ACS APPLIED ENERGY MATERIALS

www.acsaem.org Letter

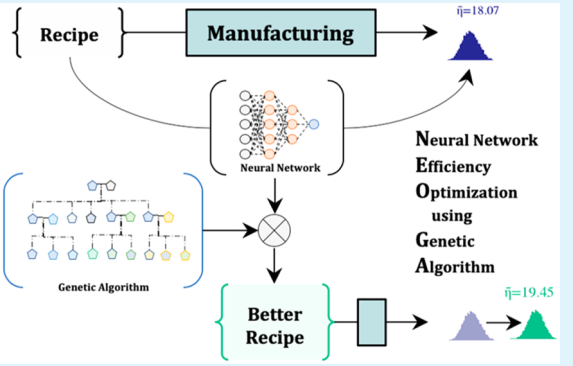
Optimization of Solar Cell Production Lines Using Neural Networks and Genetic Algorithms

Yoann Buratti, Casper Eijkens, and Ziv Hameiri*

Cite This: https://dx.doi.org/10.1021/acsaem.0c01207 Read Online

ACCESS | Metrics & More | Article Recommendations | Supporting Information

ABSTRACT: To keep improving the efficiency-to-cost ratio of photovoltaic solar cells, manufacturing lines must be continuously improved. Efficiency optimization is usually performed process-wise and can be slow and time-consuming. In this study, we propose a machine-learning-based method to perform simultaneous multiprocess optimization. Using the natural variation of a production line, we train machine learning models to investigate the relationship between process parameters and cell efficiency. We employ genetic algorithms to identify new process parameters in order to maximize cell efficiency. The proposed method is demonstrated on a simulated production line of monocrystalline aluminum-back surface field solar cells. Using neural networks, an accurate model is built to predict cell efficiencies from input process parameters with errors of <0.03% absolute efficiency. In five iterations, the mean cell efficiency increases from 18.07% to 19.45%. Provided strong process monitoring and accurate wafer tracking, the proposed method is directly applicable to production-type datasets, enabling the photovoltaic industry to build smart factories and join the fourth industrial revolution.



KEYWORDS: *solar cell, machine learning, neural network, genetic algorithm, process optimization*

Recently, the Intergovernmental Panel on Climate Changes (IPCC) argued that a third of the global renewable energy should be supplied by photovoltaic (PV) energy in order to reduce emissions to acceptable levels.[1] A key pathway to help accelerate solar growth—and to reach the IPCC's goal—is to increase the power conversion efficiency of PV systems, with even or reduced costs.[2] One easy method to achieve these goals is by optimizing the PV manufacturing processes.[3] However, optimizing a production line is neither easy nor inexpensive. For example, optimizing a single process using a full factorial design with five key parameters and three levels per parameter requires 3^5 (243) experimental runs.[4] Hence, to independently optimize 10 consecutive processes, 10 × 243 = 2430 experimental runs are needed. However, this procedure does not consider dependencies between processes, as processes are optimized individually. To measure the effect of these dependencies, the number of needed experimental runs increases to an unrealistic number ($3^{50} > 10^{23}$). Although this number can be reduced using design of experiment (DoE) approaches,[4] such as Taguchi's orthogonal design,[5] they have severe limitations when more than 40 parameters are investigated.[5]

In this study, we propose a machine learning (ML) framework for PV production line optimization. ML has the capacity to learn a wide variety of nonlinear patterns in high-dimensional datasets and can be used to build data-driven models of process intradependencies and interdependencies.[6] ML-based techniques for PV manufacturing have been explored for solar cell material design,[7] optimizing individual processes,[8] and a combination of processes.[9] It has also been explored with regard to DoE optimization,[10] quality control,[11] and troubleshooting with access to wafer tracking.[12] However, simultaneous optimization of the entire fabrication process has not yet been done. This project aims to develop this capability. First, an ML model is built, associating processes parameters with output cell efficiencies. Here, we use the *natural variation* of a production line to create an efficiency distribution. Next, a genetic algorithm (GA)[13] optimizer is implemented to identify a set of process parameters that maximizes the cell efficiency. At last, the new set of process parameters can be implemented in the production line, which generates a new distribution of cell efficiency during operation. The optimization method can then be repeated using this new dataset. We demonstrate the capabilities of the proposed method using a simulated dataset of an aluminum-back surface field (Al-BSF) production line. Using ML, we build an accurate data-driven model that

Received: May 24, 2020
Accepted: October 22, 2020

https://dx.doi.org/10.1021/acsaem.0c01207
ACS Appl. Energy Mater. XXXX, XXX, XXX−XXX

