Computational Materials Science 214 (2022) 111668

Contents lists available at ScienceDirect

Computational Materials Science

journal homepage: www.elsevier.com/locate/commatsci

Full length article

Skeletal-based microstructure representation and featurization through descriptors

Devyani Jivani, Olga Wodo *

Materials Design and Innovation Department, University at Buffalo, NY, United States

ARTICLE INFO

Keywords:
Microstructure
Machine learning
Organic photovoltaics

ABSTRACT

Modeling process–structure–property relationships using machine learning methods has become a valuable enabler for materials design and discovery. However, the machine learning models rely heavily on the featurization of the materials' structure. This paper introduces a microstructure featurization framework to compute generic topological and morphological descriptors. The framework relies on our skeletal microstructure representation. The representation allows for the seamless calculation of topological descriptors, which is the main focus of this paper.

To demonstrate the efficacy of our featurization framework, we couple it with a feature selection method to establish the structure–property model for organic photovoltaics (OPV). For this goal, we identify a *salient* set of descriptors and construct the structure–property map with high accuracy.

1. Introduction

The holy grail of materials science is to establish quantitative structure–property (SP) relationships and apply them toward materials design and discovery [1]. Recently, there has been an increasing interest in using machine learning methods (ML) to establish SP maps [2]. For this goal, the material structure needs to be converted into a format compatible with the machine learning methods and be computationally manageable. However, material structure is hierarchical in nature [3] with scales varying from electronic and atomic systems through microstructures to macrostructures. Even while focusing on a single scale, the raw structural data may contain information about several material aspects (e.g., phase distribution, crystal structure). They can be of a high dimensions (100^3 voxels) requiring large training datasets, effectively impacting the robust data-driven model construction [4].

In this paper, we focus on materials at the microstructure level. The materials' microstructures are typically imaged in 2D or 3D that capture the composition or the phases in a material. The images are of high resolutions to capture the details, making SP maps even more data-demanding to establish. Hence, one critical question is how to efficiently process microstructures to establish SP maps and then solve the inverse problem of designing microstructures with desired properties. In this context, three closely related terms are used: microstructure featurization, microstructure quantification, and microstructure representation.

Featurization is the process of converting various forms of data (e.g., microstructure) to numerical data, which can be directly used as an input to machine learning algorithms [5]. In most cases, the featurization involves vectorizing the raw data into a vector capturing physically meaningful descriptors in low dimensions [5]. Microstructure quantification refers to the process of extracting statistical information through function(s) and/or descriptors [6]. The goal of quantification is to capture spatial correlations among different locations in a microstructure and/or to capture physically meaningful characteristics of the microstructure (e.g., volume fraction, interfacial area). The outcome of microstructure quantification may be equivalent to featurization and can be directly (or with the additional step of data projection into lower dimensions) used in ML methods [7]. In most cases, the quantification provides physically meaningful reasoning used to establish scaling laws of materials behavior. Finally, microstructure representation refers to the form in which data is stored, processed, and transmitted [8]. For example, arrays (or matrices), graphs [9], the bag of visual features [10], spectral density functions [11], and skeleton [12]. Each representation is defined to ease a specific task in machine learning methods. For example, a bag of visual features has been used to capture the local features, create the vocabulary of features and ultimately enable microstructure comparison and classification [10]. Alternatively, graphs have been used to extract information about phase connectivity or path lengths at a low computational cost [9]. Similar to microstructure quantification and featurization, some forms of microstructure representation can be directly used in ML pipelines.

* Corresponding author.
E-mail address: olgawodo@buffalo.edu (O. Wodo).

https://doi.org/10.1016/j.commatsci.2022.111668
Received 15 March 2022; Received in revised form 11 July 2022; Accepted 13 July 2022
Available online 11 August 2022
0927-0256/© 2022 Elsevier B.V. All rights reserved.

