PSOPruner: PSO-Based Deep Convolutional Neural Network Pruning Method for PV Module Defects Classification

Chao Huang , *Member, IEEE*, Zhiying Zhang, and Long Wang, *Member, IEEE*

Abstract—Photovoltaic (PV) modules can be damaged by external environment, which can significantly reduce power generation efficiency. Electroluminescent (EL) imaging is an economical and widely applied technique for PV module defects detection. The analysis of EL images, however, is still labor-consuming. To solve this problem, a deep convolutional neural network (DCNN) is proposed to automatically detect and classify defects. The proposed DCNN performs well on a large public EL dataset with more than 2600 images extracted from both monocrystalline and polycrystalline PV modules. Nevertheless, the proposed DCNN cannot be directly applied on portable or embedded devices due to its requirement of large-size memory and intensive computation. To handle this challenge, an evolutionary algorithm-based DCNN pruning method, dubbed PSOPruner, is further developed. The pruning problem of DCNN is formulated as a search problem, which is solved by particle swarm optimization (PSO) algorithm. To improve the quality of the pruning scheme, a tailored trick is considered that the automatic searching process with PSO algorithm is repeated for multiple rounds. To illustrate the effectiveness of the proposed PSOPruner, we compare it with mainstream DCNNs and lightweight CNNs in terms of model complexity and model accuracy. Experimental results demonstrate that the proposed method could efficiently reduce the amount of model parameters with slight drop of accuracy.

Index Terms—Deep neural network, defect classification, electroluminescence (EL) imaging, filter pruning, particle swarm optimization (PSO), photovoltaic (PV) module.

I. INTRODUCTION

THE increasing concern on decarbonization and the depletion of fossil fuels have driven the transition from conventional energy to renewable energy, such as solar energy [1], [2].

The annual worldwide photovoltaic (PV) module production has greatly increased from 20 GW in 2010 to more than 140 GW in 2020 [3]. However, PV modules can be damaged by inappropriate manufacturing process and deployment environment [4], [5]. These defects, such as microcracks, broken cells, and finger interruptions will reduce photoelectric conversion efficiency. Hence, the detection of defects is very important for PV system maintenance [6], [7].

Visual recognition of PV module defects by professionals is labor-consuming [8]–[10]. While, these defects are visible in electroluminescence (EL) images [11] that computer vision-based techniques can be used for automatic defects detection [9], [12]–[17]. Akram et al. [14] proposed an automatic defect classification framework with machine learning technologies including support vector machine (SVM), random forest, and decision tree. Deitsch et al. [9] compared handcrafted feature-based SVM with an end-to-end convolutional neural network (CNN) for PV defects detection and the CNN outperformed the aforementioned SVM. In work [16], the authors trained a segmentation model based on U-Net [18] to analyze various defects in EL images.

The aforementioned studies are mostly based on deep CNNs (DCNNs), such as VGG [19], ResNet [20], etc. These networks are usually composed of hundreds of convolution filters and full connection layers, which require large-size memory and intensive computation. Such characteristics of DCNNs make them unsuitable for mobile, portable, and embedded devices. To extend the practical application of DCNN-based automatic PV defects detection methods, a particle swarm optimization (PSO)-based DCNN pruning method, dubbed PSOPruner, is proposed. The proposed pruning scheme will greatly reduce the requirement on storage and accelerate the network, while maintaining its classification accuracy. The main contributions of this article are as follows.

1) Common defects in EL images and causes are discussed, which is expected to be helpful for the follow-up automatic classification task.
2) A ResNet18-based DCNN is developed on the public dataset in [9] for automatic defects detection of PV modules in EL images. This DCNN is denoted as initial model for PSO-based pruning process.
3) Filter pruning-based approach is considered to compress the DCNN. The filter pruning problem is formulated as an optimization problem, which is solved by particle swarm
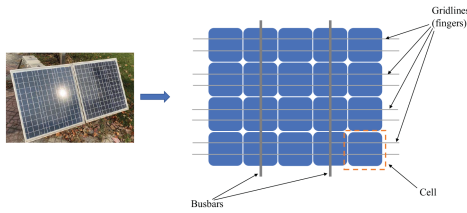
Fig. 1. Composition of PV modules.

optimization (PSO) algorithm. The proposed method provides promising pruning scheme without human intervention.

4) A tailored trick is considered for the application of PSO-Pruner that the automatic searching process is repeated for multiple rounds to refine the pruning scheme. To illustrate the effectiveness of the proposed method, we compare it with mainstream CNNs including lightweight CNNs in terms of complexity and accuracy. Experiment results demonstrate that the PSOPruner can greatly reduce the amount of model parameters with slight drop of accuracy.

The rest of this article is organized as follows. In Section II, related work on automatic PV defects detection and classification with EL images will be introduced. The development of the proposed PSOPruner will be illustrated in Section III, which is followed by experiments and results analysis in Section IV. Finally, Section V concludes this article.

II. RELATED WORK

In this section, we will introduce the principle of EL test in a well-controlled environment inside a laboratory, discuss common PV module defects, and survey automatic defects detection methods.

A. EL Test

Both polycrystalline silicon modules and monocrystalline silicon modules are composed of cells connected in series and in parallel, as shown in Fig. 1.

EL test is a recognized and effective method to detect PV module defects [21], [22]. The test principle shown in Fig. 2 is as follows. A forward bias voltage is applied to cells in a darkroom that the cell is filled with unbalanced carriers. These carriers continuously compound and emit photons [21]. The charge coupled device (CCD) camera with high sensitivity and low noise can capture these photons. The color of the normally working cell is bright and the image is clean, as shown in Fig. 3. On the contrary, the current flow will be blocked and the imaging color will be relatively dark for PV modules with defects, as shown in Fig. 4. The EL images in Figs. 3 and 4(a)–(k) are from the public dataset in [9]. It should be noted that there are multiple shadows in the EL image of functional polycrystalline silicon modules due to crystal dislocation.

