsic architecture. By having MB Conv (Mobile Inverted Bottleneck Convolution) as its building block, it allows to extract spatial features by expanding the number of channels to perform depthwise convolution and applying separate filters per channels **(verificar a nomenclatura)**, that is then compressed to the original size of the input. Whilst this happens, the mechanism of squeeze-and-excitation determines the importance of said features, further addressing the vanishing gradient problem. This architecture is then scaled via compound scaling (rather than arbitrarily), whereby the size of the CNN (depth x width and input resolution) is scaled in a balanced manner, with the compound coefficient ϕ being adjusted by grid search.

B. Object Detection

With object detection, the model is trained based on bounding boxes surrounding the subject of interest, in order to be able to identify them in new environments and types of images. With multiple classes, the model discern between labels with class probabilities, estimated from the extracted features of the assessed boxes (using a CNN) throughout the image. One of the most famous and used models is YOLO ("You Only Look Once") introduced in 2015 by Redmon & Fardhi, being a crucial model in fast object detection and localization. In a single stage, the model addresses the localization and detection problems, where a CNN backbone (that has changed with the released versions) is used, with feature fusion layers (akin to the way learned features are reused, involving various algorithms), and an output of the likelihood of a class and the location of the bounding boxes.

C. Instance Segmentation

Building on the concepts presented before on object detection, instance segmentation addresses the problem of identification by incorporating the information of object masks. These are tighter around the object (than bounding boxes), which serve to add a layer to the YOLO algorithm. A global prototype mask (that works with a separate, smaller CNN) is generated for the whole image - this way once the bounding box is determined, mask coefficients are estimated in order to combine the prototype masks into an instance specific mask. This approach avoids per mask full resolution segmentation, keeping the model size small and fast.

III. Methodology

A. Dataset & EDA

The dataset was provided within the Zindi challenge, consisting of 4419 images (from drone and satellite sources), and per image metadata, consisting of the source (drone or satellite), mask placement, and context of the installation surroundings (e.g., roof, floor, array).**info do pie chart** The proportion between photovoltaic and thermal solar panels is heavily imbalanced, with 95 % of the dataset corresponding to photovoltaic panels. Of all the images, 3312 (75 %) have detailed information on the masks (location, number of panels within, if they are photovoltaic or thermal). The remaining 1107 (25 %) do not have this information, only the number of photovoltaic and thermal panels in the image. Hence, per design of the competition the train/test split is 75/25.

Fig. 1: Images of photovoltaic panels placed on the roof, from drone (left side) and satellite (right side). The difference in resolution between them is evident.

Fig. 2: The different panel placement possibilities (top right is from satellite imagery). Bottom left image shows an image that is atypical for a drone style image.

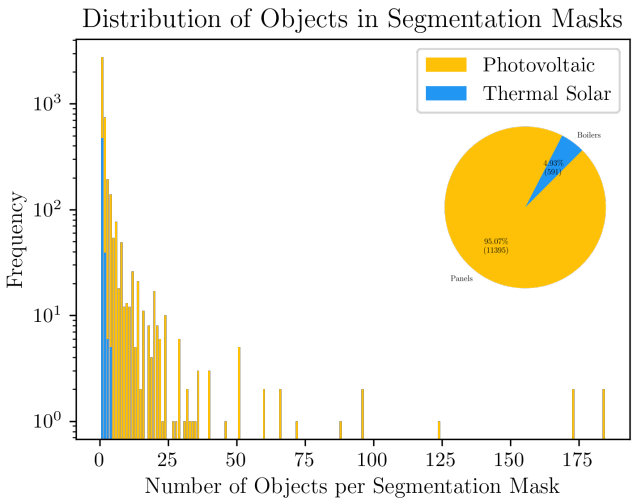


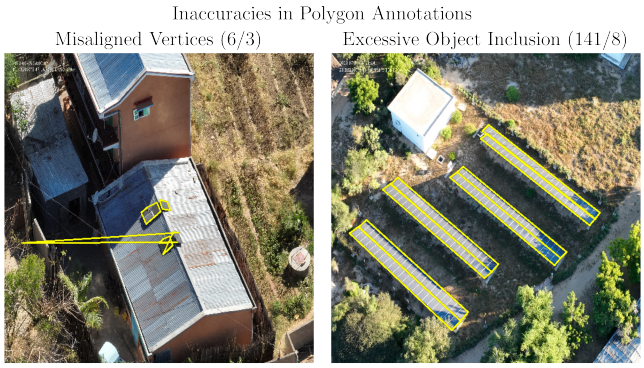
Fig. 3: Distribution of the number of panels within a single mask. -¿ verificar onde ainda se apanha boiler ou panel em vez da nomenclatura acordada