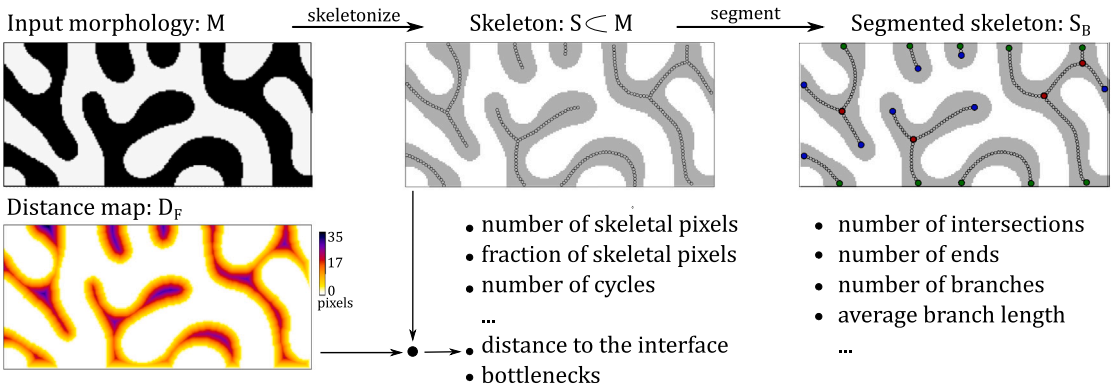


Fig. 1. Workflow to determine skeletal descriptors for two-phase microstructure. The steps involve skeletonization followed by skeletal segmentation. The example descriptors are listed below the corresponding step.

In this paper, we leverage our recently introduced skeletal representation of microstructure [12]. It allows for storage of microstructure information with a small storage footprint. Our work [12] demonstrated two orders of magnitude reduction in the number of required variables compared to the pixel-based microstructure representation. We demonstrated two-way transformation [12] from the image-based representation to skeletal representation (using skeletonization) and reconstruction of the image-based representation from the representation (using convolution surfaces). Finally, in its mathematical form, it can be used for topology optimization where the skeleton and kernels can be modified independently. However, it is non-trivial to use the skeletal representation in its original form in ML approaches to establish SP maps. Nevertheless, the skeletal representation can be featurized with the suite of seamlessly computable descriptors for ML models. This is the scope of this paper. Using the skeletal microstructure representation, we define and implement the suite of physically meaningful skeletal descriptors. The descriptor suite includes comprehensive features related to the topological and morphological characteristics of the microstructures. We extract the descriptors like number and length of branches, number of intersections, domain widths, among others, from the microstructure skeleton. These descriptors are seamlessly computed using our skeletal representation either directly from the skeleton or by combining with the information stored in the distance maps (distance from the given phase to the interface). And although most of these descriptors encode the topological information, we refer to them as "skeletal" based on their origin.

To illustrate the utility of the suite, we discuss an application in the results section. We use the feature selection method to identify a smaller set of descriptors that explain the most variability in the material properties. Effectively, the aim is to show that one can construct SP maps with the derived descriptors. We show that for the application discussed, out of 20 descriptors, using a feature selection method, only seven descriptors are needed for building predictive models with high accuracy.

2. Method

In this section, we describe the methodology for calculating descriptors based on the skeletal representation of microstructures. Fig. 1 depicts the workflow for the descriptor calculation. Panels of the figure depict the intermediate steps to calculate descriptors with two major algorithms: (i) skeletonization and (ii) skeletal segmentation. The input is the microstructure, labeled M in Fig. 1, which is skeletonized to determine the skeleton S from the distance map D_F. The example distance map is plotted as a heat map in Fig. 1. The skeleton is then segmented into branches for descriptor calculations.

Three levels of information and two associated algorithms are critical for seamless descriptor calculations. Subsets of descriptors are calculated at intermediate steps based on the information available at that step. For example, basic descriptors like the skeleton length and cycles are calculated from the skeleton S. But only when the skeleton is segmented into branches and stored as S_B more complex descriptors can be computed (e.g., the number of intersections, the number of end pixels, or branch length). In this sense, the skeletal representation provides a granularity of information for calculating descriptors and allows for seamless integration of information. This is the case when information about microstructure domains (contained in the distance map D_F) and the skeleton are combined to identify bottlenecks along the skeleton (more details below).

In the following section, we define the basic concepts used to calculate the descriptors. In the later sections, we define the skeletal descriptors and include an application to demonstrate the utility of the proposed descriptors.

2.1. Basic definitions

The formal definition of the skeletal representation can be found in our recent work [12], below we provide the definitions used in this paper.

1. Microstructure $M = \{M_{i,j}\}_{n_x \times n_y}$ is formally defined through a local states $M_{i,j}$ that can take values as defined in Eq. (1):

$$M_{i,j} = \begin{cases} 0, & \text{if } (x_{i,j}, y_{i,j}) \in \text{ phase A} \\ 1, & \text{if } (x_{i,j}, y_{i,j}) \in \text{ phase } B \end{cases} \tag{1}$$