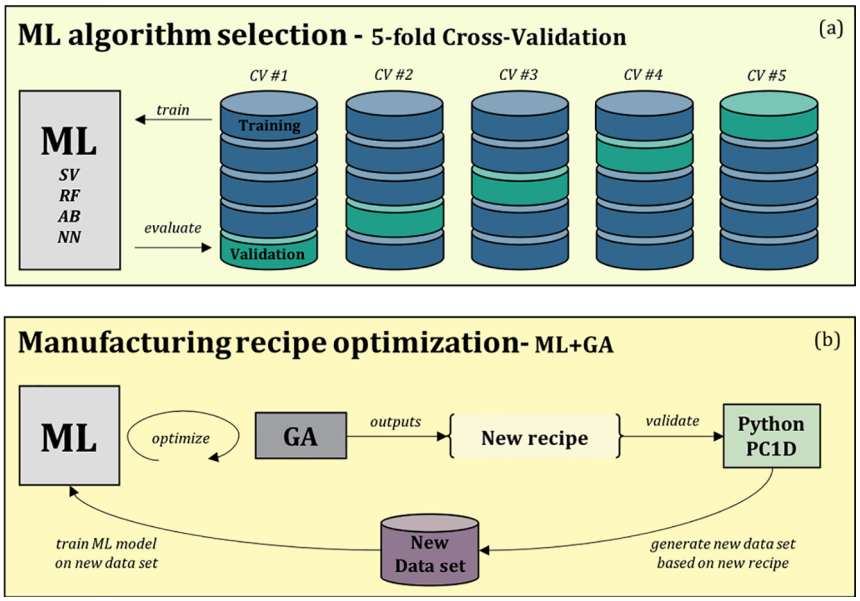


Figure 1. Machine learning algorithm selection using (a) 5-fold cross-validation and (b) an efficiency optimization framework using a genetic algorithm.

predicts cell efficiencies from more than 40 input process parameters, with prediction errors of <0.03% absolute efficiency. The proposed optimization method increases the mean cell efficiency of the simulated production line from 18.07% to 19.45%. The method can be extended to other cell structure, such as passivated emitter and rear contact (PERC) or silicon heterojunction (SHJ) cells and implemented in current solar production lines, taking PV manufacturing one step closer to industry 4.0.[14]

Inspired by UNSW's Virtual Production Line and PV Lighthouse's PV Factory,[15] a Python package has been developed to simulate the industrial manufacturing of monocrystalline Al-BSF silicon solar cells. The simulation features 10 processing steps (such as saw damage etching, diffusion, and passivation) and 47 different process parameter inputs (such as etching duration, diffusion temperature, and deposition gas flow ratio). The outputs of the Python simulation (such as wafer thickness, diffusion profile, and surface recombination velocity) are then fed to PC1D[16]—a finite-element numerical solver for modeling semiconductor devices—to determine the cell efficiency. A combination of process input parameters (a "recipe"), achieving 18% cell efficiency, is chosen to be used as a baseline recipe for this study. A Monte Carlo[17] approach is used to randomly generate process inputs parameters following a Gaussian distribution, centered around the baseline recipe and allowing ±1% variation for each input parameter. This generates the "production" efficiency distribution spread typically seen in Al-BSF solar cell manufacturing lines.[18] It contains ~400 000 cells, with cell efficiencies ranging from 15% to 18.8% with a mean of 17.86% and a standard deviation of 0.32%. The details of the process input parameters and their allowed range are given in the Supporting Information.

Several commonly used ML models are trained and evaluated on the dataset: support vector regression (SV),[19] random forests (RF),[20] adaptive boosting (AB),[21] and neural networks (NN).[22] A 5-fold cross-validation[23] (CV) is used to assess which ML model is best-suited to learn the relationship between the inputted recipe and the resulting cell efficiency, while mitigating the risk of overfitting.[23] As illustrated in Figure 1a, the dataset is randomly split into five blocks, and for each iteration, four blocks are used for training the ML model, while the remaining block is used to evaluate the model ("validation block"). The models are trained independently using the 5-fold cross-validation approach and are scored using the root mean squared error (RMSE) of the absolute cell efficiency and the coefficient of determination (R^2) on the validation block, which are defined as[24]

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{c=1}^{N}(\widehat{\mathrm{Eff}}_c - \mathrm{Eff}_c)^2} \tag{1}$$

and

$$R^2 = 1 - \frac{\sum_{c=1}^{N}(\widehat{\mathrm{Eff}}_c - \mathrm{Eff}_c)^2}{\sum_{c=1}^{N}(\mathrm{Eff}_c - \overline{\mathrm{Eff}})^2} \tag{2}$$

where N is the total number of cells in the validation set, $\widehat{\mathrm{Eff}}_c$ is the predicted cell efficiency of cell c, Eff_c the true efficiency, and $\overline{\mathrm{Eff}}$ the mean efficiency of the validation dataset.