B. Defects Classification and Causes Analysis

According to previous studies [23], [24], we illustrate five well-known types of PV module defects in Fig. 4. For each
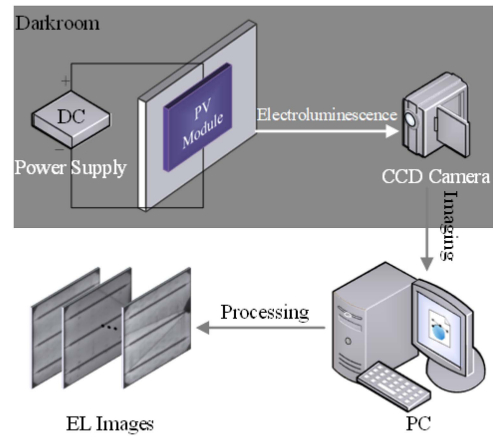

Fig. 2. EL imaging principle [14].

Monocrystalline PV Module - Functional
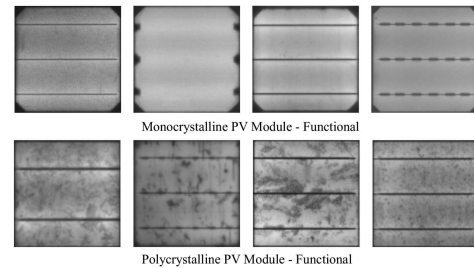
Polycrystalline PV Module - Functional

Fig. 3. EL images for functional cells of mono- and polycrystalline PV modules.

type of defect, their causes and impacts on PV efficiency can be different.

Finger Interruption:

1) Finger interruption shown in Fig. 4(a) happens often in the manufacturing process of screen-printed silicon cells. Fortunately, this kind of finger interruption does not necessarily cause a decrease in cell power generation efficiency.

2) Finger interruption shown in Fig. 4(b) is caused by cell cracks, and the distribution of such kind of finger interruption tends to follow cracks.

3) Finger interruption in Fig. 4(c) is a typical defect caused by improper soldering with high strain at the solder joint.

Cracks:

1) Production is one of the major sources of cell cracks. The cracks seen in Fig. 4(d) starting from the busbars are caused by the residual stress in the soldering process. This kind of crack leads to the fracture of gridlines and affects the collection of current.

2) Another source of crack is the transportation and installation. Drops or collisions will cause malfunctions to cells, which is mainly crack. External force collision can cause cracks starting from the edge of cells, as shown in Fig. 4(e).

3) Fig. 4(f) indicates serious mechanical damage that breaks the cell and leads to large scale black areas.

Contact Forming:

1) The defect in Fig. 4(g) presents a tire-like imprint. It is caused by uneven temperature during the firing process
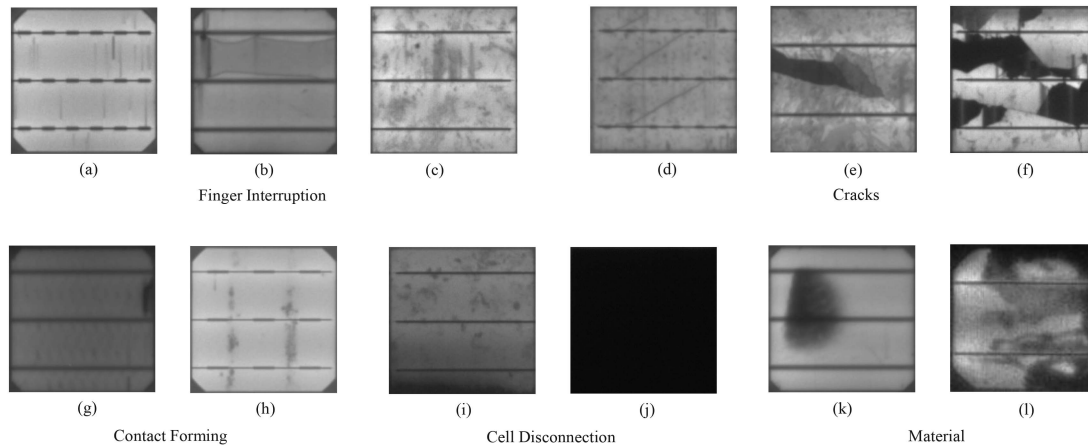
Fig. 4.   Different defect types on EL images of PV modules.

of transport belt. It may have little impact on the power generation efficiency.

2) The thick black lines perpendicular to the busbars in Fig. 4(h) are associated with very hot parts of modules during solder bonding. This defect will lead to open-circuit failure between busbars and gridlines.

Cell Disconnection:

1) The black areas at the bottom of the cell in Fig. 4(i) indicate the degradation in cell connection.

2) Fig. 4(j) illustrates a black EL image due to short-circuit failure caused by cell disconnection. Such cells hardly generate electricity.

Material:

1) The black area in Fig. 4(k) mostly comes from the defects of silicon material, such as low nonequilibrium minority carrier concentration.

2) High humidity leads to corrosion of silicon material, which seriously reduces power generation efficiency. Irregular shadows in Fig. 4(l) [22] indicate typical corrosion defects.

C. Deep Learning-Based Automatic Classification of PV Defects in EL Images

Two models, i.e., SVM and CNN, were applied for automatic classification of PV defects in [9]. The SVM was developed on handcrafted features from EL images, while image pixel values were directly fed into VGG19-based CNN model. The experimental results showed that the VGG19-based CNN was much more precise than the SVM for PV defects classification.

To obtain a larger dataset for model training, thereby to obtain a better classification model, data augmentation was used by Akram et al. [14] to extend the dataset in [9]. To further extend the dataset, cropping was deployed in [8] that the EL images featuring 300×300 pixels were cropped into several images featuring 100×100 pixels. The authors tested the performance of VGG-based deep structure by varying the depth and analyzed the impacts of batch normalization and dropout on model performance.

In [13], a deep feature-based method is proposed that image features extracted by deep neural network are fed into machine learning models including SVM, k-nearest neighbor, random forest, and naïve Bayes for defects classification. To improve classification performance, the minimum redundancy maximum relevance algorithm [25] was applied for feature selection. The authors also compared the proposed deep feature-based method against a linear lightweight CNN to illustrate its advantage. Both supervised learning and unsupervised learning were used by Karimi et al. [26]. With supervised learning, i.e., CNN and SVM, PV cells were classified into degraded and nondegraded cells with greater than 98% accuracy. With unsupervised learning, i.e., hierarchical clustering algorithm, the PV cells were classified into two clusters with 66% coherence.