In Fig. 2 the image origin and placement are displayed, which according to the challenge information were labelled by expert personnel. Where the placement class wasn't conclusive, the images are labelled as "S-unknown" (the remaining examples are self explanatory). Besides the two classes to be identified, the context and origin of the images means a considerable number of combinations for the model to learn, where some of these combinations might (and certainly are), under-represented given the relatively small dataset.

The masks are not consistent throughout the dataset, with varying number of panels within them (some contain a single panel, others might contain a complete array of up to 200), which can be clearly seen in Fig. 3.

From the sample images below, the difference in the quality of the masks is stark. Several images were found to have misaligned vertices of the masks, and several masks had a large amount of panels (as seen from the distribution aforementioned). In principle, with the reference of the number of panels within each mask, it might have a small impact for some types of models, but the same model is in fact learning features for different representations: rather than for the representation of individual panels. This was an evident obstacle to the implementation of YOLO type models, which was dealt with, and further detailed below.

(a) Sample image of accurate labelling, with a single panel per mask.



(b) Sample images of incorrectly marked masks: (on the left) mask distorted, misrepresenting the panel; (on the right) excessive objects within a single mask.

Fig. 4: Sample images from the Zindi dataset.

B. Preprocessing

The discussion forum for the competition was a fruitful source of information on the dataset and how to address it. As seen above, the incorrectly defined masks, and the masks with several panels represented hindrances to achieve the best possible performance. Besides that, some of the masks were shifted from the actual position of the panels, which considering that all were identically shifted, seemed deliberate from the competition. By manually analysing every image it was possible to detect such images (with wrongly drawn masks and shifted), and correct them. Upon completing the dataset revision, 263 of the training samples were lost due to poorly drawn masks (the remaining images that had misaligned masks were corrected).

Once the dataset was ready, the (online) data augmentation process was devised in order to increase the number of training and diversify it, in order to enhance the generalizability of the trained models. The transformations HorizontalFlip, VerticalFlip, RandomRotate90, GaussianBlur, CLAHE (Contrast Limited Adaptive Histogram Equalization), HueSaturationValue and Normalize were applied before all training cycles. The images were all resized to 512x512, to lower the computational burden and homogeneize the code throughout the pipeline.

IV. Models

In the following subsections, A through C, we present the approach and results of the developed models. The hybrid model *zulo40* is presented as benchmark, as it was one of the best performing in the competition and was kindly shared in the discussion forum. The models from in the section *A. Image-based Regression* are based on this model's approach.

The training examples were split between training and validation (80/20) across all the models' training. The best set of hyperparameters was selected from a randomized search in a cross validation (CV) set up. Unlike what is commonly done with K-fold CV, due to the long processing times and high computational load, the number of CV repetitions was adjusted 3 times (and not the usual 5 times), and averaged over them. We acknowledge the downside of this approach, particularly the fact that only 60 % of the data is exposed to validation and the reduced diversity of the models trained. These aspects become particularly evident due to the relatively small size of the dataset.

A. Image-based Regression

This approach had the purpose of incorporating the information from the metadata in the labels with the feature extracted with CNN backbone by transfer learning. In Fig. 5 we present the schematics for the model's architecture.

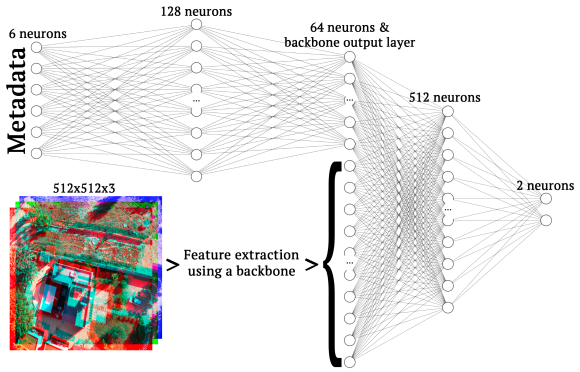


Fig. 5: Architecture of the model shared by user zulo40 in the discussion forums. [ref!]