The best performing ML model—i.e., the one scoring the lowest RMSE and highest R^2—is chosen to model the simulated production line. A genetic algorithm (GA)[13] is then employed to maximize the ML-based objective function in order to identify a set of input process parameters—a "recipe"—that results in higher cell efficiencies. This recipe is then validated using the developed Python/PC1D simulation. The final step, as presented in Figure 1b, is to seed a new Monte Carlo-generated distribution based on the GA-optimized recipe. The entire ML and GA optimization method is iterated and results in even higher efficiency recipes.

The dataset simulation, ML model training and GA optimization are implemented using Python packages such as Scikit-learn[25] and DEAP.[26] The data and code supporting this study are available from the corresponding author upon request.

To compare the ML models, a 5-fold cross-validation method[23] is applied. The RMSE and R^2, as a function of the

https://dx.doi.org/10.1021/acsaem.0c01207
ACS Appl. Energy Mater. XXXX, XXX, XXX−XXX

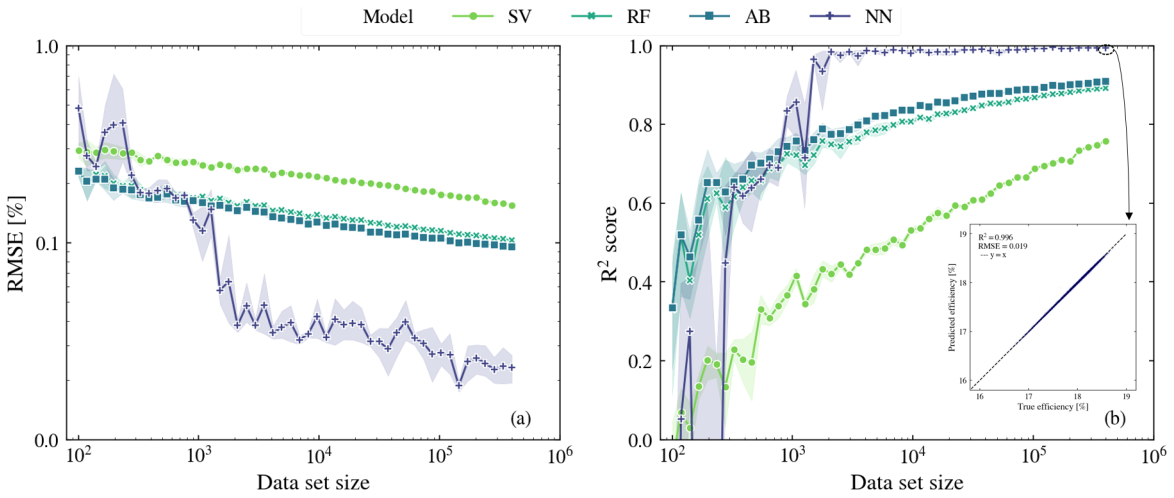


Figure 2. Machine learning algorithm comparison; each symbol represents the average score on the validation block, while the shaded area indicates the 95% confidence interval for both (a) RMSE scores and (b) R^2 scores of models, as a function of dataset size. The best performing neural network efficiency prediction selected by the cross-validation is shown as the true versus predicted cell efficiency on the validation dataset in the inset of panel (b).