In [27], the authors pointed out that most existing automatic detection methods lack defective location on PV modules. They proposed an end-to-end pipeline combining detection [28], classification [29], and weakly supervised segmentation to detect, locate, and segment defects in EL images. A postprocessing stage is attached at the end of the pipeline, which marked the defective and functional cells on a complete PV module for further maintenance.

In [30], the authors proposed a weakly supervised learning strategy to segment cracks on cells using image-level annotations in which defect classification was used to train a deep network, i.e., ResNet50. The activation maps from the deep network were considered to be a guide of abnormal regions. However, the segmentation results were coarse due to the limited annotation information.

There are many works on automatic detection of PV module/cell defects; nevertheless, they all lack the evaluation of network complexity. The AlexNet, for example, requires 290-MB SD card storage and over 300 MB RAM storage in Android format [31], while the VGG16 network requires more than 500 MB of storage in 32-bit floating-point format. These bulky and computation-intensive models cannot be carried out on resource-limited embedded or mobile devices for practical scenarios. To make deep learning models applicable on embedded devices, Tsai et al. [32] compressed the network by reducing
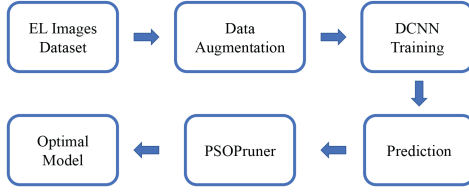
Fig. 5. Flowchart of the proposed method.

Fig. 6. Filter pruning for convolution layer.

the number of filters, layers, and the size of the convolution kernel. The problem of DCNNs in complexity also hinders their application in PV-related image processing tasks, e.g., unmanned aerial vehicle-based on-line PV faults detection [33], [34]. In [35], EL images of filed PV modules were used for the development of deep learning models to identify PV defects. However, it is difficult to apply DNNs for filed faults detection on portable devices. Dealing with this challenge, we train an extremely lightweight CNN model while maintaining high accuracy in defective classification by an evolutionary algorithm-based pruning method. To the best of our knowledge, it is the first time to consider the practical application problem in automatic PV defects detection and give the well-directed solution.

III. PROPOSED PSOPRUNER

In this section, the principles of filter pruning will first be illustrated and then the determination of best pruning scheme will be formulated as an optimization problem. Finally, the proposed PSOPruner will be introduced to solve the optimization problem. The flowchart of the proposed method is shown in Fig. 5.

A. Filter Pruning

Filter pruning is an effective network compression method [36]–[38], which also helps alleviate overfitting for small datasets. The convolutional layers in CNNs perform multiple accumulation operations, which is time-consuming. Pruning convolutional filters can speed up the network and reduce storage requirement [38]. The saving of computational cost by pruning is discussed following [24]. Let L_i denotes the ith convolutional layer, M_i denotes the input feature of L_i, and M_{i+1} denote the output feature of L_i, as illustrated in Fig. 6. The output feature map M_{i+1} of L_i will serve as the input feature map of L_{i+1}. The dimensions of input and output feature maps are $c_i \times H_i \times W_i$ and $c_{i+1} \times H_{i+1} \times W_{i+1}$, respectively. The K_{ij} denotes the jth filter of L_i with size of $k \times k \times c_i$. The amount of convolutional operation on L_i is defined as amount_i, calculated by

$$\text{amount}_i = c_i \times c_{i+1} \times H_{i+1} \times W_{i+1} \times k \times k \quad (1)$$

If the K_{ij} is pruned, the memory for storing parameters of K_{ij} will be saved and the reduced amount of convolutional operation on L_i expressed as reduce_{ij}

$$\text{reduce}_{ij} = c_i \times H_{i+1} \times W_{i+1} \times k \times k \quad (2)$$

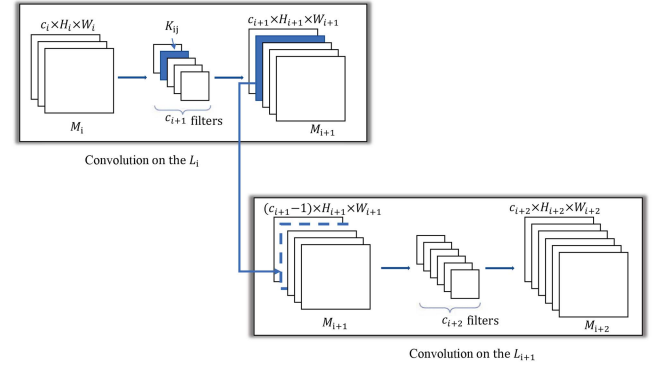Filter pruning also reduces convolutional operation for the following layer as the input of L_{i+1} no longer carries the feature map generated by K_{ij}. The reduced amount of convolutional operation on L_{i+1} expressed as reduce_{i+1j}

$$\text{reduce}_{i+1j} = c_{i+2} \times H_{i+2} \times W_{i+2} \times k \times k. \quad (3)$$

Based on the above analysis, it is clear that filter pruning will reduce both storage and convolutional operations. However, the selection of filters to be pruned is a challenge. In this article, the sum of absolute values of filter parameters (weights) denoted by sum_{ij} is used to represent the importance of the filter K_{ij}. It is meaningful in semantic that a smaller value will lead to less information transmitted by the filter. Hence, filter pruning for a convolutional layer is to prune the filters with the smallest sum of absolute values of filter parameters. Let r_i denote the pruning rate for L_i, which defines the rate of the number of filters to be pruned with respect to the number of filters in L_i. Considering a network $N = (L_1, L_2, \ldots, L_i, \ldots, L_n)$ with n convolutional layers, the pruning scheme can be expressed as a combination of r_i

$$R = (r_1, r_2, \ldots, r_i, \ldots, r_n), r_i \in (0, 1). \quad (4)$$

The filter pruning process on N is as follows.
1) Obtain the sum of absolute weights of all the filters on L_i, denoted as $\text{sum}_{i1}, \ldots, \text{sum}_{ij}, \ldots, \text{sum}_{ic_{i+1}}$.
2) Sort all the filters by the values of $\text{sum}_{i1}, \ldots, \text{sum}_{ij}, \ldots, \text{sum}_{ic_{i+1}}$.
3) Prune $c_{i+1} \times r_i$ filters with the smallest value according to step 2.
4) Repeat the above step 1–3 on all convolution layers $L_1, L_2, \ldots, L_i, \ldots, L_n$ of N.
5) Update N and copy the remain filters to the pruned model N', expressed as $N' = (L_1,' L_2,' \ldots, L_i,' \ldots, L_n')$, retrain N'.