After extracting the features they are processed through fully connected layers, where a multi-head attention mechanism is applied after to enhance embeddings. The multi-head attention mechanism is an important technique of CNN from the Transformers architecture, whereby the *attention* (i.e., the importance of certain features) is estimated several times in parallel, allowing to learn different patterns and relationships between different sets of features. Finally a regression head predicts the number of panels (photovoltaic and/or thermal solar) in a given image.

The backbones tested were DenseNet121, EfficientNetv2B3 and ResNet101, and after an initial assessment, EfficientNetv2B3 provided with the best results and the hyperparameters were further fine tuned with the ranges specified in Table I.

The criteria for selection of the best model was the MAE (mean absolute error, the metric used in the Zindi challenge), which gave the following hyperparameters marked in bold in Table I. The hyperparameters were selected for a small number of combinations, given how time consuming training each model is.

TABLE I: Hyperparameter space for the hybrid model *zulo40*. The bold values correspond to the selected hyperparameters.

Hyperparameter	**Possible Values**
Batch Size	$\{\mathbf{16}, 32, 64\}$
Optimizer	**AdamW**
Learning Rate	$[\mathbf{10^{-5}}, 10^{-3}]$
Weight Decay	$[\mathbf{10^{-5}}, 10^{-3}]$
Dropout	$\{0.2, 0.3, \mathbf{0.4}\}$
Scheduler	**CosineAnnealingWarmRestarts**
T_0	$\{3, \mathbf{5}, 7, 10\}$
T_mult	$\{1, \mathbf{2}, 3, 5\}$
Loss	**HuberLoss**
δ	**1**

Upon selecting the best model, the final inference on the test data was done resorting to TTA, Test-Time Augmentation, where the model makes multiple predictions on augmented versions of the same image (e.g., flipping, scaling, cropping). With this, the final result is an average of the group of predictions. The scheduler selected (CosineAnnealingWarmRestarts) dynamically changes the learning rate following a cosine schedule, with a defined period (T_0 = 5), where at each period the learning rate decays to a minimum, restarting at the highest set learning rate. This allows to prevent locking into local minima, by passing through higher learning rates periodically, allowing for fine tuning with lower learning rates for longer periods (given T_mult = 2, where the period increases two fold at each restart).

The erratic behaviour of the CV loss curve is evident from Fig. 6, likely due to the small validation set (20 % of an already small dataset), which leads to variance spikes due to the changing samples within. The training loss is rather smooth, with periodical increases well in line with the scheduler's algorithm (5, 5 + 10, 15 + 20, ...).

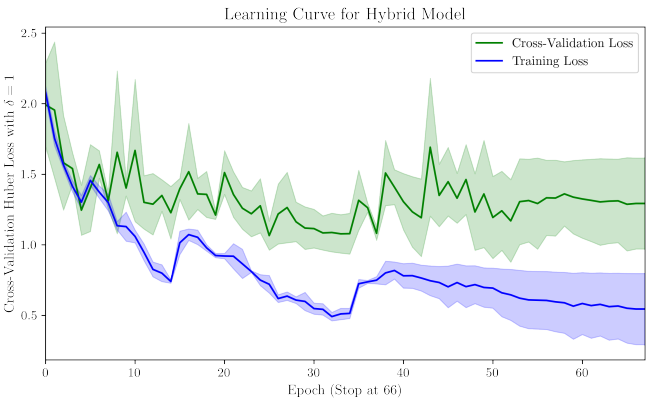


Fig. 6: Learning curve for the *zulo40* hydrid model, with Training and Cross-Validation Loss plotted (with standard deviation in shade).

The training proceeded for 75 epochs, where the best model was selected based on the best validation MAE. From the learning curve, it does not seem to be overfit, but considering the MAE metric and the test set from the Zindi challenge, the test MAE is considerably higher than the training MAE (Table II, indicating that the model is not capable of generalizing so well.

TABLE II: Error metrics for the Hybrid Model, for the train and test set, along with the number of samples for each.

Dataset	**MAE**	**Support**
Train Set	0.5127	3312
Test Set	0.8434	1107

B. Object Detection

Upon initial testing of YOLO models in the dataset, it was clear from the beginning that the labelling style was a major hindrance to the type of models resorting to the masks/labels, which resulted in significant loss in performance. Based on this assessment, we have individually reviewed roughly 300 images **(confirm number)**, and created individual masks for each panel (either solar thermal or photovoltaic), which resulted in roughly 1000 individual masks (mostly for photovoltaic). To train these models, only images with individual masks were considered, resulting in a reduction in size of the dataset of roughly **checkar qual o valor**.