   where $M_{i,j}$ corresponds to the local state at the location i, j in the input matrix M. The local state can take one of two values, $\{0, 1\}$ to denote phase A or B, respectively. We describe all the operations and descriptors for phase A (in the figures encoded in black or when needed in gray for clear display). However, the procedure is general and can be applied to phase B, or it can be extended to multi-phase material systems.
   For descriptor definition, we distinguish two vectors M_A, and M_B that store the pixels belonging to phase A and phase B only. For instance, M_A, consists of positions of pixels (i, j) that belong to phase A of the microstructure. Therefore, the number of pixels in phase A is $|M_A|$, where, $|.|$ is the cardinality of the set M_A, and similarly, the number of pixels in phase B is $|M_B|$.
2. Distance map D_F is an array of distances from each pixel of a given phase to the interface between two phases [13]. Distance map is defined for all the continuous domains in the microstructure. An example distance map for phase A is plotted in Fig. 1 below the input microstructure. The darker pixels on the microstructure represent a greater distance to the interface, and the lighter pixels lie close to the interface with the opposite phase. Map D_F is used to determine the skeleton and calculate some of the descriptors.

3. Skeleton S, also known as the "medial axis" [14–16], is a thin (one pixel thick) set of skeletal pixels that captures the topology of a phase [16]. The pixels belonging to the skeleton are equidistant from phase boundaries (here the interface between two phases). The skeletonization algorithm is applied to each domain of a given phase and involves three major steps: (i) calculating distance map D_F to the boundary (interface between phase A and phase B), (ii) deleting pixels from the boundary using D_F, and (iii) repeating step (ii) until no more pixels can be deleted without affecting the connectivity and the topology of the skeleton. The aim is to reduce domain dimensionality of the continuous domain while preserving its topology. As a consequence, the set of pixels on the skeleton are defined as the subset of the microstructure pixels:

$$S = \{s_1, s_2, s_3, \dots, s_n\}, \quad S \subseteq M \tag{2}$$

where s_i is the skeletal pixel. Fig. 1 illustrates an example of an input two-phase microstructure M, and the corresponding skeleton S extracted for phase A (marked with black circles corresponding to skeletal pixels in the middle top panel).

4. Neighborhood $N_8(p)$ is the set of pixels surrounding given pixel p in the microstructure M. It consists of the orthogonal and diagonally adjacent pixels to p (in 2D this corresponds to eight pixels surrounding the pixel under consideration). This type of neighborhood is known as Moore's neighborhood [17]. In this work, we also use the $N_4(p)$ neighborhood[1] known as von Neumann neighborhood [18].

5. End skeletal pixels S_E are terminating pixels in the skeleton S that have one skeletal pixel in its neighborhood N_8. For every pixel s_i in the set S, the number of other skeletal pixels are counted in the neighborhood N_8. If N_8 contains exactly one pixel that belongs to S, the pixel s_i is classified as an end and is stored in S_E. Formally,

$$S_E = \{s_i\} \quad \text{where} \quad \sum_{m=1}^{8} N_8(s_i) = 1 \quad \forall \quad N_8(s_i) \in S \tag{3}$$

In Fig. 1, the end skeletal pixels are denoted with blue circles. In this work, we classify ends into two categories: boundary ends and bulk ends. Boundary end skeletal pixels belong to the top and bottom boundaries of the input microstructure, and bulk end skeletal pixels are the other remaining ends that lie in the volume of the microstructure.

6. Intersection skeletal pixels S_J are the skeletal pixel that belong to at least three branches (see branch definition below). Simply, an intersection skeletal pixel is the junction pixel between the branches. Similar to the end skeletal pixels, the intersections are determined by checking the neighborhood N_8 of $s_i \in S$. For each pixel $s_i \in S$, the skeletal pixels in the neighborhood N_8 are counted. If N_8 contains more than two pixels that belong to S, the pixel s_i is classified as an intersection and stored in S_J.[2] In Fig. 1, intersections are denoted with filled black circles in the panel containing the segmented skeleton S_B. Formally,

$$S_J = \{s_i\} \quad \text{where} \quad \sum_{m=1}^{8} N_8(s_i) \geq 3 \quad \forall \quad N_8(s_i) \in S \tag{4}$$

7. Skeletal segmentation is an operation of dividing the skeleton into segments that we call branches. The branches are determined by traversing[3] through the set S, and segmenting S at an intersection or an end pixel. Each set in S_B is a branch of the skeleton and contains the sequence of pixels that lie on that branch (denoted as b_i in Eq. (5)). Formally,

$$\begin{aligned} & S_B = \{b_1, b_2, b_3, \dots, b_{n_B}\}, \\ \textit{where} \quad & b_1 = \{s_1, s_2, s_3, \dots, s_{l_1}\} \\ & b_2 = \{s_1, s_2, s_3, \dots, s_{l_2}\} \\ & b_3 = \{s_1, s_2, s_3, \dots, s_{l_3}\} \\ & \vdots \\ & b_{n_B} = \{s_1, s_2, s_3, \dots, s_{l_{n_B}}\} \end{aligned} \tag{5}$$

where l_i is the length of individual branch b_i, and n_B is the total number of branches in the skeleton. The branches are non-overlapping, except for the intersections. Each intersection skeletal pixel belongs to all branches where it intersects. In Fig. 3, the skeleton S is segmented into S_B, where S_B is a set of sets. The intersections appear in all the branches that emerge from it. Fig. 3 illustrates an example of skeletal segmentation into branches $(b_1, b_2, \dots, b_{15})$ in the microstructure.