B. Optimization Problem

The objective of filter pruning is to find an optimal pruning scheme. The scheme minimizes model complexity, while maintaining model classification accuracy. In this article, the amount of model parameters, which highly depends on the pruning rate, is used to represent model complexity. Given a network $N = (L_1, L_2, \ldots, L_i, \ldots, L_n)$, training dataset D_{train}, test dataset D_{test}, the optimal pruning scheme is achieved by
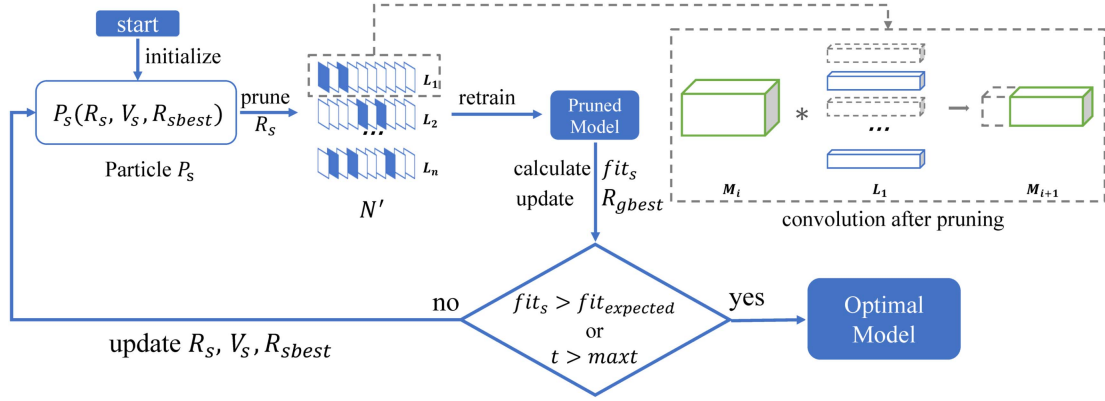
Fig. 7. Iterations process in PSOPruner.

maximizing the objective function

$$f(R) = \omega \times f_1\left(N'_R, D_{\text{train}}, D_{\text{test}}\right)$$
$$- (1 - \omega) \times f_2\left(N'_R, D_{\text{train}}, D_{\text{test}}\right) \quad (5)$$

where

$$N'_R = fp\left(N, D_{\text{train}}, R\right). \quad (6)$$

The $fp(N, D_{\text{train}}, R)$ represents filter pruning on the network N conditioned on the pruning scheme R and returns the pruned model N'.

The $f_1(N'_R, D_{\text{train}}, D_{\text{test}})$ denotes the accuracy of the pruned model N'_R on the test set D_{test}. The $f_2(N'_R, D_{\text{train}}, D_{\text{test}})$ indicates the complexity of the network by measuring the amount of parameters of N'_R.

The objective function in (5) strives to balance the importance between accuracy and network complexity with the weight parameter ω within [0, 1].

The considered optimization problem is a search problem, which strives to find the lightest network structure, while maintaining accuracy. Moreover, the objective function is implicit on the decision variables, which further complicates the problem. Heuristic optimization algorithm is a widely considered technique to solve such kind of optimization problems. In this article, a PSO-based pruning scheme will be introduced.

C. PSOPruner

The PSO algorithm is a swarm-based optimization algorithm. The principle of PSO algorithm is that a population of candidate solutions are first initialized and evaluated. These candidate solutions are updated according to their fitness to the optimization problem. Such process is repeated until certain stopping condition is met and the best candidate will be selected as the solution to the optimization problem. The flowchart of the proposed PSOPruner is shown in Fig. 7. The associated variables are defined as follows.

1) $\{P_s\}_{s=1}^{P}$: Population of candidate particles P_s in size of P.
2) $P_s(R_s, V_s, R_{\text{sbest}})$: A candidate particle P_s is defined by the tuple where R_s represents the pruning scheme, V_s

denotes particle velocity, and R_{sbest} records the optimal scheme that P_s has ever found during iterations.
3) R_{gbest}: It records the current global optimal scheme by all particles in the population.
4) t: Iteration index starting from 0.

The automatic search process with PSOPruner consists of initialization and iteration.

Initialization: For a particle P_s, the ith component of $R_s = (r_1, r_2, \ldots, r_i, \ldots, r_n)$ and $V_s = (v_1, v_2, \ldots, v_i, \ldots, v_n)$ are initialized by randomly selecting a value from the set of 0.1, 0.2 ,..., 0.9, and $R_{\text{sbest}} = R_s$. The setting that R_s and V_s are constrained to numbers in tenths will reduce the search space and accelerate the search process. However, such setting may reduce the solution quality by PSOPruner. To handle this challenge, a tailored trick, namely, multiround PSOPruner will be applied, which will be discussed in the next subsection.

The particles are evaluated by their fitness to the optimization problem as

$$\text{fit}_s = f(R_s) \quad (7)$$

Iterations: In iterations, these particles cooperate and share information to search the optimal solution. During the iterations, the particles update their position and speed with

$$V_s = \alpha_1 \times V_s + c_1 \times \gamma \times (R_{\text{sbest}} - R_s)$$
$$+ c_2 \times \gamma \times (R_{\text{gbest}} - R_s) \quad (8)$$
$$R_s = R_s + \alpha_2 \times V_s. \quad (9)$$

The parameter α_1 denotes the inertia factor of particle speed. The parameter γ denotes a random value within [0, 1].