As for the previous section, the criteria for selection of the best model was the MAE, which rendered the following hyperparameters marked in bold in Table III. The hyperparameters were selected for a small number of combinations, given how time consuming training each model is.

TABLE III: Hyperparameter space for the YOLO model. The bold values correspond to the selected hyperparameters.

Hyperparameter	Possible Values
Batch Size	$\{16, \mathbf{32}\}$
yolo	{yolo11l, yolo11m, **yolov8l**}
imgsize	**512**
augment	
textbfTrue	
patience	$[15, \mathbf{25}]$
cls	$[0.5, \mathbf{1.5}]$
lr0	$[\mathbf{20^{-5}}, 10^{-3}]$
lrf	$[0.1, \mathbf{1}]$
mixup	$[0, \mathbf{0.75}]$
copy_paste	$[0, \mathbf{0.75}]$
scale	$[0.5, \mathbf{1}]$

From the different model fits, in Fig. 7 the MAE ranges are displayed with boxplots, where the model YOLOv8L has been fit three different times for different combinations of hyperparameters, evidencing the relevance of this step. From the group, the model with lowest mean MAE was selected (the hyperparameters selected are in bold in Table III).

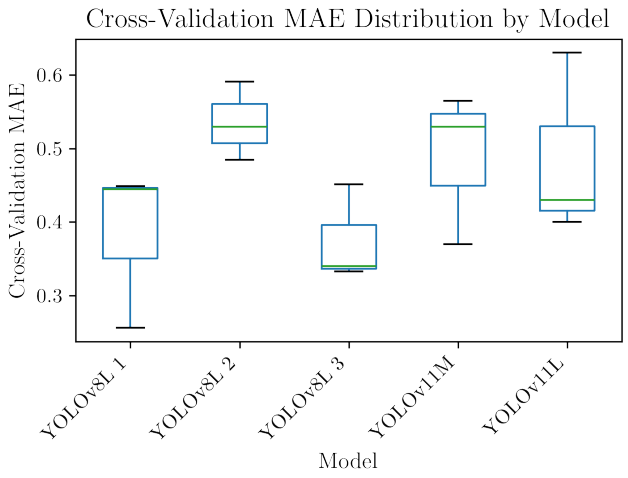


Fig. 7: CV MAE for each model. Each boxplot represents the values obtained during the cross validation of each model.

The results from the best model are shared below, where it is important to bear in mind that the metrics shared are for the model's main goal (i.e., object identification), and only after the training it was possible to count apply the model to the test set and count the number of panels of each kind.

Unlike in the previous section, the cross-validation and training loss curves do not evidence the spikes seen before (no learning rate scheduler used), but the axis scale might deceive the interpretation: the cross-validation loss still has a considerable variance in comparison with the training loss curve. This still is due to the difficult in generalizing the learning, with such small dataset (and the fact that 40 % of the data are never in the validation portion of the K-fold CV). Despite this fact, the learning curves progress well and do not evidence overfitting, although the variance of the training curve increases drastically at the very late stages of training.

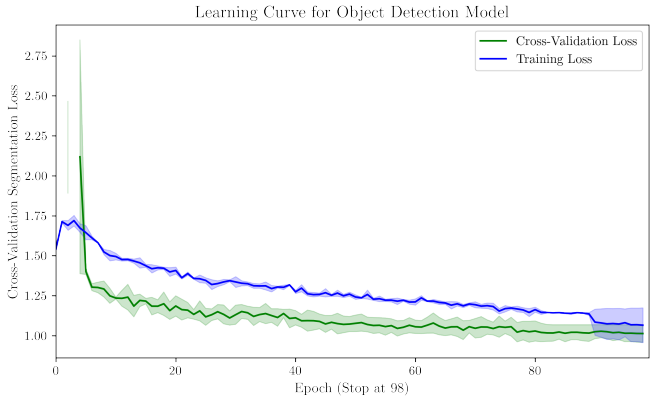


Fig. 8: Learning curve for YOLOv8L, with Training and Cross-Validation Loss plotted (with standard deviation in shade).

