

Lawrence Berkeley National Laboratory

LBL Publications

Title

Automated defect identification in electroluminescence images of solar modules

Permalink

<https://escholarship.org/uc/item/2mt97497>

Authors

Chen, Xin

Karin, Todd

Jain, Anubhav

Publication Date

2022-08-01

DOI

10.1016/j.solener.2022.06.031

Copyright Information

This work is made available under the terms of a Creative Commons Attribution-NonCommercial License, available at <https://creativecommons.org/licenses/by-nc/4.0/>

Peer reviewed

Automated Defect Identification in Electroluminescence Images of Solar Modules

Xin Chen *, Todd Karin *, Anubhav Jain *

*Lawrence Berkeley National Laboratory, Berkeley, CA, U.S.A

Abstract

Solar photovoltaic (PV) modules are susceptible to manufacturing defects, mishandling problems or extreme weather events that can limit energy production or cause early device failure. Trained professionals use electroluminescence (EL) images to identify defects in modules, however, field surveys or inline image acquisition can generate millions of EL images, which are infeasible to analyze by rote inspection. We develop a rapid automatic computer vision pipeline (~ 0.5 seconds/module) to analyze EL images and identify defects including cracks, intra-cell defects, oxygen-induced defects, and solder disconnections. Defect identification is achieved with a machine learning model (Random Forest, ResNet models and YOLO) trained on 762 manually-labeled EL images of PV modules. We compare model performance on an imbalanced real-world validation set containing 134 EL images and determine that ResNet18 and YOLO are the optimal models; we next evaluated these models on a dedicated testing set (129 module images) with resulting macro F1 scores of 0.83 (ResNet18) and 0.78 (YOLO). Using a field EL survey of a PV power plant damaged in a vegetation fire, we analyze 18,954 EL images (2.4 million cells) and inspect the spatial distribution of defects on the solar modules. The results find increased frequency of ‘crack’, ‘solder’ and ‘intra-cell’ defects on the edges of the solar module closest to the ground after fire. We also find an abnormal increase of striation rings on cells which were assumed to be caused mainly in fabrication process. Our methods are published as open-source software. It can also be used to identify other kinds of defects or process different types of solar cells with minor modification on models by transfer learning.

Keywords

Solar Module Defect, Electroluminescence Image, Deep Learning, Computer Vision, Big Data

I. INTRODUCTION

With the continuing push toward higher performance and reliability of photovoltaic (PV) modules to enable the energy transition, diagnostic tools to assess module damage during manufacturing, installation or operation are becoming ever more important. Electroluminescence (EL) imaging is a fast, non-destructive and established method for detecting defects in solar modules [1], [2], [3], [4]. Although EL images can be acquired very quickly on the ground or by drone [5], it takes a trained professional 10-30 seconds to analyze each resulting image, severely limiting the number of modules that can be inspected. Further, different examiners can make different conclusions from the same dataset, limiting the extent to which comparisons can be made [6].

As machine learning has become increasingly fast and accurate, it is now possible for computers to perform tasks that would otherwise require a large labor expenditure. Previous research has shown that the application of computer vision and machine learning has significant potential in the analysis of EL images, for example using independent component analysis [7], anisotropic diffusion filter followed by support vector machine (SVM) [8], random forests [9], [10], *etc.* Ever since around 2010, convolutional neural networks (CNNs) have been significantly developed and widely used in image classification and object detection, *e.g.*, AlexNet [11], VGG [12], ResNet [13] and YOLO [14], [15]. The automatic feature engineering and outstanding performance trigger an increasing trend of application of CNN models in EL image analysis.

Deitsch *et al.* [16] performed one of the earliest studies of applying CNN models to EL image analysis. The authors trained a CNN model with an accuracy of 88.42%, which exceeded the performance of an SVM model utilized in their paper. The authors also published their data set consisting of 2,624 cell images as a benchmark [17], [18]. Based on this public data set, Akram *et al.* [19] utilized data augmentation and designed a light CNN model with 93.02% accuracy. However, all these models can only perform binary classification (‘functional’ or ‘defective’) without determining the specific defect categories.

Karimi *et al.* [10] designed an end-to-end pipeline to process EL images. In the pipeline, modules in raw EL images were first transformed and cropped into individual cell images automatically. They then trained three models, including random forest

The project was primarily funded and intellectually led as part of the Durable Modules Consortium (DuraMAT), an Energy Materials Network Consortium funded under Agreement 32509 by the U.S. Department of Energy (DOE), Office of Energy Efficiency & Renewable Energy, Solar Energy Technologies Office (EERE, SETO). Lawrence Berkeley National Laboratory is funded by the DOE under award DE-AC02-05CH11231.

The authors declare no conflicts of interest. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. government. Instruments and materials are identified in this paper to describe the experiments. In no case does such identification imply recommendation or endorsement by LBL. The U.S. government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. government purposes.

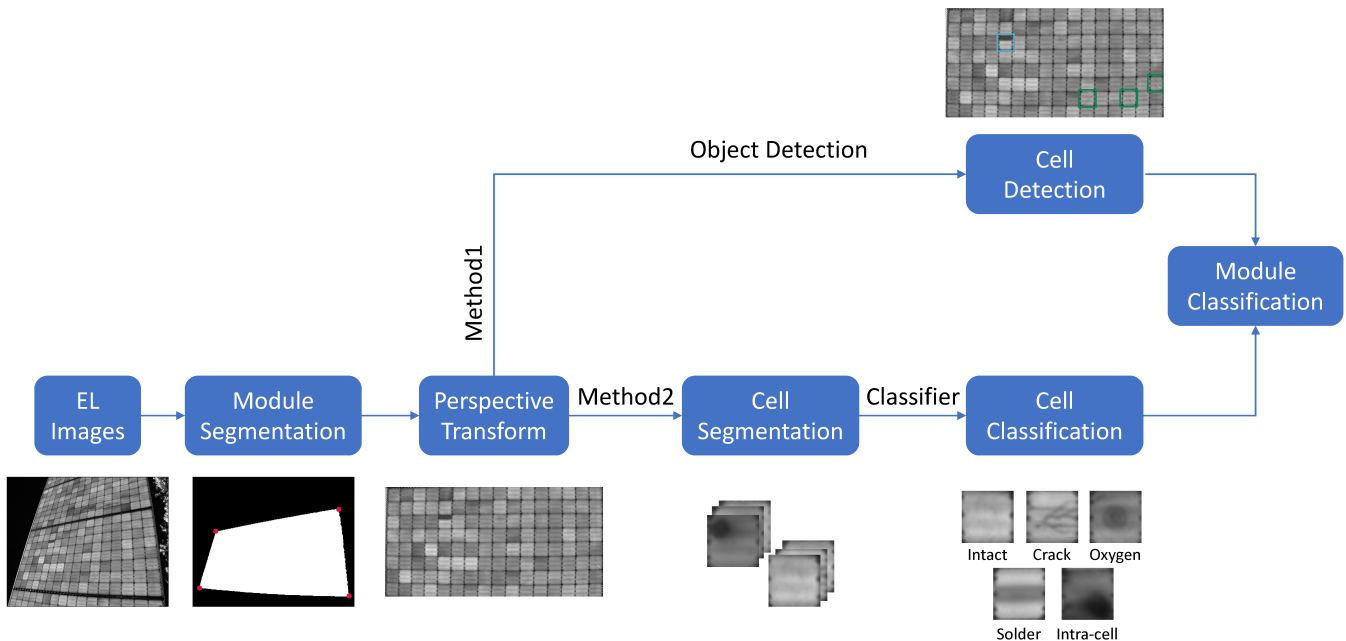


Fig. 1. The process of the automatic pipeline.