The parameters c_1 and c_2 represent the learning rates in (8). The self-cognitive term $(R_{\text{sbest}} - R_s)$ indicates the impact by the optimal scheme that the particle P_s has reached. The group-cognitive term $(R_{\text{gbest}} - R_s)$ indicates the impact by the optimal scheme that the population has reached.

The parameter α_2 is the speed influence factor, which is used to avoid a very large update of the particle position.

Stopping Conditions: If one of the following conditions is met, the search process by the proposed PSOPruner will stop.
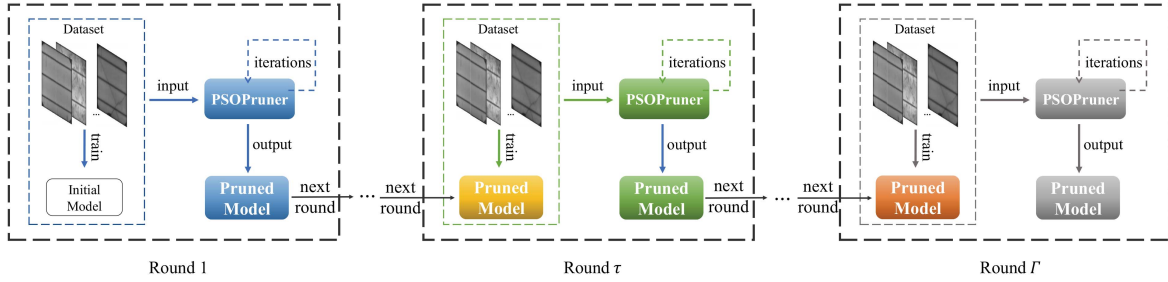
Fig. 8. Multiround optimizations process. Each round initializes different PSOPruner (expressed in different colors).

1) A maximum iteration number maxt is predefined. If the maximum iteration number is reached, the PSOPruner will stop and return the optimal solution.
2) A threshold of fitness value $fit_{expected}$ is considered. If the fitness value of a candidate is greater than the threshold, the PSOPruner will stop.

D. Multiround Optimization

In this article, a tailored trick known as multiround optimization is applied. Such a trick will make great contributions to obtain an extreme lightweight model. The framework of multiround optimization is illustrated in Fig. 8. The initial DCNN serves as the input of the first-round PSOPruner, which will generate a pruned model. The pruned model will be considered as the input of the following round PSOPruner. Such process will be repeated for a certain number of rounds.

Now let us reconsider the problem associated with the assignment of rates to numbers in tenths in the PSOPruner. It is assumed that a total of Γ rounds are carried out and each round constrains pruning rates to 1 decimal place. The final pruned model will gain a precision of Γ decimal places, hence, the overall pruning rate will range in $[0.1^{\Gamma}, 0.9^{\Gamma}]$. The multiround optimization will approximate the search space [0, 1] with increasing Γ.

IV. EXPERIMENT AND RESULT ANALYSIS

A. Multiround Pruning on EL Dataset

Dataset: The high-resolution EL images published by Deitsch et al. [9] in 2019 are used for model development and test. There are 2624 solar cell EL images ($300 \times 300 \times 1$) extracted from both monocrystalline and polycrystalline PV modules. The dataset is divided into 60% for training, 20% for validation, and 20% for test.

Data Augmentation: Data augmentation including scaling, rotation, horizontal flip, vertical flap, and mirror flip can be used to expand small datasets. It can improve the accuracy, enhance the network robustness, as well as alleviate overfitting in training. It is pointed out by [8] that the maximum scaling of the training samples in the dataset is 0.02 times of the original resolution. The rotation interval shall be $(-3°, +3°)$.

Platform: The experimental platform is Ubuntu 20.04.3, PyTorch 1.8.1, CUDA version 11.4 attached with NVIDIA GeForce RTX 3080.

Training Settings: The Adam optimizer is used to train the network. The learning rate is set to 1e-3, while the weight decay with a value of 1e-4 is adopted to reduce overfitting. The momentum decay and scaling decay are set to 0.9 and 0.999, respectively.

The fitness threshold in the stopping conditions is determined by the demanded accuracy and the amount of parameters. This value can be estimated following (5) and can be tuned by prior experiments. The final value is set to 0.7435.

Initial Model: A pretrained ResNet18 is fine-tuned for PV defects classification based on the EL dataset. The well-trained ResNet18 is considered as the initial model for the multiround PSOPruner. In another words, the developed multiround PSO-Pruner is used to prune the filters in the convolutional layers of ResNet18.

The ResNet18 is composed of BasicBlocks. In each BasicBlock, there are two convolutional layers with the same number of filters. The number of filters of BasicBlocks in ResNet18 are 512, 526, 128, and 64. In the proposed PSOpruner, the two convolutional layers in each BasicBlock share the same pruning rate.

Multiround Pruning: Fig. 9 illustrates the distributions of weights of filters in ResNet18. It is noticeable that the filters contain a large number of parameters with the value of 0. Weights with large absolute values carry more information. Fig. 9 reveals that the network may contain redundant convolution kernels in different layers. To study the variability between layers at different depths, experiments with various fixed pruning rates, i.e., 10%, 20%, 30%, and 40% per layer, are conducted. Results in Fig. 10 illustrate that large fixed pruning rates can lead to overpruning on important layers, thereby greatly reducing accuracy. A small fixed pruning rate maintains accuracy, however, its effect on complexity reduction is minor. This proves the rationality of layerwise pruning rate optimization.

Fig. 11 shows the changes of filter number in BasicBlocks during the multiround pruning process. With multiround pruning, the number of filters has changed from (512, 256, 128, 64) to (17, 49, 13, 21), which will greatly reduce the storage to save model parameters and reduce the computational intense in convolutional operation.