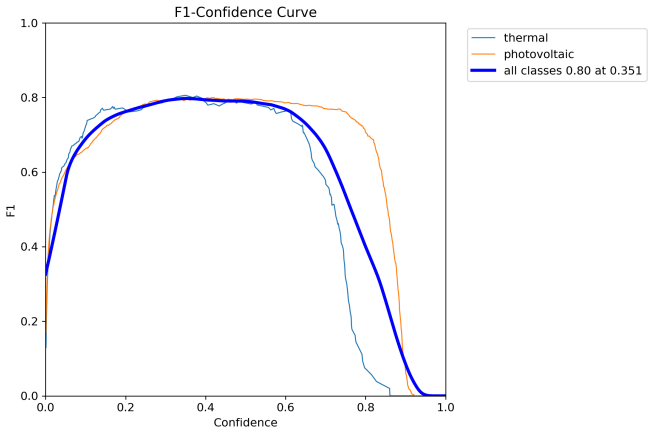


Fig. 9: F1-Confidence curves for YOLOv8L (graph automatically generated from the ultralytics package **não nos podemos esquecer de referenciar o código usado**).

The resulting F1-Confidence curve (Fig. IV) is a typical analysis for this type of models, as it relates two important metrics: F1-score, which is a result of the precision and recall of the predicted samples; and confidence, which consists of the product of the probability of a given object existing within the bounding box, and the probability of a given class given that object is in the box. The equation for confidence is available below.

$$\text{Confidence} = P(\text{object}) \times P(\text{class} \mid \text{object})$$

Moreover, confidence is a result of the training process, where during each image pass through the CNN the probabilities for a objectness (i.e., presence of object) and class are learned from training. This output serves as the final layer's activation, shaped by the training, which will reflect the

training and be used during inference stage to assess detection certainty.

The evaluation of predictions (i.e., evaluating true/false positive/negative) is based on IoU (Intersection over Union), comparing the predicted mask with the ground-truth for that image. It also allows to avoid the usage of overlapping predictions, removing redundant bounding boxes.

$$\text{IoU} = \frac{\text{Area of } (B_p \cap B_{gt})}{\text{Area of } (B_p \cup B_{gt})}$$

Despite our efforts to attain a better model, the MAE of the train and test set was among the worst evaluated, especially when considering how low the CV training MAE was.

TABLE IV: Error metrics for the Object Detection Model, for the train and test set, along with the number of samples for each.

Dataset	MAE	Support
Train Set	1.4330	3312
Test Set	1.2645	1107

C. Instance Segmentation

o conjunto de pesquisa de hyperparemetros é dado pela tabela seguinte, mas é de notar q nsao sao testadas todas as combinacoes mas apenas algumas aleatorias

TABLE V: yolo seg model hyp

Hyperparameter	Possible Values
Batch Size	$\{\mathbf{8}, 32, 16\}$
yolo	{**yolo11l-seg**, yolo11m-seg, yolov8l-seg}
imgsize	**512**
augment	**True**
patience	$[10, \mathbf{25}]$
cls	$[0.5, 2.5]$
lr0	$[10^{-4}, \mathbf{10^{-3}}]$
lrf	$[0.01, \mathbf{0.1}, 1]$
mixup	$[\mathbf{0}, 0.5]$
copy_paste	$[\mathbf{0}, 0.8]$
scale	$[\mathbf{0.5}, 1]$

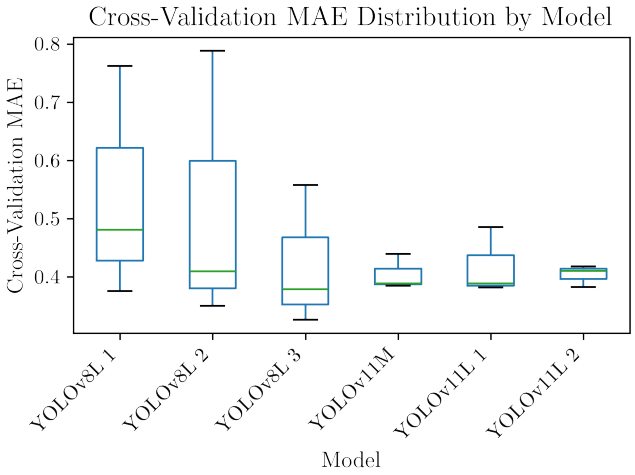


Fig. 10: CAPTION CAPTION CAPTION CAPTION CAPTION -¿ acrescentar seg na designação dos modelos, senão vai gerar confusão

a fig 10 demonstra o mae para os diferentes modelos ajustados na busca dos melhores hyperparemtros, tendo sido o YOLOv11L 2 a obter o menos MAE médio nas suas validações, tendo sido ele o escolhido. seguem-se abaixo os seus hyperparemetros