(RF), SVM, and CNN to classify cells into three categories: ‘good’, ‘cracked’ and ‘corroded’. The CNN model outperformed other models with a 99.71% accuracy on their own testing set. The automatic preprocessing tool is impressive. However, the background in their images is relatively clean and free of other objects, so the tool requires further testing on field images with complex backgrounds. Karimi *et al.* [20] also did unsupervised learning to cluster EL images into degraded and non-degraded categories.

Tang *et al.* [21] utilized a generative adversarial network (GAN) to expand the training set of a limited sample size. Their CNN models were trained on the generated images, and high accuracy (over 80%) was obtained for detecting ‘defect-free’, ‘micro-crack’, ‘finger-interruption’ and ‘break’ categories. They also compared the performance of various CNN models, including VGG16, ResNet50, Inception V3 and MobileNet using transfer learning.

Apart from the image classification technique, object detection is also applied in EL image analysis [22], [23], [24]. Object detection can not only identify different defects but also localize the position of defects with bounding boxes. Zhao *et al.* [23] trained a CNN model to localize 14 types of defects, achieving a mean average precision of 70.2%. Meng *et al.* [24], based on YOLO model, designed YOLO-PV with 94.55% of average precision and 35 fps inference speed.

Semantic segmentation is also applied to detect defects on pixel-level. Mayr *et al.* [25] used ResNet50 as backbone and normalized L_p layer to segment cracks from cell images. Fiorese *et al.* [26] utilized a Deeplabv3 model with ResNet50 backbone to segment cracks and achieved a weighted F1-score of 0.95.

While summarizing the progress in automatic analysis of EL images, we find that there are still demands for further exploration from various aspects: (1) Most of the work mentioned above focused on training a classification/object detection model. However, data preprocessing requires further attention, especially for field images with a poor background obstructing the automatic transformation from raw EL images to individual cell images. (2) Object detection methods are impressive for localizing defects, but it remains unknown whether its performance in identifying defects is competitive with classification models. (3) Most tools developed in the papers are not open-source and cannot be readily tested or applied.

In this paper, we developed an open-source pipeline to analyze EL images of PV modules. As an overall dataset, we used 19,228 EL images of solar modules (16×8 cells) from a 50 MW DC power plant located in Southern California, with data acquired and provided by PV Evolution Labs (PVEL) [27]. We focused our analysis on four key defects: cracks, striation defects, intra-cell defects and solder disconnection. The variety of EL images observed for each defect type makes this a prime problem addressable with machine learning tools.

This paper presents the image preprocessing tools and the two methods for defect identification. Image preprocessing first uses deep learning to detect the contour of the tilted solar modules in the EL images and perform perspective transform; it subsequently crops out single solar cells. We investigated two strategies of identifying defects: object detection and classification. The object detection method starts with an image of an entire solar module and puts a bounding box on each defective cell (*i.e.*, the object to identify). The classification method begins with an image of a single PV cell and classifies the cell into a category (*e.g.*, intact cell, cracked cell, cell with solder disconnection, *etc.*). We trained the YOLO [15] model for object detection and ResNet18, ResNet50 and ResNet152 [13] models for classification. We also trained a Random Forest (RF) classifier as a

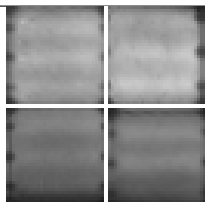
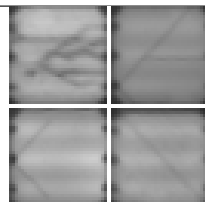
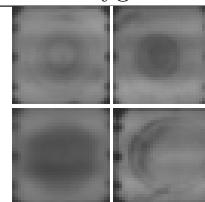
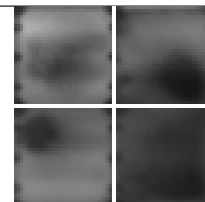
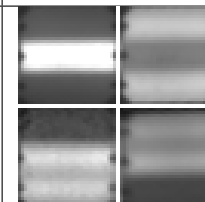
Category	Intact	Crack	Oxygen	Intra-cell	Solder
Image					
Training 762 modules	95048 97.44%	1367 1.40%	709 0.73%	279 0.29%	143 0.15%
Validation 134 modules	16618 96.87%	345 2.01%	127 0.74%	47 0.27%	18 0.10%
Testing 129 modules	16082 97.4%	244 1.48%	126 0.76%	45 0.27%	17 0.10%

TABLE I
IMAGES OF CELLS WITH DIFFERENT DEFECTS AND DEFECT TYPE BREAKDOWN IN THE TRAINING, VALIDATION AND TESTING SET.

baseline.

We compared the model performance on the real-world dataset and quantified the performance using recall, precision, F1 score and F2 score [28]. We further applied the YOLO model on all 18,954 successfully transformed EL images (around 2.41 million cells) of solar modules in the overall dataset. We presented the distribution of the defective cells detected by the YOLO model and found that cracks, solder disconnection and intra-cell defects are more prevalent on the two shorter edges of the PV module. We also found an abnormal increase of striation defects.

A schematic of the analysis pipeline as mentioned above is depicted in Figure 1. Whether the object detection or classification method is best depends on the demand of the users; their accuracy is similar. For object detection, the solar modules do not need to be segmented into single solar cells and bounding box can be directly viewed on solar modules. For cell classification, single cells need to be cropped out from the module images but the classification costs less computing resource. The ‘module classification’ in the final step is based on the number of different defects in the solar modules. The criterion is given by the users.

In summary, the main contributions of this paper are:

- (1) We published an open-source pipeline that includes preprocessing, classification/detection and post-processing.
- (2) Our tools can handle field images with a complex background (*e.g.*, vegetation, racks, other modules). Most of other papers report results on images with a clean background (totally dark background or with very little noise).
- (3) We compared the performance of classification and object detection neural networks on our validation set.
- (4) We applied our pipeline to field images to find potentially new phenomena (the abnormal growth of striation defects) using a large data set affected by fires (2.41 million cells).

The remainder of this paper is organized as follows: Section II presents the description of our dataset. Section III provides details of image preprocessing tools and models. Section IV shows the results of our processing pipeline and the comparison of various models’ performance. Position distribution of defects from big data is also presented in this section. The conclusions and outlook are given in section V.

The algorithms and trained models developed in this paper are available as part of the pv-vision open source software [29].

II. DATASET

Our dataset is composed of 19,228 EL images of interdigitated back contact (IBC) solar modules (16×8 cells). Each raw image has a size of 640×512 pixels. Our perspective transform tool finds the target module and transforms it into a module image with a size of 600×300 , as shown in III. This dataset is the result of a field survey performed to determine which modules to replace after a vegetation fire occurred beneath the modules in the plant. The EL images were acquired during daytime using an InGaAs camera and lock-in detection using the commercial DaySy system [30]. In this research, we focused on four categories of defects: cracks, striation rings caused by oxygen impurities during fabrication, intra-cell defects caused by series resistance to metal contacts or alternate carrier recombination pathways and defects caused by solder disconnection [31]. Cells with other features are considered as intact class. For instance, some cells may have potential induced degradation but we still label them as ‘intact’ because they do not contain one of the target defect types. Examples of the identification of these defects from the EL image is shown in Table I. The ‘oxygen’ category denotes the striation-ring defects.