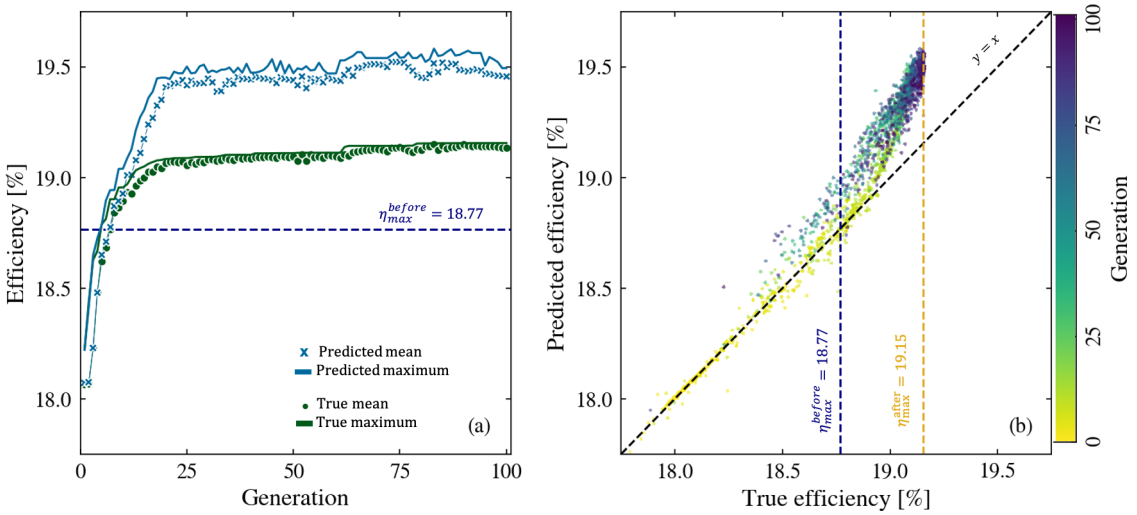


Figure 3. Predicted (NN) and true (Python/PC1D) efficiency from the GA optimization search: (a) averaged and maximum for each generation and (b) plotted against each other for each individual.

size of the used dataset (from N = 100 to N = 400 000), is plotted in Figure 2. For each model, the average score ("symbol") on the validation block and its 95% confidence interval ("shaded area") are shown. The RMSE is shown on a semilogarithmic plot to accentuate the differences between the ML models. For a small dataset size ($N \approx 100$), the four ML models are comparable, with RMSEs averaging 0.3% absolute efficiency error. As the dataset size increases ($N \approx 1000$), the NN-based models significantly improve and outperform the other three ML models in both RMSE and R^2. This remains true for each fold of the CV, mitigating the risk of overfitting.[23] NN models have a higher variance than the other models, because of its sensitivity to the random initial weights and the random order in which the data are processed during training.[27] With access to the entire dataset ($N \approx 400\,000$), the NN models achieve a correlation value of R^2 = 0.994 ± 0.002 and predict efficiency with an RMSE of 0.023 ± 0.004% (absolute). The ability of NN to model more complex relationships without overfitting[28] is assumed to be the main reason for their superior performance. It is concluded that NN has the highest potential as a modeling algorithm for solar cell efficiency prediction. Hence, it is the model chosen for the GA optimizer. Focusing on the best NN model, trained on 80% of the full dataset, the inset in Figure 2b compares the NN's prediction of cell efficiency to their true efficiency (simulated using Python/PC1D). On the validation set, the NN achieves an R^2 of 99.6% and an RMSE of 0.019% absolute efficiency, corresponding to prediction with a relative error of <0.1%. This is a strong demonstration of the ability of NN to learn the complex relationship between a large number of input parameters (47) and the output cell efficiency solely using natural variation within the cell manufacturing processes. This level of accuracy and modeling of interprocess dependencies would be very costly and time-consuming to obtain through a Department of Energy (DoE) approach.[5]

In order to maximize the production line's efficiency, the NN model is passed to a GA optimizer as its objective function. The optimizer starts with a population of 100 random individuals, corresponding to 100 recipes (sets of input parameters) from the "production" dataset. The

https://dx.doi.org/10.1021/acsaem.0c01207
ACS Appl. Energy Mater. XXXX, XXX, XXX−XXX

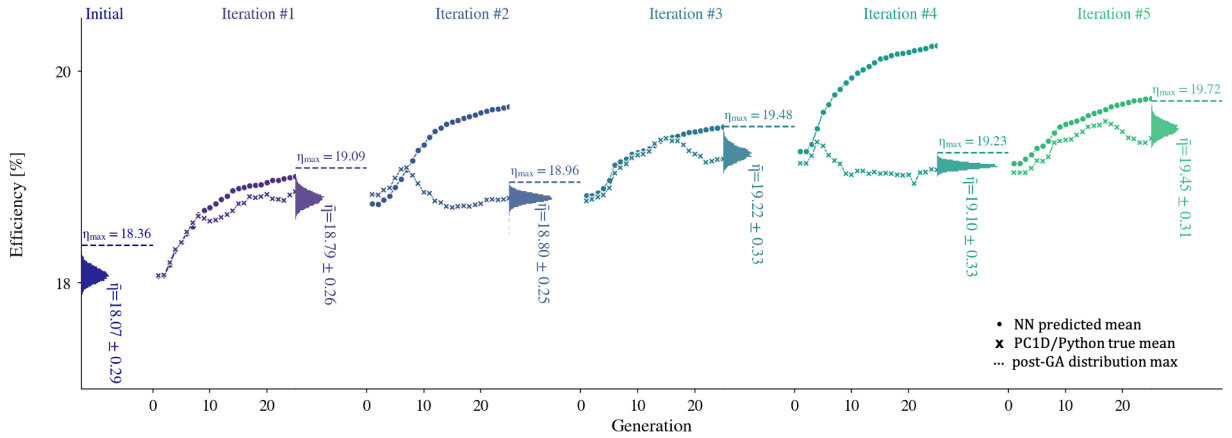


Figure 4. Successive iteration of the NEO-GA approach, showing the GA predicted (NN) and true (Python/PC1D) efficiency of the average of each generation, followed by the efficiency distribution generated by the new recipe found (Python/PC1D); each dotted line reports the maximum efficiency of the distribution, generated from the previous GA step.

optimization process is inspired by Charles Darwin's natural selection theory of the survival of the fittest. In our study, a "fit individual" corresponds to a recipe producing high-efficiency cells, estimated by the NN. A set of selection, breeding, crossover and mutation operations[13,26] transforms the first generation of recipes into a new population of 100 individuals (i.e., new recipes). With each new generation, the overall average fitness (in our study, efficiency) increases.