For each pruning scheme, the pruned model is retrained to maintain the accuracy. The overall classification performance on all the types of defects and the performance on each type of defect of the initial network and the pruned network with multiround pruning are presented in Tables I and II, respectively.
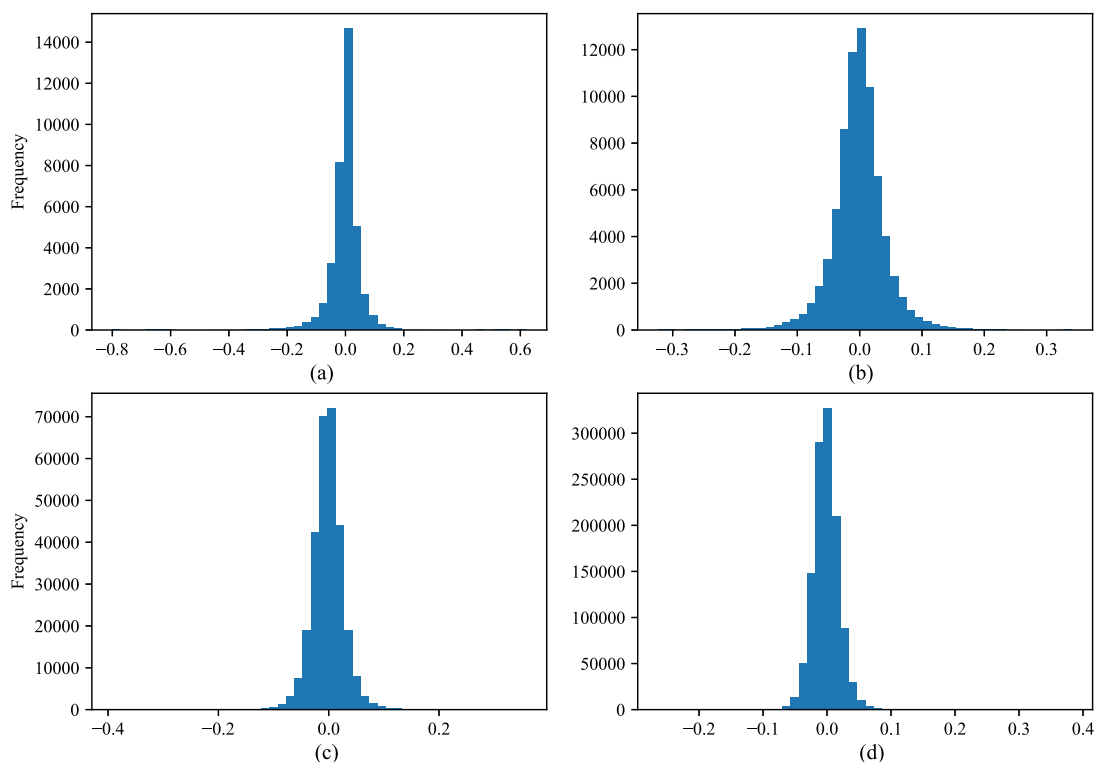
Fig. 9. Distributions of convolution kernel weights from different layers of ResNet18. (a) ResNet18.layer1.0.conv1.weight. (b) ResNet18.layer2.0.conv1.weight. (c) ResNet18.layer3.0.conv1.weight. (d) ResNet18.layer4.0.conv1.weight.
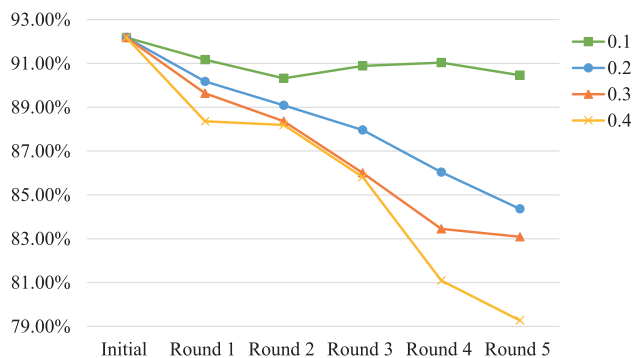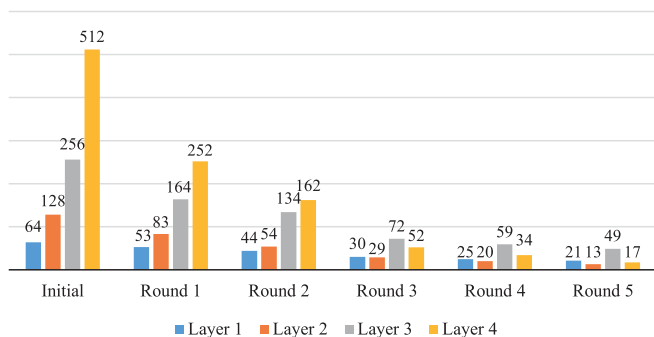

Fig. 10. Accuracy variations with fixed pruning rate.

Fig. 11. Number of filters changed during each round.

TABLE I
CONFUSION MATRICES FOR MULTIROUND PRUNING

Pred. True	Initial		Round 5	
	Fun.	Def.	Fun.	Def.
Fun.	99.03%	28.68%	97.34%	28.68%
Def.	0.97%	71.32%	2.66%	71.32%

TABLE II
PREDICTIONS FOR DIFFERENT DEFECT TYPES

Defects	Initial	Round 5
Finger Interruption	64.10%	61.54%
Cracks	74.24%	71.21%
Contact Forming	80.00%	80.00%
Cell Disconnection	80.00%	90.00%
Material	62.50%	75.00%

From Table I, the defective cells (Def.) can be classified into functional cells (Fun.) both by the initial network and pruned network. This is mainly due to the crystal relocation phenomenon in polycrystalline cells, which produces multiple shadows similar to the defects illustrated in Fig. 3. Without surprising, a small drop of classification performance can be observed from the confusion matrix in Table I by pruning the network. Table II presents the classification specificity for each type of defect, which denotes the rate of defective cells being correctly classified. It should be noted that the cells are classified into

TABLE III
MULTIROUND OPTIMIZATION PERFORMANCE

Multi-round	Initial	Round 1	Round 2	Round 3	Round 4	Round 5
Accuracy	92.18%	90.36%	91.09%	90.55%	91.45%	90.91%
F1 score	95.01%	93.96%	94.21%	93.70%	94.41%	94.16%
Parameters (MB)	11.69	4.50	1.44	0.75	0.49	0.26
FLOPs (GB)	1.82	1.03	0.48	0.33	0.24	0.16
No. of Models	1	31	12	16	22	11
Time Consumption (h)	13.4	23.4	10.7	12.5	16.7	7.3