18,954 images of the whole dataset could be successfully transformed by our image preprocessing tools and were used for further investigation. Among these 18,954 modules, 129 of them are in control group that were placed in the same installation, but at a different part of the plant that wasn’t hit by the fire. The remaining 18,825 modules were influenced by fire. Because

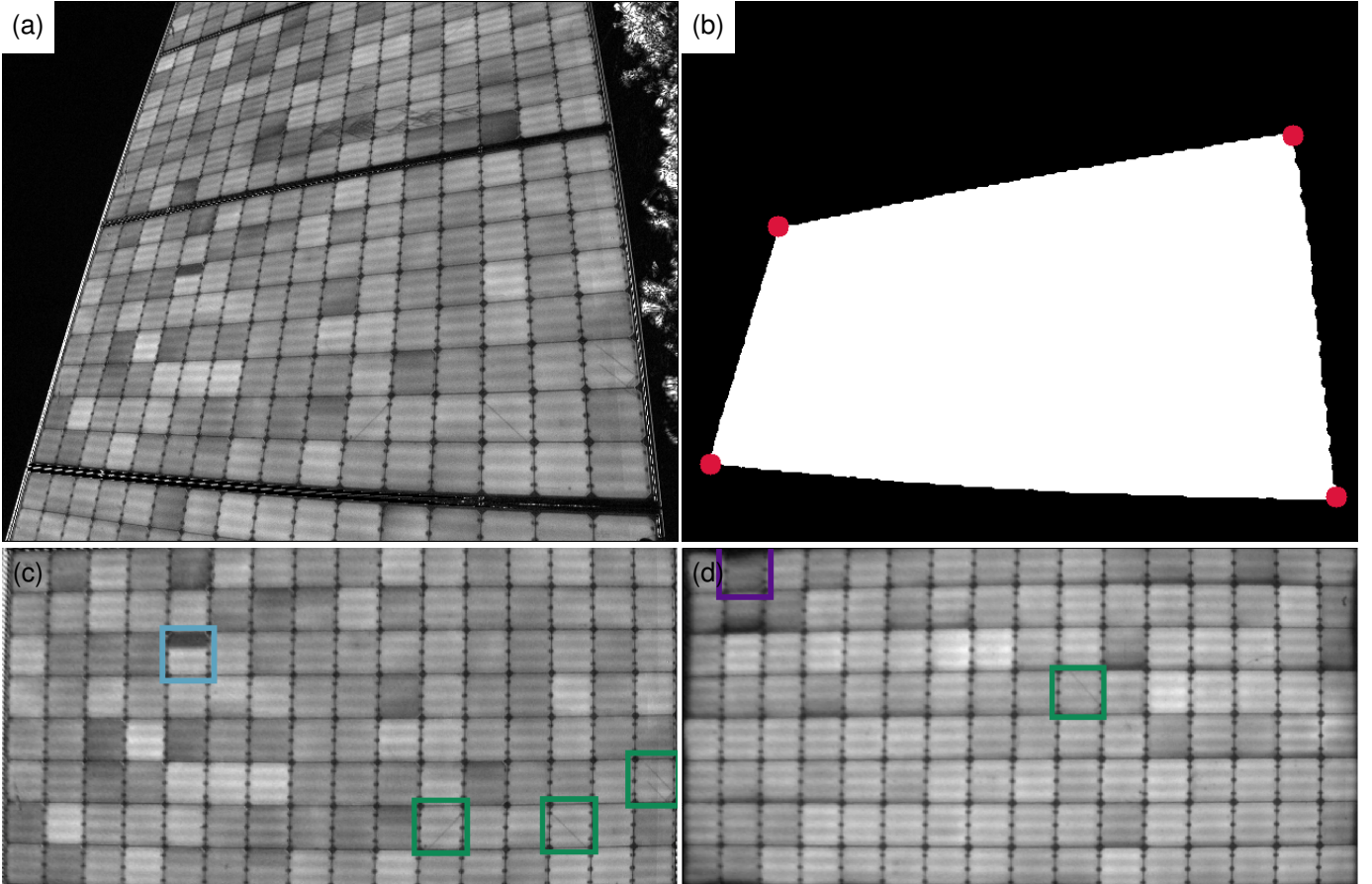


Fig. 2. (a) Example EL image of PV module. (b) Mask of the target module predicted by the UNet model and corners detected in the mask (c) Solar module after perspective transform. Bounding boxes identifying defects predicted by YOLO model are shown. (d) A different transformed solar module where the edge cells are dark.

we are interested in the effect of fire on this 1-in-portrait single-axis tracking system, we attempt to preserve the orientation of the processed images. Presumably, the entire array was tilted to one direction when the fire occurred, and was also tilted in a uniform direction when the EL survey was performed. Because the camera is mounted on a tripod, the short edge of the module that appears larger in the image will be the one closest to the ground during the EL survey. Most images, apart from 602 images, in our dataset have the longer vertical edge aligned on the right side in images. Therefore, we apply a 180 degree rotation on those images with an opposite direction to obtain a uniform orientation. After the re-orientation, we assume that the vertical edge on the right side in each image is closer to the ground.

III. METHODS

A. Automatic perspective transform

Figure 2(a) illustrates an example of a raw EL image from the dataset, with the target solar module in the center of the image. All raw EL images of solar modules were preprocessed by our open-source tools. The module background with vegetations, racks, and parallel modules makes it difficult to localize the target module and do perspective transformation automatically. In our pipeline, we first trained a UNet [32] model to do semantic segmentation on the raw image and determine the mask that covers the target module, as shown in Figure 2(b). 1692 labeled images of modules are sampled at random and fed into the UNet model. 181 additional images are sampled as the testing set and the Intersection over Union (IoU) score reached 99% (see Supporting Information for more details). The images processed by the UNet model are then processed to perform perspective correction and cell segmentation. More details of the training and testing of our UNet model for module detection are presented in the supplementary information (SI).

After the UNet model predicted module masks, we applied an algorithm combining Hough line detection [33] and corner detection [34] from OpenCV [35] to detect the four corners of the mask. The detected corners are shown in Fig. 2(b). Using the positions of the four corners, we apply a perspective transformation to generate the aligned image shown in Fig. 2(c). This transformation ignores the barrel distortion of the image. The size of the transformed image is 600×300 pixels to approximate the original size of the target module in raw images.