Applied to our optimization problem, Figure 3a shows the mean (symbol) and maximum (line) efficiency for each generation as predicted by the NN model (light blue). To verify that the newfound recipes are improvements over the original recipes, each new recipe is evaluated by the Python/PC1D simulation (green). The maximum cell efficiency of the original production dataset (18.77%) is marked by the dashed dark blue line. As can be seen, only a few generations (~10) are required to surpass this maximum. Impressively, the GA algorithm identifies recipes with significantly higher efficiencies, compared to the original production dataset, with a maximum effective efficiency of 19.15%, corresponding to a 2% relative increase of efficiency. With each new generation, the disparity between the NN predicted efficiency and the actual Python/PC1D average efficiency increases. As the GA explores recipes further away from the boundaries where the NN model was trained, extrapolation and fitting error widens the disparity seen between predicted (NN) and true (Python/PC1D) efficiency. Above 25 generations, the disparity becomes constant, because a "plateau" stage is reached, corresponding to a local maximum in the efficiency hyperplane modeled by the NN objective function. Even with extrapolation and fitting error, that local maximum still corresponds to a recipe with higher cell efficiency (19.15%) than the maximum of the original dataset (18.77%). An alternative presentation of these data is provided in Figure 3b. Here, every individual of each generation is marked by a colored cross symbol ("+"). Bright colors represent the first generations, while the darker colors represent the later generations. Also shown are the maximum efficiency of the original dataset (blue dashed line), the new obtained maximum efficiency (yellow dashed line) and the diagonal $y = x$ axis (black dashed line). In the early generations (light colors), the individual's efficiency agrees with the line $y = x$, indicating that the NN is still accurate. However, after 25 generations (green and darker), the predictions diverge from their true efficiency, as more and more extrapolation errors are committed by the NN. The optimization algorithm still provides recipes that reach higher efficiencies than the original dataset, achieving a maximum true efficiency of 19.15%, corresponding to a 2% relative increase, compared to the original production dataset.

From the GA optimizer, a new recipe is generated from the average of the best 10 recipes of the last generation. The new recipe is fed back to the Python/PC1D Monte Carlo simulation to generate a new production-type dataset, which is then used to iterate the process of training the NN model and optimizing through the GA. The results of this process (coined as an NEO-GA-NN efficiency optimizer using GA) are shown in Figure 4. Each iteration of NEO-GA consists of three steps: (1) train a NN to model the input−output relationships of a certain distribution, (2) perform GA optimization and select a new recipe, and (3) generate a new "production" data set from the new recipe with the Python/PC1D simulation. For each new distribution, 10 000 cells are simulated, centered on the new recipe, allowing a ±1% variation for each input parameter. From Figure 2, we expect the RMSE of Step (1) to be ~0.033%, providing an accurate model of the relationship between the input parameters and the output cell efficiency. Step (2) is performed with 25 generations, as this amount is sufficient to reach the "plateau" seen in Figure 3. A total of five iterations are performed.

For this part of the study, a new seed production dataset is generated, with a mean efficiency of 18.07% and a maximum efficiency of 18.36%. After five iterations, the production line efficiencies reach a mean of 19.45% (a 7.6% relative increase) and a maximum of 19.72% (a 7.4% relative increase). For each GA optimization step, the same behavior as in Figure 3 is observed, as both NN-predicted (circle) and Python/PC1D-true (cross) average efficiencies match during the first few generations, before reaching a "plateau" from increased extrapolation and fitting errors. The final generation of each GA optimization step is used to generate a new production dataset, which will seed the next iteration of the NEO-GA process. The standard deviation of that new dataset is comparable with the original production dataset (~0.3%), which indicates that the local maximum reached in the plateau phase is stable (less than ±0.5% efficiency variation from ±1% input variation). In iterations 2 and 4, it is seen that the maximum efficiency is lower than the previous distribution. This can be explained by the random approach of the GA and

https://dx.doi.org/10.1021/acsaem.0c01207
ACS Appl. Energy Mater. XXXX, XXX, XXX−XXX

is expected as a consequence of the large parameter space under investigation. However, in both cases, the disparity between the mean efficiency and the maximum efficiency is reduced, allowing the NN to learn the relationship between the parameters and the cell efficiency as close as possible to the maximum reached efficiency, which will minimize extrapolation error in the following iteration.