TABLE IV
PERFORMANCE COMPARISONS

Method	Accuracy	F1 score	Parameters (MB)	FLOPs (GB)
SqueezeNet1_0	87.26%	92.01%	1.25	0.82
ShuffleNetv2_x05	89.75%	93.23%	1.37	0.05
GoogleNet	89.81%	93.60%	6.62	1.50
MobileNetV2	90.45%	93.51%	3.50	0.32
Pruned model	90.91%	94.16%	0.26	0.16
AlexNet	87.33%	91.82%	61.10	0.71
VGG16	86.98%	91.12%	138.36	15.48
VGG19	89.12%	93.40%	143.67	19.65
ResNet18	92.18%	95.01%	11.69	1.82
ResNet50	91.68%	94.84%	25.56	4.11
Pruned model	90.91%	94.16%	0.26	0.16

functional cells or defective cells in [9]. In this article, to investigate the classification performance on each type of defect, the cells are further classified into one type of the defects according to their characteristics discussed in Section II. A small drop of specificity is also observed for defects of finger interruption and cracks. However, for some types of defects, i.e., cell disconnection and material, the specificity is significantly improved. This can be explained by the following two factors: 1) the number of samples in these types of defects is small, hence, the rate is subject to change with a small variation of samples being correctly classified; and 2) network pruning reduces overfitting on these types of defects subsequently improving network performance.

B. Evaluation Metrics

For classification tasks, F1 score can better evaluate the comprehensive performance of the model than accuracy as it balances precision and recall. The following variables are used for the definition of F1 score.

True Positive (tp): The prediction is functional and the truth label is functional.

True Negative (tn): The prediction is defective and the truth label is defective.

False Positive (fp): The prediction is functional and the truth label is defective.

False Negative (fn): The prediction is defective and the truth label is functional.

The F1 score is defined

$$F1\text{score} = \frac{2 \times (\text{precision} \times \text{recall})}{\text{precision} + \text{recall}} \quad (10)$$

where,

$$\text{precision} = \frac{tp}{tp + fp} \quad (11)$$

$$\text{recall} = \frac{tp}{tp + fn}. \quad (12)$$

In this article, the floating-point operations (FLOPs), which depends on the number of filters, the size of the filters, and the size of feature maps, is used as a measurement of computational intense for the forward propagation of the model. The FLOPs for a convolutional layer is computed by [39]

$$\text{FLOPs} = 2 \times H_i \times W_i \times (c_i \times k \times k + 1) \times c_{i+1}. \quad (13)$$

The number of parameters indicates the size of model. Both the FLOPs and the number of parameters amount represent the complexity of the model. The experimental results of the multiround PSOPruner are shown in Table III. It can be observable that the model is greatly compressed in terms of the number of

parameters and FLOPs with a slight drop in accuracy and F1 score. The number of models trained by PSOPruner and time consumption in each round are also included, which illustrates the efficiency of early stopping with predefined fitness threshold in time-saving.

C. Model Comparison

To further illustrate the effectiveness of the proposed multiround PSOPruner in network compression, we compare the performance of the pruned model against the mainstream CNNs including lightweight CNNs in PV defects classification. The results are shown in Table IV. It is observable that the proposed model is the best one in terms of model complexity and processing speed, while the classification performance is also comparative against benchmarks. The results support that the proposed multiround PSO-based pruning can efficiently find a pruning scheme to construct a light network with adequate accuracy for automatic PV defects classification on mobile and embedded devices.

V. CONCLUSION

DCNNs have been widely used for automatic PV defects classification with EL images. However, it is difficult to implement these models on mobile and embedded devices due to the shortage of memory storage and computational capacity. To solve this problem, multiround PSOPruner is proposed for automatic search of optimal DCNN pruning scheme. The proposed multiround PSOPruner deploys the PSO algorithm as the search engine, while a multiround trick is applied to accelerate and simplify the search process. The proposed method is verified on a large public dataset, and the optimal pruning scheme by the multiround PSOPruner can efficiently compress the DCNN in terms of model complexity with a slight drop in model accuracy. The performance of the pruned model is compared with mainstream CNNs. The results illustrate that the pruned model ranks first in terms of model complexity, while its classification performance is comparative. This demonstrates that the proposed multiround PSOPruner is capable of finding a pruning scheme to construct an extreme light network for automatic PV defects classification on mobile and embedded devices.

References

[1] R. H. F. Alves, G. A. de Deus Júnior, E. G. Marra, and R. P. Lemos, "Automatic fault classification in photovoltaic modules using convolutional neural networks," *Renewable Energy*, vol. 179, pp. 502–516, 2021.

[2] B. Mitchell et al., "PERC solar cell performance predictions from multicrystalline silicon ingot metrology data," *IEEE J. Photovolt.*, vol. 7, no. 6, pp. 1619–1626, Nov. 2017.

[3] A. Jäger-Waldau, "Overview of the global PV industry," in *Proc. Reference Module Earth Syst. Environmental Sci.*, 2021, pp. 161–177.

[4] M. L. Stern and M. Schellenberger, "Fully convolutional networks for chipwise defect detection employing photoluminescence images," *J. Intell. Manuf.*, vol. 32, no. 1, pp. 113–126, 2021.

[5] A. M. Karimi et al., "Generalized and mechanistic PV module performance prediction from computer vision and machine learning on electroluminescence images," *IEEE J. Photovolt.*, vol. 10, no. 3, pp. 878–887, May 2020.

[6] A. Dolara, G. C. Lazaroiu, S. Leva, G. Manzolini, and L. Votta, "Snail trails and cell microcrack impact on PV module maximum power and energy production," *IEEE J. Photovolt.*, vol. 6, no. 5, pp. 1269–1277, Sep. 2016.

[7] A. Dolara, S. Leva, G. Manzolini, and E. Ogliari, "Investigation on performance decay on photovoltaic modules: Snail trails and cell microcracks," *IEEE J. Photovolt.*, vol. 4, no. 5, pp. 1204–1211, Sep. 2014.

[8] S. Sobri, S. Koohi-Kamali, and N. A. Rahim, "Solar photovoltaic generation forecasting methods: A review," *Energy Convers. Manage.*, vol. 156, pp. 459–497, 2018.