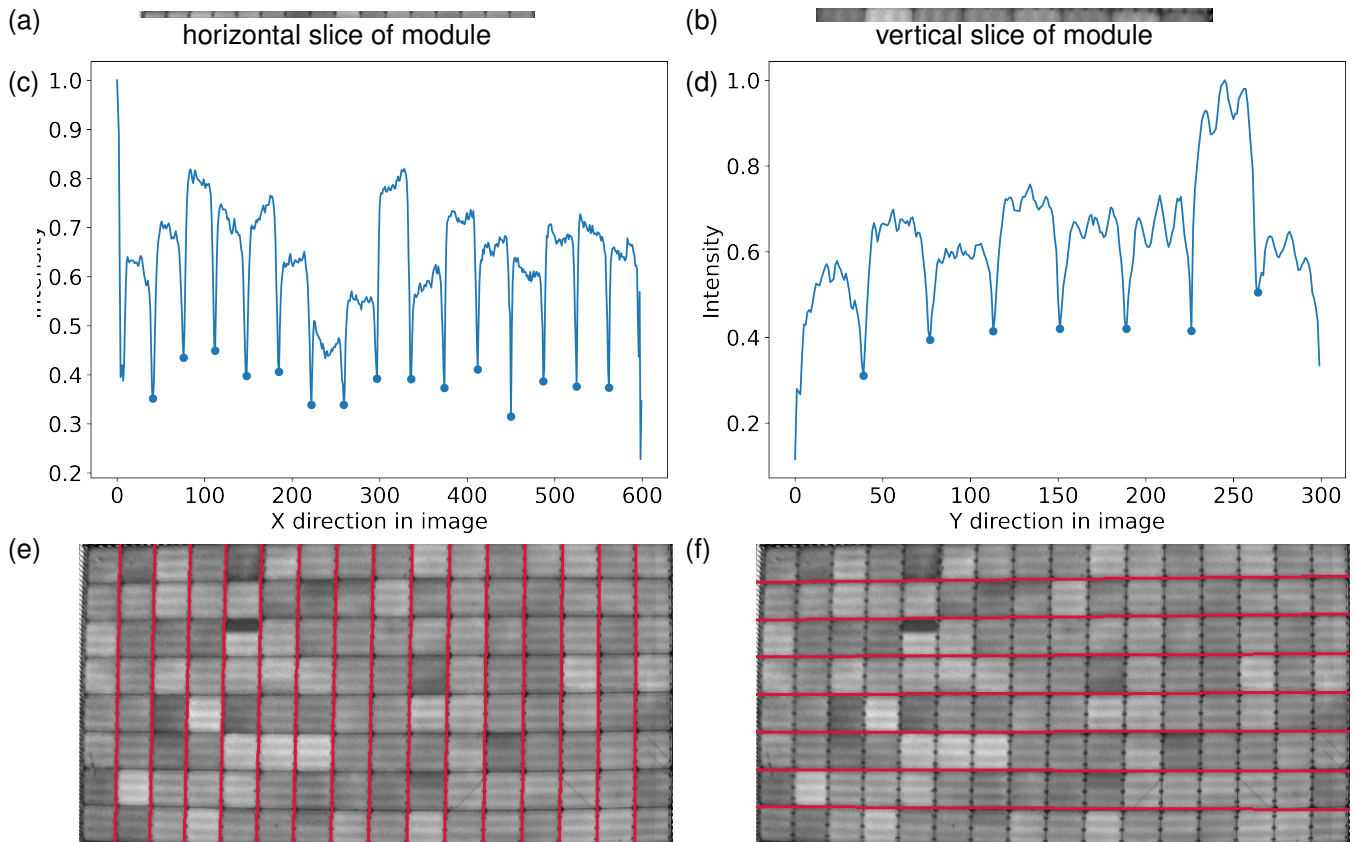


Fig. 3. Example of detecting the edges of single solar cells. (a)(b) Horizontal and vertical splits of solar modules. Split in (b) is rotated from vertical to horizontal position in favor of typesetting. (c)(d) Detected peaks on the positions of the edges. (e)(f) Fitted lines of inner edges on solar module.

B. Automatic cell segmentation

To train classifiers (see Method 2 in Figure 1), we need to crop out single cells from the perspective-transformed solar module. This is performed by splitting the images into different segments along the y-axis (*i.e.*, the vertical axis) with the size of each split equal to 600×10 pixels. An example of one split is shown in Figure 3(a). For each split, we sum the image in the vertical direction to get the distribution of gray-scale values shown in Fig. 3(c). The gray-scale intensity is normalized and ranges from 0 (approaching black) to 1 (approaching white). The edges of solar cells are the darkest and appear as dips in Fig. 3(c). We use ‘`signal.find_peaks`’ tool from Scipy [36] to find the positions of those dips. After we find the positions of edges of solar cells in each split, we fit those positions to compute a line that represents each edges, shown in Fig. 3(e). We perform an analogous procedure to find the splits in the horizontal direction, shown in Figure 3(b)(d)(f). Based on the intersection of these lines, we determine the four corners of each solar cell and crop them out (size of 32×32 pixels). The size is selected to maintain the cell’s original size.

C. Classification and object detection models

To develop the YOLO model and classifiers, 896 images of solar modules were sampled randomly and split into a training set (762 modules) and a validation set (134 modules). This data is used to evaluate several models and select two models for further study (a classification approach and an object detection approach). Following model selection, 129 additional images were sampled randomly for testing and performance evaluation. The ratio of train-val-test is about 7.5:1.5:1. There are five categories of solar cells labeled in the dataset: ‘intact’ cells, cells with ‘cracks’, cells with ‘oxygen’ induced defects (*i.e.*, striation rings), cells with ‘intra-cell’ defects and cell with ‘solder’ issues. The ‘intact’ cells are not labeled as objects to detect for YOLO model. The annotation is performed on Supervisely [37] by two annotators. The annotation results are cross verified by each annotator. For the YOLO model, the training data is augmented by vertical flip, horizontal flip, 180-degree rotation, and random crop. For random crop, the input image is cropped at random to 80% - 95% of its width and 90% - 95% of its height.

The same 1,025 images (train, validation, and test) of modules used to develop the YOLO model are used to train and evaluate the classifiers. The cell segmentation tool is used to cropped module images into single solar cells, creating 131,200 images of solar cells. Among them, 15 cells have two defects, *e.g.*, both ‘crack’ and ‘solder’ defects. We duplicated these cell images

and placed them into two categories. The training set for classifiers is augmented through vertical flip, horizontal flip, and 180-degree rotation. The distribution of each category in the dataset is summarized in Table I. Since categories are extremely imbalanced in the training set, we compare the performance of models with different sampling methods including undersampling and oversampling. We found that using the original training set is optimal for ResNet models while undersampling is optimal for the RF model. The details are shown in section ‘comparison of sampling methods’ of SI.

For annotation and training convenience, the YOLO v3 [15] (pretrained on COCO [38]) model deployed on Supervisely is selected to do transfer learning. The single channel in the grayscale image is duplicated and expanded to three channels so the size of the input images is $300 \times 600 \times 3$. The default size of the input layer of YOLO model is 416. The training batch size is 12. The model is trained for 10 epochs and the model at the epoch that has the optimal loss on the validation set is selected as the final model.

CNN models are constructed using Pytorch (version 1.6.0) [39]. ResNet models are pretrained on ImageNet [40] and fine-tuned during our training process. The input of the CNN model is a grayscale image of a segmented cells with size $32 \times 32 \times 3$ pixels (the single channel of the grayscale image is replicated 3 times to match the RGB format of the model). Cross-entropy loss is used as the loss function. Stochastic gradient descent (SGD) was used as the optimizer. The learning rate is tuned from 0.001 with learning rate scheduler. The batch size of training is 128. The images are trained for 20 epochs and the model at the epoch that has the optimal loss on the validation set is selected as the final model.

The RF model is constructed with Scikit-learn (version 0.23.2) [41]. As the input of RF, each grayscale image with the size of 32×32 is flattened. Each flattened array is concatenated into the input matrix. This flattening method was also used in a previous work [10]. Hyperparameter tuning is conducted to return the best model; details are in the SI. The optimal hyperparameters determined for the RF model are {‘number of trees’: 200, ‘split criterion’: ‘gini’, ‘maximum depth’: None, ‘bootstrap’: False}.

To compare the performance of each model, we used recall, precision, f-1 score, and f-2 score:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (1)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (2)$$

$$F_1 = \frac{\sum 2\text{TP}}{\sum 2\text{TP} + \text{FN} + \text{FP}} \quad (3)$$

$$F_\beta = (1 + \beta^2) \frac{\text{Precision} * \text{Recall}}{\beta^2 * \text{Precision} + \text{Recall}}, \beta = 2 \text{ for } F_2 \quad (4)$$

where TP are true-positives (model correctly identifies an existing defect), FP are false-positives (model detects a defect when none exists), and FN are false-negatives (model does not detect a defect when one exists).

Transformation, segmentation scripts and training of RF model are executed on MacBook Pro (2.8 GHz Quad-Core Intel Core i7 processor, 16 GB 2133 MHz LPDDR3 memory). UNet and YOLO models are trained on 4 NVIDIA TESLA K80 GPUs (48 GB GDDR5 memory) and deployed on 1 NVIDIA TESLA K80 GPU (12 GB GDDR5 memory). CNN classifiers are trained and deployed on 1 NVIDIA TESLA K80 GPU (12 GB GDDR5 memory).