- yolo yolo11l-seg - imgsize 512 - augment True - patiente 25 - lr0 10^{-3} - lrf 0.1 - cls 0.5 - mixup 0 - copy paste 0 - scale 0.5

os restantes parametros sao os que vêm por defeito no package ultralytics referentes ao modelo yolo11l-seg

quanto aos resultados do modelo em referencia, é de notar que o mesmo foi treinado tendo em conta o segmentar os objetos (pan e boilers) da melhor forma possivel, pelo que as metricas do ajuste do modelo sao referentes a isso e nao ao problema de contagem, sendo esta contagem apenas feita apos o modelo ajustar os segmentos as imagens

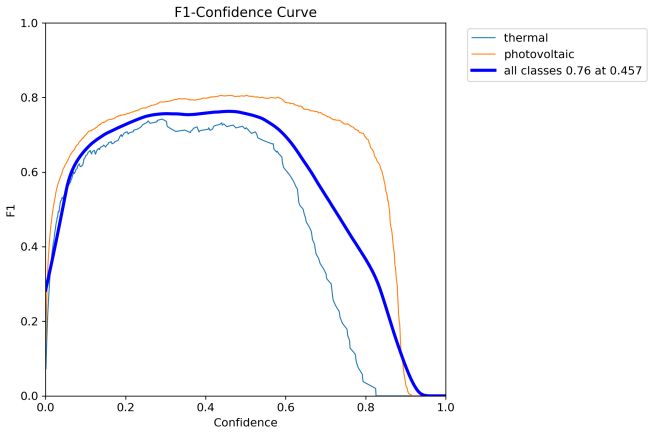


Fig. 11: CAPTION CAPTION CAPTION CAPTION CAPTION (gerado pelo modulo do yolo automaticamente) referir isso pq ele é ”diferente” dos outros ig

a fig. 11 mostra a metrica f1 relativa a segemntacao do segundo fold do modelo, revelando um melhor desempenho dos photovoltaic em relacao aos thermal, mas sem uma diferenca significativa.

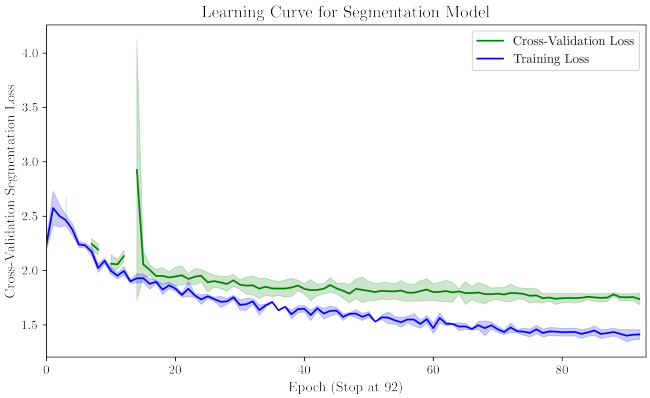


Fig. 12: CAPTION CAPTION CAPTION CAPTION CAPTION

na fig. 12 pode-se verificar a learning curve do melhor modelo, a demonstrar a sua convergencia sem sinais de overiffint. apenas com um possivel outlier na loss, que pode ter devido a algum erro de computacao por parte do calculo do segmentation loss por parte do yolo. é possível ver tbm a paragem do modelo após 92 epochs por early stopping.

TABLE VI: Error metrics for the Segmentation Model, for the train and test set, along with the number of samples for each.

Dataset	**MAE**	**Support**
Train Set	1.5645	3312
Test Set	1.3415	1107

quanto as metricas relacionadas com a contagem dos pan e boil, o nosso problema original, o seu resultado é apresentado na tabela VI

V. Discussion

https://zindi.africa/competitions/lacuna-solar-survey-challenge/discussions/25674

A. Performance Metrics

discussao

VI. Conclusion

conc

Work Load

Both authors contributed equally to the project.

References