[9] S. Deitsch et al., "Automatic classification of defective photovoltaic module cells in electroluminescence images," *Sol. Energy*, vol. 185, pp. 455–468, 2019.

[10] F. Haase et al., "Fracture probability, crack patterns, and crack widths of multicrystalline silicon solar cells in PV modules during mechanical loading," *IEEE J. Photovolt.*, vol. 8, no. 6, pp. 1510–1524, Nov. 2018.

[11] G. A. dos Reis et al., "Drone-based daylight electroluminescence imaging of PV modules," *IEEE J. Photovolt.*, vol. 10, no. 3, pp. 872–877, May 2020.

[12] X. Li, W. Li, Q. Yang, W. Yan, and A. Y. Zomaya, "An unmanned inspection system for multiple defects detection in photovoltaic plants," *IEEE J. Photovolt.*, vol. 10, no. 2, pp. 568–576, Mar. 2020.

[13] M. Y. Demirci, N. Beşli, and A. Gümüşçü, "Efficient deep feature extraction and classification for identifying defective photovoltaic module cells in electroluminescence images," *Expert Syst. Appl.*, vol. 175, 2021, Art. no. 114810.

[14] M. W. Akram et al., "CNN based automatic detection of photovoltaic cell defects in electroluminescence images," *Energy*, vol. 189, 2019, Art. no. 116319.

[15] W. Tang, Q. Yang, K. Xiong, and W. Yan, "Deep learning based automatic defect identification of photovoltaic module using electroluminescence images," *Sol. Energy*, vol. 201, pp. 453–460, 2020.

[16] L. Pratt, D. Govender, and R. Klein, "Defect detection and quantification in electroluminescence images of solar PV modules using u-net semantic segmentation," *Renewable Energy*, vol. 178, pp. 1211–1222, 2021.

[17] J. Fioresi et al., "Automated defect detection and localization in photovoltaic cells using semantic segmentation of electroluminescence images," *IEEE J. Photovol.*, vol. 12, no. 1, pp. 53–61, Jan. 2022.

[18] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervention*, 2015, pp. 234–241.

[19] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[21] G. B. Nair and S. Dhoble, "Introduction to Luminescence," in *Proc. Fundamentals Appl. Light-Emitting Diodes*, G. B. Nair and S. Dhoble, Eds., 2021, pp. 3–33.

[22] R. Ebner, B. Kubicek, and G. Újvári, "Non-destructive techniques for quality control of PV modules: Infrared thermography, electro- and photoluminescence imaging," in *Proc. 39th Annu. Conf. IEEE Ind. Electron. Soc.*, 2013, pp. 8104–8109.

[23] M. Köntges et al., "Review of failures of photovoltaic modules," IEA Int. Energy Agency, Paris, France, 2014.

[24] S. Kajari-Schröder, I. Kunze, U. Eitner, and M. Köntges, "Spatial and orientational distribution of cracks in crystalline photovoltaic modules generated by mechanical load tests," *Sol. Energy Materials Sol. Cells*, vol. 95, no. 11, pp. 3054–3059, 2011.

[25] M. Radovic, M. Ghalwash, N. Filipovic, and Z. Obradovic, "Minimum redundancy maximum relevance feature selection approach for temporal gene expression data," *BMC Bioinf.*, vol. 18, no. 1, 2017, Art. no. 9.

[26] A. M. Karimi, J. S. Fada, J. Q. Liu, J. L. Braid, and R. H. French, "Feature extraction, supervised and unsupervised machine learning classification of PV cell electroluminescence images," in *Proc. IEEE 7th World Conf. Photovoltaic Energy Convers. (WCPEC) (A Joint Conf. 45th IEEE PVSC, 28th PVSEC 34th EU PVSEC)*, 2018, pp. 0418–0424.

[27] U. Otamendi et al., "Segmentation of cell-level anomalies in electroluminescence images of photovoltaic modules," *Sol. Energy*, vol. 220, pp. 914–926, 2021.

[28] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-cnn: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[29] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.* 2019, pp. 6105–6114.

[30] M. Mayr, M. Hoffmann, A. Maier, and V. Christlein, "Weakly supervised segmentation of cracks on solar cells using normalized LP norm," in *Proc. IEEE Int. Conf. Image Process.*, 2019, pp. 1885–1889.

[31] S. S. L. Oskouei, H. Golestani, M. Hashemi, and S. Ghiasi, "CNNdroid: GPU-accelerated execution of trained deep convolutional neural networks on android," in *Proc. 24th ACM Int. Conf. Multimedia*, 2016, pp. 1201–1205.

[32] T.-H. Tsai, P.-T. Chi, and K.-H. Cheng, "A sketch classifier technique with deep learning models realized in an embedded system," in *Proc. IEEE 22nd Int. Symp. Des. Diagnostics Electron. Circuits Syst.*, 2019, pp. 1–4.

[33] B. Du, Y. He, Y. He, J. Duan, and Y. Zhang, "Intelligent classification of silicon photovoltaic cell defects based on eddy current thermography and convolution neural network," *IEEE Trans. Ind. Inform.*, vol. 16, no. 10, pp. 6242–6251, Oct. 2020.

[34] W. Tang, Q. Yang, X. Hu, and W. Yan, "Edge intelligence for smart EL images defects detection of PV plants in the IoT-based inspection system," *IEEE Internet Things J.*, to be published, doi: 10.1109/JIOT.2022.3150298.

[35] A. Chindarkkar, S. Priyadarshi, N. S. Shiradkar, A. Kottantharayil, and R. Velmurugan, "Deep learning based detection of cracks in electroluminescence images of fielded PV modules," in *Proc. 47th IEEE Photovoltaic Specialists Conf.*, 2020, pp. 1612–1616.

[36] Y. Guo, A. Yao, and Y. Chen, "Dynamic network surgery for efficient DNNs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 1379–1387.

[37] J. H. Luo et al., "ThiNet: Pruning CNN filters for a thinner net," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 41, no. 10, pp. 2525–2538, Oct. 2019.

[38] T. J. Yang, Y. H. Chen, and V. Sze, "Designing energy-efficient convolutional neural networks using energy-aware pruning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5687–5695.

[39] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.