IV. RESULTS AND DISCUSSION

A. Image preprocessing

Figure 2(a)-(c) illustrate that the UNet model successfully segments the target module even with multiple complicating image features. Figure 2(d) shows another example of a successfully transformed image where some cells have darkened edges. We apply the perspective correction algorithm to all 19,228 images of solar modules, finding 98.6% (18,954 images) are transformed successfully. Of the failed cases, around 68% are due to the wrong mask predicted by the model while the rest are due to poor images. Figure 4 shows examples of failed images. Most incorrectly predicted masks are similar Figure 4 (a). The orientation of this module is different from most of the images in the training set. Also the solar module was too tilted for the model to recognize properly. By including more images like this in the training set, the model performance can be improved. In Fig. 4(b), the corner of the solar module is too dark to recognize. The module image is truncated in 4(c). Another poorly shot image like 4(d) can also cause the transformation algorithm to fail. We consider Fig. 4(b)-(d) an data acquisition error while errors in Fig. 4(a) can be fixed in the future version of models.

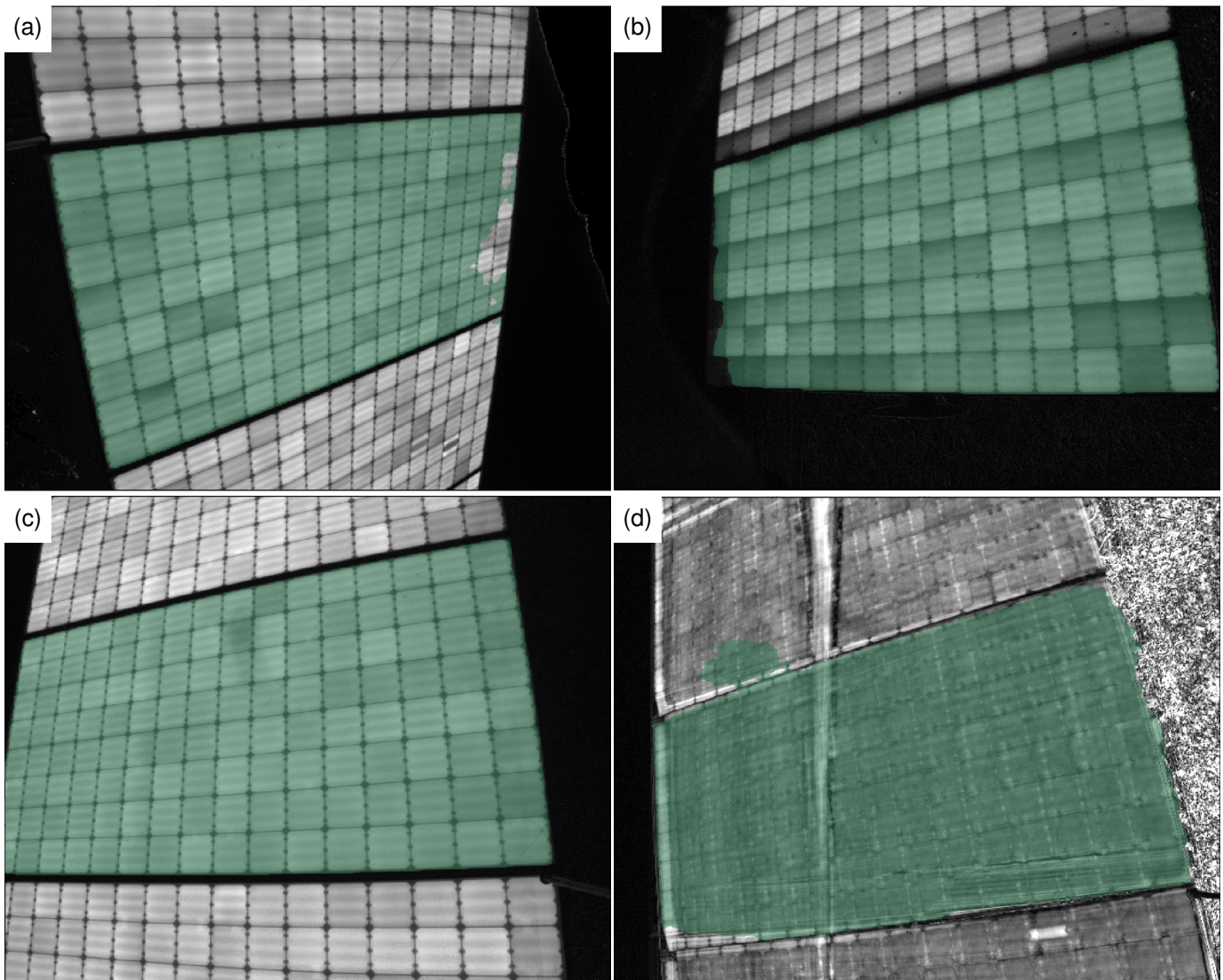


Fig. 4. Example of failed images. (a) Mask error due to orientation of the module. (b) Extremely dark edges that cannot be recognized from background. (c) Truncated module in the image. (d) Another poor photograph.

Model	YOLO	ResNet18	ResNet50	ResNet152	RF
Avg F1 (val)	0.86	0.87	0.87	0.87	0.57
Avg F1 (test)	0.78	0.83	Not tested		

TABLE II

MACRO AVERAGE(UNWEIGHTED MEAN) F1 SCORE OF EACH MODEL ON THE VALIDATION AND TESTING SETS

B. Cell classification and detection

The performance of each model on the validation set is illustrated in Figure 5 and Table II. The hyperparameter tuning and confusion matrices of each model are shown in the SI. Recall is the fraction of actual defects that were correctly labeled by the model, whereas precision is the fraction of predicted defects that are identified correctly. A desirable algorithm balance precision and recall so that most of the defects are found without too many spurious detections. One way to combine precision and recall is with the F1 score where the weights of each metric are the same. The F2 score combines precision and recall in a way that makes recall the more important score. Which metric is best depends on the demands of the users. For example, some applications need high sensitivity to detect all defective cells in order to determine which modules to replace. In this case, recall is more important and the user would select the algorithm with the best F2 score.

The macro average F1 score on validation set in Table II shows that the performance of ResNet models and YOLO model is similar. All these neural network models are better than RF model. The performance of the ResNet models isn't improved as the architecture of the model becomes more complex from ResNet18 to ResNet50 and ResNet152. Considering that deeper neural networks cost longer time to train and perform inference [42], ResNet18 is selected as the optimal model among classifiers.