To put the significance of such an iterative improvement in perspective, in the International Technology Roadmap for Photovoltaics (ITRPV) reports of 2010, mono-Si production average cell efficiencies were reported to be 18% and predicted to reach 19.5% in 2015, five years later.[29] The proposed NEO-GA method has the potential to achieve rapid optimization of solar cell manufacturing within only a few iterations, and could be implemented in a matter of days. Assuming that a ±1% input parameter variation is within statistical variation of a manufacturing line, by implementing rigorous statistical process control, monitoring processes actual conditions and wafer tracking, which is the direction predicted by the last ITRPV report,[30] it would be possible to build an initial dataset from a standard industrial production line. By implementing the NEO-GA iteration method, running a new production batch with the newly found recipe in lieu of Step (3), it can be expected to significantly improve the average and maximum efficiency of the line, within an acceptable time frame. Note that we validate that the parameters outputted by the NEO-GA are >1% from the baseline recipe (because we assume that changes below this threshold are too small to be practical) and stay within the expected process parameter window. On average, GA optimization lead to input process parameters that differed by 10% from the baseline recipe (maximum change of 25%), which offers useful insights on the production line. Furthermore, by training the NN, an accurate model of the production line can be formed that can be used for R&D projects or further optimization schemes.

In this study, we proposed a method (coined NEO-GA) to optimize production lines of solar cells. The method was demonstrated using a simulated Al-BSF production line. An NN algorithm was shown to accurately model the relationship between input process parameters and output cell efficiency, achieving an RMSE of <0.03% absolute efficiency. The model produced by the NN is then used as an objective function for a GA optimizer and outputs a new set of input process parameters, resulting in a relative maximum efficiency increase of 2%. The proposed NEO-GA can be looped to continue increasing the production line efficiency and building more accurate NN models. Starting with a mean efficiency of 18.07%, the NEO-GA loops increased the production line efficiencies by 7.6%, relative to a mean of 19.45%. Considering an actual production line natural variation, the NEO-GA approach can be applied and maximize the efficiency after a few iterations, as well as building an accurate ML model of the production line. The proposed approach is not limited by the number of input parameters or the complexity of the model and can be extended from Al-BSF to other cell structures, such as PERC or HJT. Furthermore, the NEO-GA approach can be applied to groups of cells, such as cassettes or diffusion boats, to help enhance the adoption of ML and data-driven models in the PV industry.

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acsaem.0c01207. The ML training, GA optimization code, simulated production data, and graph data are available on GitHub [www.github.com/WhyBeU/NEOGA].

Table of input process parameters with upper and lower boundaries and baseline recipe (PDF)

■ AUTHOR INFORMATION

Corresponding Author

Ziv Hameiri − *The University of New South Wales, Sydney, NSW 2052, Australia;* orcid.org/0000-0002-2934-4478; Email: z.hameiri@unsw.edu.au

Authors

Yoann Buratti − *The University of New South Wales, Sydney, NSW 2052, Australia;* orcid.org/0000-0002-3621-9712

Casper Eijkens − *Delft University of Technology, Delft 2628, CD, Netherlands;* orcid.org/0000-0002-8720-182X

Complete contact information is available at: https://pubs.acs.org/10.1021/acsaem.0c01207

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

The authors thank Dr. Keith McIntosh and Dr. Malcolm Abbott (PV Lighthouse, Australia) and Dr. Halvard Haug (Institute for Energy Technology, Norway) for their help and feedback in developing the Al-BSF simulation package and adapting PC1D for Python. This work was supported by the Australian Government through the Australian Renewable Energy Agency [ARENA; Project Nos. 2017/RND001 and 2017/RND017] and the Australian Centre for Advanced Photovoltaics (ACAP). The views expressed herein are not necessarily the views of the Australian Government, and the Australian Government does not accept responsibility for any information or advice contained herein.

■ REFERENCES

(1) IPCC. Summary for Policymakers. In *Global Warming of 1.5 °C—An IPCC Special Report on the Impacts of Global Warming of 1.5 °C above Pre-Industrial Levels and Related Global Greenhouse Gas Emission Pathways, in the Context of Strengthening the Global Response to the Threat of Climate Change, Sustainable Development, and Efforts to Eradicate Poverty*; Masson-Delmotte, V., Zhai, P., Pörtner, H.-O., Roberts, D., Skea, J., Shukla, P. R., Pirani, A., Moufouma-Okia, W., Péan, C., Pidcock, R., Connors, S., Matthews, J. B. R., Chen, Y., Zhou, X., Gomis, M. I., Lonnoy, E., Maycock, T., Tignor, M., Waterfield, T., Eds.; World Meteorological Organization, Geneva, Switzerland, 2018.