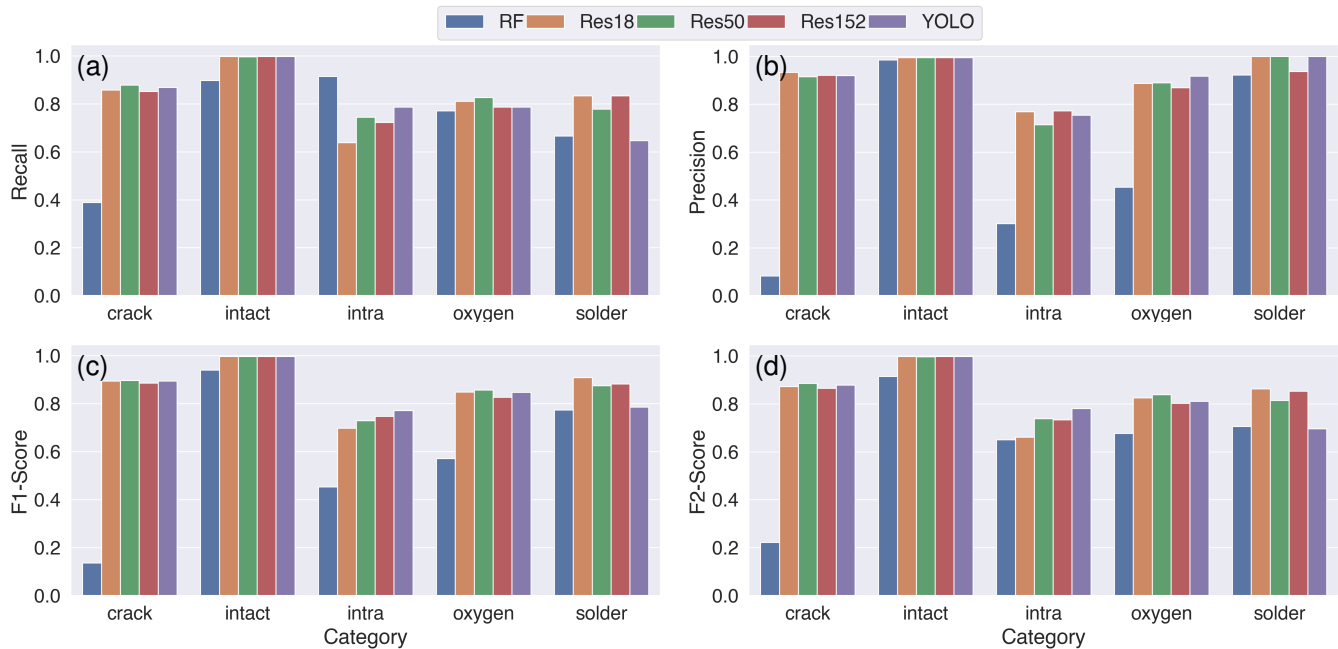


Fig. 5. Comparison of the performance of different models on the validation set. (a) Recall (b) Precision (c) F1 score (d) F2 score

YOLO model is also an optimal model since it is conceptually simpler. Although our models were trained on IBC cell images and are specific to this cell type, they can be applied to other types of solar cells using transfer learning. We demonstrated the transfer learning of ResNet18 to address other module types in SI.

When comparing the YOLO model versus the ResNet models, we note that the YOLO model doesn't require segmenting single solar cells, and the output bounding boxes (as shown in Figure 2c) of the YOLO model can directly show the positions of defective cells without further processing. However, the ResNet models may transfer better to other types of modules (*e.g.*, solar modules with 12×6 cells) because they take only cell images as inputs, not module images; whereas the ResNet model may only need adjustment to the cell segmentation procedure, the YOLO model may need retraining from scratch when it deals with other types of modules. The recall shows that YOLO model is more likely to find 'intra-cell' defects whereas ResNet18 is more likely to identify 'oxygen' induced defects. Therefore, the selection of models depends on which categories of defects the user is more interested in, and whether recall or precision is more important. We tested the performance of ResNet18 and YOLO model on the testing set. The macro average F1 scores are 0.83 (ResNet18) and 0.78 (YOLO), as shown in Table II.

Figure 6 shows the confusion matrix for the validation set of the best performing ResNet18 and YOLO models (confusion matrices for other models given in SI). Both models are somewhat confused between 'crack' and 'intact'. One possible reason is that some cracks are very small and more training examples are needed to distinguish these from intact cells. Also, the models may mistake some debris on the module surface as cracks. Table III illustrates some examples of the incorrectly predicted images from the ResNet18 and YOLO models. The first column of images in Table III have short cracks on the corner of the solar cell but the models did not recognize this kind of short cracks. The first two images in the second column of Table III have objects which are mistaken as cracks. We assume these objects are wider than normal cracks so they are more like debris on solar modules rather than cracks. The remaining images in the second column of Table III have a horizontal crack which is not obvious to see, so the annotators did not recognize it but the model detected it. The images in the third column have inconspicuous round striation which were not recognized by the models. Images of the fourth column for both model are confusing even to annotators. Those cells are darker than their neighbour cells and all their surface are evenly dark, which is different from the intra-cell defects in Table I. This charge trapping may not be necessarily caused by intra-cell defects generated in fire. They can also be caused by other degradation modes before exposure to high temperature.

C. Defects distribution on the solar modules

We apply the YOLO model to all 18,954 transformed EL images (2.41 million cells) and enumerate the defects at different positions in the 18,825 solar module that were influenced by fire and the 129 modules in control group. The occurrence of defects at particular locations in fire-influenced modules are shown in Figure 7. Figure 8 shows the significant difference of the proportion of defects in the control group and fire-influenced group. The value of proportion and corresponding p value is shown in SI. To reduce the influence of incorrectly predicted defects, we also use the combination of validation and testing

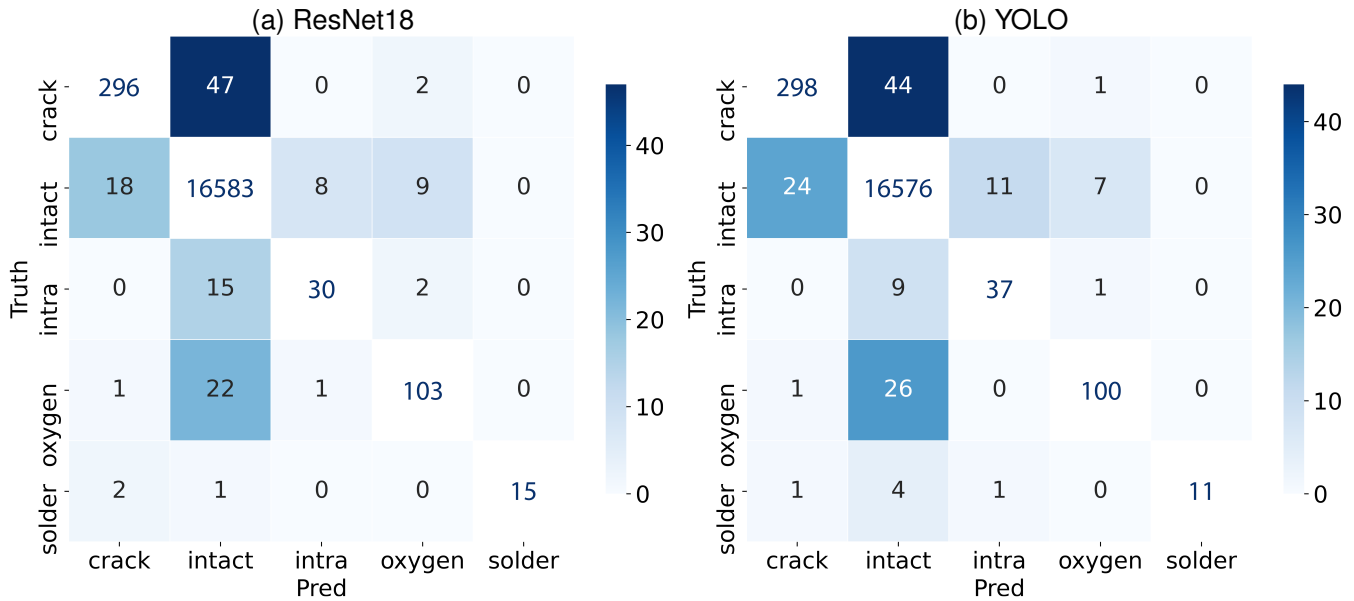


Fig. 6. Confusion matrix of (a) ResNet18 and (b) YOLO models tested on validation set. Colors of boxes on the diagonal of confusion matrix are removed for better visualization of incorrectly predicted labels.

Cells (YOLO)				
True label	Crack	Intact	Oxygen	Intact
Prediction	Intact	Crack	Intact	Intra-cell
Cells (Res18)				
True label	Crack	Intact	Oxygen	Intra-cell
Prediction	Intact	Crack	Intact	Intact

TABLE III
EXAMPLES OF INCORRECTLY PREDICTED CELLS FROM YOLO AND RESNET18 MODEL

set to estimate the precision of the predictions and subtract the estimated false positive in the predicted defects. However the distribution is still similar to Figure 7. Details are presented in the SI.

Figure 7(b) shows that the ‘oxygen’ defect is (to a first approximation) distributed uniformly on the solar module whereas Figure 7(a)(c)(d) show that the ‘crack’, ‘solder’ and ‘intra’ defects tend to occur on the two shorter edges of the solar module, especially the right edge which should be closer to the ground. One hypothesis is that the shorter edges are closer to the fire source when the solar modules are tilted at a certain angle in the field. It is confirmed by the data provider PVEL that shorter edges are mounted to the ground, although which side (left or right) is always closer to the fire is based on our analysis of the perspective of the image. We have attempted to rotate images such that the right edge corresponds to the side closer to the ground. Based on this analysis, the fire may accelerate the formation and propagation of ‘crack’, ‘solder’, and ‘intra’ defects. Also, the high occurrence of defects on two short edges may be caused during installation or mounting, since these two edges are more susceptible to mechanical stress. Although the ‘oxygen’ defects distribute evenly on the edges and inner part of modules, they still show a tendency of distributing on the right side and Figure 8 also show the significant increase of ‘oxygen’ defects after fire. This is beyond our expectation since ‘oxygen’ defects are formed during manufacturing [31] and are not expected to be impacted by fire. This result is worthy of further investigation. The distribution of defects implies there

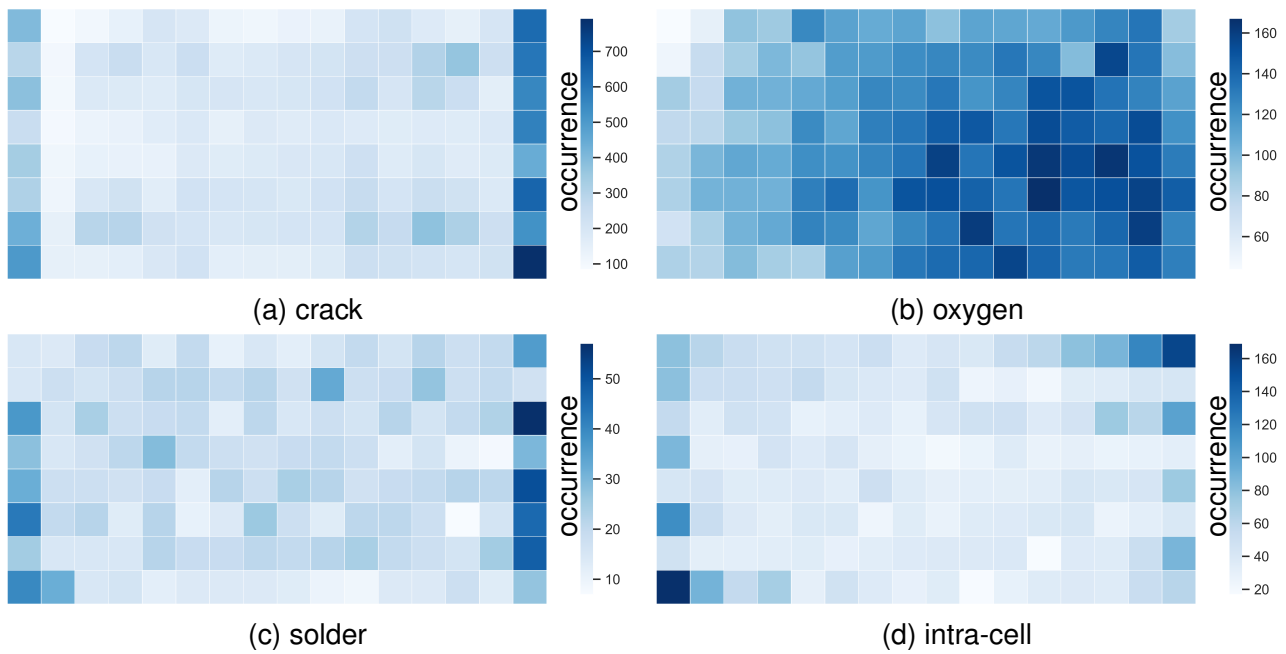


Fig. 7. Distribution of defects on PV modules affected by fire. Here each heatmap shows the quantity of defects observed in each cell in a 16×8 solar module. Defects are recognized by YOLO model. The right-hand side of the image is the side of the module closest to the ground during the EL survey. Defects shown are (a) Crack, (b) Oxygen induced defects, (c) Solder disconnection and (d) Intra-cell defects

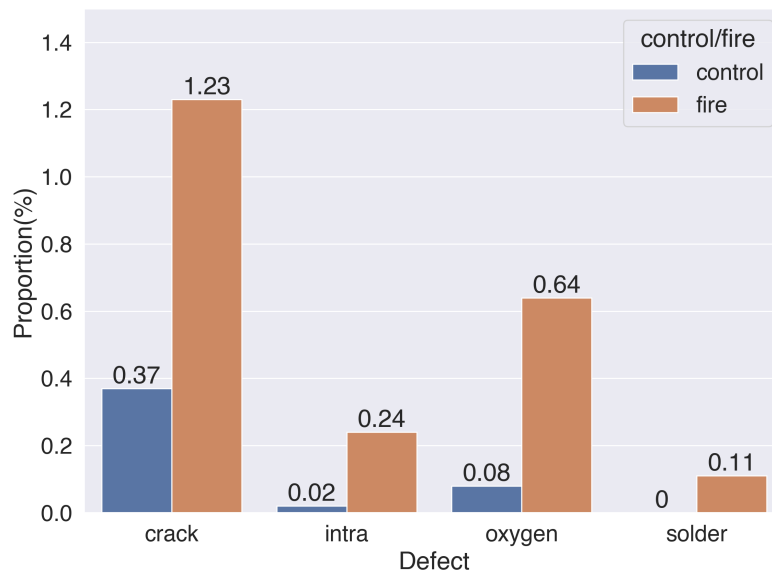


Fig. 8. Proportion of defects in fire-influenced group versus the control group.

is a possibility to improve quality by more carefully handling modules during installation, especially on the edges.

V. CONCLUSION AND OUTLOOK

This paper introduces an automatic pipeline for detecting defective cells in EL images of solar modules. The tool performs a perspective transformation of the tilted solar module and either performs direct object detection of defects or crops out single cells for further classification into defect categories. We train different machine learning models (classification and object detection) on the sampled images and compare their performance based on the imbalanced real-world data, concluding that the YOLO model used for object detection and the ResNet18 model used for classification are optimal models. Both models perform similarly well, with macro F1 scores on a real-world testing set of 0.83 (ResNet18) and 0.78 (YOLO). We also apply the YOLO model to 18,825 solar modules affected by fire and analyze the distribution of defects on the solar modules.

Future work could generalize the model by feeding in more EL images from other modules. We can further improve the performance of the models by improving the quality of our annotations. Also our tools can be used for other kinds of defects with minor modification on models by transfer learning. Comparisons of power loss data from current-voltage (IV) curve measurements can be correlated to the observed defects to determine the impact of defect quantity on power loss. The open-source tools and model weights are published on Github [29].

ACKNOWLEDGMENT

The authors would like to thank Jenya Meydbray and Beryl Weinshenker for kindly providing EL images for this paper and offering insights on the analysis of the results. We also would like to thank technicians from PVEL for their efforts in early annotations of EL images.