(2) Green, M. A. How Did Solar Cells Get so Cheap? *Joule* **2019**, *3* (3), 631−633.

(3) Kavlak, G.; McNerney, J.; Trancik, J. E. Evaluating the Causes of Cost Reduction in Photovoltaic Modules. *Energy Policy* **2018**, *123*, 700−710.

(4) Myers, R. H.; Montgomery, D. C.; Anderson-Cook, C. M. *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*; John Wiley & Sons, 2016.

(5) Roy, R. K. *Design of Experiments Using The Taguchi Approach: 16 Steps to Product and Process Improvement*; Wiley: Hoboken, NJ, 2001.

(6) Hornik, K.; Stinchcombe, M.; White, H. Multilayer Feedforward Networks Are Universal Approximators. *Neural Networks* **1989**, *2* (5), 359−366.
(7) Fenning, D. P.; Hofstetter, J.; Morishige, A. E.; Powell, D. M.; Zuschlag, A.; Hahn, G.; Buonassisi, T. Darwin at High Temperature: Advancing Solar Cell Material Design Using Defect Kinetics Simulations and Evolutionary Optimization. *Adv. Energy Mater.* **2014**, *4* (13), 1400459.
(8) Bae, H.; Jeon, T.-R.; Kim, S.; Kim, H.-S.; Kim, D.; Han, S.-S.; May, G. S. Optimization of Silicon Solar Cell Fabrication Based on Neural Network and Genetic Programming Modeling. *Soft Comput.* **2010**, *14* (2), 161−169.
(9) Wagner-Mohnsen, H.; Altermatt, P. P. A Combined Numerical Modeling and Machine Learning Approach for Optimization of Mass-Produced Industrial Solar Cells. *IEEE Journal of Photovoltaics* **2020**, *10* (5), 1441−1447.
(10) Cao, B.; Adutwum, L. A.; Oliynyk, A. O.; Luber, E. J.; Olsen, B. C.; Mar, A.; Buriak, J. M. How to Optimize Materials and Devices via Design of Experiments and Machine Learning: Demonstration Using Organic Photovoltaics. *ACS Nano* **2018**, *12* (8), 7434−7444.
(11) Mevawalla, Z. N.; May, G. S.; Kiehlbauch, M. W. Neural Network Modeling for Advanced Process Control Using Production Data. *IEEE Transactions on Semiconductor Manufacturing* **2011**, *24* (2), 182−189.
(12) Kloter, B. Application of Machine Learning for Production Optimization. In *2018 IEEE 7th World Conference on Photovoltaic Energy Conversion (WCPEC) (A Joint Conference of 45th IEEE PVSC, 28th PVSEC & 34th EU PVSEC)*, Waikoloa Village, HI, June 10−15, 2018; IEEE, *2018*; pp 3489−3491.
(13) Holland, J. H. Genetic Algorithms and Adaptation. In *Adaptive Control of Ill-Defined Systems*; Selfridge, O. G., Rissland, E. L., Arbib, M. A., Eds.; NATO Conference Series II: Systems Science, Vol. *16*; Springer: Boston, MA, 1984; pp 317−333.
(14) Kagermann, H. Change Through Digitization—Value Creation in the Age of Industry 4.0. In *Management of Permanent Change*; Albach, H., Meffert, H., Pinkwart, A., Reichwald, R., Eds.; Springer Fachmedien: Wiesbaden, Germany, 2015; pp 23−45.
(15) Abbott, M. D.; McIntosh, K. R.; Lennon, A. J.; Cotter, J. E.; Li, Y.; Lu, Z.; Li, A. Online Education with PV Factory. In *42nd Photovoltaic Specialist Conference*; IEEE, 2015; pp 1−5.
(16) Haug, H.; Greulich, J. PC1Dmod 6.2−Improved Simulation of c-Si Devices with Updates on Device Physics and User Interface. *Energy Procedia* **2016**, *92*, 60−68.
(17) Rubinstein, R. Y.; Kroese, D. P. *Simulation and the Monte Carlo Method (Wiley Series in Probability and Statistics)*, 2nd Edition; Wiley, 2007.
(18) Green, M. A. Forty Years of Photovoltaic Research at UNSW. In *J. Proc.—R. Soc. New South Wales*; **2015**, *148*, 2−14.
(19) Basak, D.; Pal, S.; Patranabis, D. C. Support Vector Regression. *Neural Inform. Process.—Lett. Rev.* **2007**, *11* (10), 203−224.
(20) Breiman, L. Random Forests. *Machine Learning* **2001**, *45* (1), 5−32.
(21) Friedman, J. H. Stochastic Gradient Boosting. *Computational Statistics & Data Analysis* **2002**, *38* (4), 367−378.
(22) Hinton, G. E. Connectionist Learning Procedures. *Artificial Intelligence* **1989**, *40* (1−3), 185−234.
(23) Kohavi, R. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Vol. *2*; Association for Computing Machinery, 1995; pp 1137−1143.
(24) Glantz, S. A.; Slinker, B. K. *Primer of Applied Regression and Analysis of Variance*; McGraw−Hill: New York, 1990.
(25) Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; Duchesnay, E. Scikit-Learn: Machine Learning in Python. *J. Machine Learning Res.* **2011**, *12*, 2825−2830.
(26) Fortin, F.-A.; De Rainville, F.-M.; Gardner, M.-A.; Parizeau, M.; Gagné, C. DEAP: Evolutionary Algorithms Made Easy. *J. Machine Learning Res.***2012**, *13*, 2171−2175.
(27) Nielsen, M. A. *Neural Networks and Deep Learning*; Determination Press, 2015.
(28) LeCun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *521* (7553), 436−444.
(29) Fischer, M. International Technology Roadmap for Photovoltaics; 2010.
(30) Fischer, M.; Woodhouse, M.; Herritsch, S.; Trube, J. International Technology Roadmap for Photovoltaics, 2019 Results, 11th Edition; VDMA Photovoltaic Equipment: Frankfurt am Main, Germany, 2020.