REFERENCES

- [1] U. Jahn, M. Herz, M. Köntges, D. Parlevliet, M. Paggi, I. Tsanakas, J. Stein, K. Berger, S. Ranta, R. French, M. Richter, T. Tanahashi, E. Ndrío *et al.*, *Review on Infrared (IR) and Electroluminescence (EL) Imaging for Photovoltaic Field Applications*. IEA International Energy Agency, 2018.
- [2] K. Bedrich, M. Bokalic, M. Bliss, M. Topic, T. R. Betts, and R. Gottschalg, "Electroluminescence Imaging of PV Devices: Advanced Vignetting Calibration," *IEEE Journal of Photovoltaics*, vol. 8, no. 5, pp. 1297–1304, sep 2018.
- [3] T. Fuyuki, H. Kondo, T. Yamazaki, Y. Takahashi, and Y. Uraoka, "Photographic surveying of minority carrier diffusion length in polycrystalline silicon solar cells by electroluminescence," *Applied Physics Letters*, vol. 86, no. 26, pp. 1–3, jun 2005. [Online]. Available: <http://aip.scitation.org/doi/10.1063/1.1978979>
- [4] M. Abdelhamid, R. Singh, and M. Omar, "Review of microcrack detection techniques for silicon solar cells," *IEEE Journal of Photovoltaics*, vol. 4, no. 1, pp. 514–524, 2014.
- [5] K. Bedrich. Quantified energy labs. [Online]. Available: <https://qe-labs.com/>
- [6] Y. Yan, R. Rosales, G. Fung, M. Schmidt, G. Hermosillo, L. Bogoni, L. Moy, and J. Dy, "Modeling annotator expertise: Learning when everybody knows a bit of something," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 932–939.
- [7] D.-M. Tsai, S.-C. Wu, and W.-Y. Chiu, "Defect detection in solar modules using ica basis images," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 122–131, 2012.
- [8] S. A. Anwar and M. Z. Abdullah, "Micro-crack detection of multicrystalline solar cells featuring an improved anisotropic diffusion filter and image segmentation technique," *EURASIP Journal on Image and Video Processing*, vol. 2014, no. 1, pp. 1–17, 2014.
- [9] J. S. Fada, M. A. Hossain, J. L. Braid, S. Yang, T. J. Peshek, and R. H. French, "Electroluminescent image processing and cell degradation type classification via computer vision and statistical learning methodologies," in *2017 IEEE 44th Photovoltaic Specialist Conference (PVSC)*. IEEE, 2017, pp. 3456–3461.
- [10] A. M. Karimi, J. S. Fada, M. A. Hossain, S. Yang, T. J. Peshek, J. L. Braid, and R. H. French, "Automated pipeline for photovoltaic module electroluminescence image processing and degradation feature classification," *IEEE Journal of Photovoltaics*, vol. 9, no. 5, pp. 1324–1335, 2019.
- [11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.
- [12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
- [13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-December. IEEE Computer Society, dec 2016, pp. 770–778. [Online]. Available: <http://image-net.org/challenges/LSVRC/2015/>
- [14] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.
- [15] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv*, apr 2018. [Online]. Available: <http://arxiv.org/abs/1804.02767>
- [16] S. Deitsch, V. Christlein, S. Berger, C. Buerhop-Lutz, A. Maier, F. Gallwitz, and C. Riess, "Automatic classification of defective photovoltaic module cells in electroluminescence images," *Solar Energy*, vol. 185, pp. 455–468, 2019.
- [17] C. Buerhop-Lutz, S. Deitsch, A. Maier, F. Gallwitz, S. Berger, B. Doll, J. Hauch, C. Camus, and C. J. Brabec, "A benchmark for visual identification of defective solar cells in electroluminescence imagery," in *35th European PV Solar Energy Conference and Exhibition*, vol. 12871289, 2018.
- [18] S. Deitsch, C. Buerhop-Lutz, E. Sovetkin, A. Steland, A. Maier, F. Gallwitz, and C. Riess, "Segmentation of photovoltaic module cells in uncalibrated electroluminescence images," *Machine Vision and Applications*, vol. 32, no. 4, pp. 1–23, 2021.
- [19] M. W. Akram, G. Li, Y. Jin, X. Chen, C. Zhu, X. Zhao, A. Khaliq, M. Faheem, and A. Ahmad, "Cnn based automatic detection of photovoltaic cell defects in electroluminescence images," *Energy*, vol. 189, p. 116319, 2019.
- [20] A. M. Karimi, J. S. Fada, J. Liu, J. L. Braid, M. Koyutürk, and R. H. French, "Feature extraction, supervised and unsupervised machine learning classification of pv cell electroluminescence images," in *2018 IEEE 7th World Conference on Photovoltaic Energy Conversion (WCPEC)(A Joint Conference of 45th IEEE PVSC, 28th PVSEC & 34th EU PVSEC)*. IEEE, 2018, pp. 0418–0424.
- [21] W. Tang, Q. Yang, K. Xiong, and W. Yan, "Deep learning based automatic defect identification of photovoltaic module using electroluminescence images," *Solar Energy*, vol. 201, pp. 453–460, 2020.
- [22] X. Zhang, Y. Hao, H. Shangguan, P. Zhang, and A. Wang, "Detection of surface defects on solar cells by fusing multi-channel convolution neural networks," *Infrared Physics & Technology*, vol. 108, p. 103334, 2020.
- [23] Y. Zhao, K. Zhan, Z. Wang, and W. Shen, "Deep learning-based automatic detection of multitype defects in photovoltaic modules and application in real production line," *Progress in Photovoltaics: Research and Applications*, vol. 29, no. 4, pp. 471–484, 2021.
- [24] Z. Meng, S. Xu, L. Wang, Y. Gong, X. Zhang, and Y. Zhao, "Defect object detection algorithm for electroluminescence image defects of photovoltaic modules based on deep learning," *Energy Science & Engineering*, vol. 10, no. 3, pp. 800–813, 2022.
- [25] M. Mayr, M. Hoffmann, A. Maier, and V. Christlein, "Weakly supervised segmentation of cracks on solar cells using normalized l_p norm," in *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019, pp. 1885–1889.
- [26] J. Fiorelli, D. J. Colvin, R. Frola, R. Gupta, M. Li, H. P. Seigneur, S. Vyas, S. Oliveira, M. Shah, and K. O. Davis, "Automated defect detection and localization in photovoltaic cells using semantic segmentation of electroluminescence images," *IEEE Journal of Photovoltaics*, vol. 12, no. 1, pp. 53–61, 2021.
- [27] J. Meydbray. Pv evolution labs. [Online]. Available: <https://www.pvel.com/>
- [28] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, <http://www.deeplearningbook.org>.
- [29] X. Chen. github: pv-vision. [Online]. Available: <https://github.com/hackingmaterials/pv-vision>
- [30] Daylight system luminescence. [Online]. Available: <https://www.solarzentrum-stuttgart.com/en/products/daysyl/>
- [31] M. Köntges, S. Kurtz, C. Packard, U. Jahn, K. A. Berger, K. Kato, T. Friesen, H. Liu, M. Van Iseghem, J. Wohlgemuth *et al.*, *Review of failures of photovoltaic modules*. IEA International Energy Agency, 2014.
- [32] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
- [33] Hough line transform. [Online]. Available: https://docs.opencv.org/4.5.3/d9/db0/tutorial_hough_lines.html
- [34] Shi-tomasi corner detector and good features to track. [Online]. Available: https://docs.opencv.org/4.5.3/d4/d8c/tutorial_py_shi_tomasi.html
- [35] G. Bradski, "The OpenCV Library," *Dr. Dobbs Journal of Software Tools*, 2000.
- [36] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, I. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.
- [37] Supervisely. [Online]. Available: <https://supervisely/>
- [38] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2015.

- [39] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “PyTorch: An Imperative Style, High-Performance Deep Learning Library,” in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>
- [40] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, “ImageNet Large Scale Visual Recognition Challenge,” *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.
- [41] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, and Others, “Scikit-learn: Machine learning in Python,” *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.
- [42] J. Johnson. github: cnn-benchmarks. [Online]. Available: <https://github.com/jcjohnson/cnn-benchmarks>