https://dx.doi.org/10.1021/acsaem.0c01207
ACS Appl. Energy Mater. XXXX, XXX, XXX−XXX

Optimization of Solar Cell Production Lines using Neural Networks and Genetic Algorithms

Yoann Buratti[1], Casper Eijkens[2], and Ziv Hameiri[1]*

[1]The University of New South Wales, Sydney, NSW, 2052, Australia

[2]Delft University of Technology, Delft, 2628 CD, Netherlands.

[*]Corresponding author: z.hameiri@unsw.edu.au

SUPPORTING INFORMATION

Process step	Variable	Lower bound	Upper bound	Baseline recipe
Incoming wafer quality	cellTh0	160	280	211.50
Incoming wafer quality	tau0	50	1000	1000.00
Incoming wafer quality	bulkFactor	0	2	1.00
Incoming wafer quality	sawDmg0	8	12	8.00
Saw-damage etch	etchT	0	90	51.54
Saw-damage etch	etch_t	0	40	10.09
Saw-damage etch	etchPrev	0	40	0.00
Rantex	NaOH	1	3	3.00
Rantex	propanol	1	10	4.98
Rantex	textT	50	100	63.69
Rantex	text_t	0	30	26.83
Rantex	exhaust	0	4	2.07
Pre-diffusion clean	HFconc	0	30	26.71
Pre-diffusion clean	HClconc	0	30	26.33
Pre-diffusion clean	clean_t	0	10	8.02
Pre-diffusion clean	rinse_t	0	10	9.36
Belt diffusion	Pconc	10	100	89.69
Belt diffusion	diffV	0.1	1	0.31
Belt diffusion	diffTdry	100	300	123.96

Belt diffusion	diffT2	700	1000	900.00
Plasma edge isolation	plasmaP	150	1500	568.60
Plasma edge isolation	plasma_t	2	20	11.05
PECVD SiNx	PECVD_t	1	30	15.61
PECVD SiNx	PECVD_T	0	400	153.52
PECVD SiNx	SiH4	1	5	1.59
PECVD SiNx	NH4	1	5	1.51
Screen-print Al paste	dAl	50	250	243.67
Screen-print Al paste	posAl	0	30	4.21
Screen-print Al paste	negAl	0	30	28.23
Screen-print Al paste	phiAl	20	120	20.00
Screen-print Al paste	pAl	1	10	5.02
Screen-print Al paste	muAl	1	10	2.06
Screen-print Al paste	vAl	1	10	3.80
Screen-print Ag paste	dAg	50	250	154.69
Screen-print Ag paste	posAg	0	30	3.62
Screen-print Ag paste	negAg	0	30	20.83
Screen-print Ag paste	phiAg	20	120	50.71
Screen-print Ag paste	pAg	1	10	7.29
Screen-print Ag paste	muAg	1	10	6.50
Screen-print Ag paste	vAg	1	10	8.11
Screen-print Ag paste	wAg0	50	250	152.85
Screen-print Ag paste	pitchAg	1	6	3.04
Cofiring	O2	0	100	37.97
Cofiring	N2	0	20	12.64
Cofiring	fireV	1	10	6.20
Cofiring	fireTdry	100	300	210.86
Cofiring	fireT2	700	1000	830.00