8. Branch of the skeleton S is a sequence of pixels that is terminated by end point or intersection pixels (formally defined in Eq. (5)). Three combinations or terminating points are possible: end–end or an end-intersection or an intersection–intersection. The branches are determined by segmenting the skeleton via skeletal segmentation operation.

9. Cycle is a consecutive sequence of skeletal pixels that are connected in a closed path. To calculate this value, the skeleton S is represented as an undirected graph for which the paths can be computed using classic algorithms in the graph theory. Intuitively, for a graph, a cycle is defined as a non-empty subset of the graph that forms a path such that the first skeletal pixel on the path corresponds to the last one.[4]

2.2. List of skeletal descriptors

Given an input microstructure M, the distance map D_F and the segmented skeleton S (and S_B), we define the skeletal descriptors. Formally, each descriptor is denoted as d_i and constitutes the vector of descriptors for each phase of a microstructure. For phase A:

$$d_A = \{d_1, d_2, d_3, \dots, d_{n_d}\} \tag{6}$$

where n_d is the total number of descriptors for a given phase. In an analogous way, the descriptor vector for phase B is defined by d_B, where each descriptor d_i belongs to the vector:

$$d_B = \{d'_1, d'_2, d'_3, \dots, d'_{n_d}\} \tag{7}$$

The complete descriptor vector for the input two-phase microstructure is denoted here by d_M (Eq. (8))

$$d_M = \{d_A, d_B\} \tag{8}$$

The skeletal descriptors include two descriptors: descriptors computed directly from the skeleton and descriptors computed from the segmented skeleton. Fig. 2 illustrates examples of the skeletal characteristics of a segmented skeleton. Panels A and B of that figure depict

[1] This is also known as 4-neighborhood and composes the central pixel and the 4 adjacent pixels.

[2] Actual implementation is more complex and the steps are included in Appendix.

[3] Graph traversal refers to the process of visiting each vertex in a graph. Graph-traversal algorithms are based on the order in which the vertices are visited, e.g., depth-first search and breadth-first search. In this work, traversing occurs along the skeleton to visit each skeletal pixel.

[4] An undirected graph $G = (V, E)$ is defined by a set of vertices, V, and a set of edges, E, where each edge in E is an unordered pair of vertices drawn from V. The vertices correspond to the pixels in S. The edge is the unordered pair of skeletal pixels directly adjacent in M. The set of edges ensures the connectivity of S. A path between a source vertex, $v_o \in V$, and a target vertex, $v_k \in V$ is a sequence $w = [v_0, v_1, \dots v_i \dots v_k]$ of vertices such that for each i from 0 to $i-1$, vertices v_i and v_{i+1} are adjacent in G. For a cycle, the source v_o and the target v_k correspond to the same vertex in G. The microstructure is assumed to be nonperiodic for calculating the cycles.

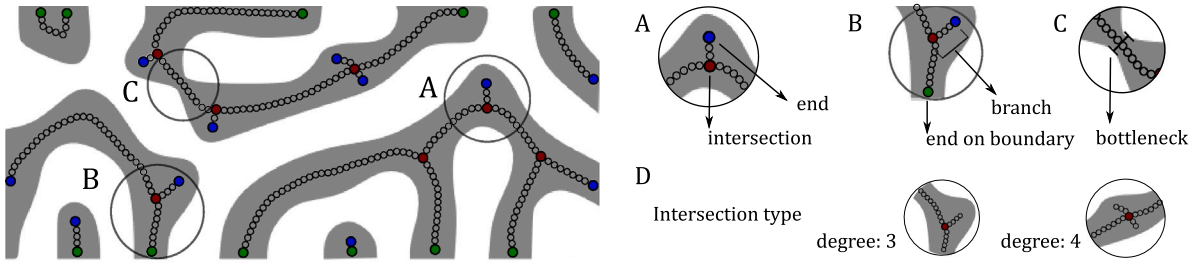


Fig. 2. Selected skeletal descriptors are labeled for a microstructure. The panels include examples of: an intersection and an end pixel (A), the end on boundary and a branch (*B*), bottlenecks in the microstructure (C), and two examples of intersection degree (D).

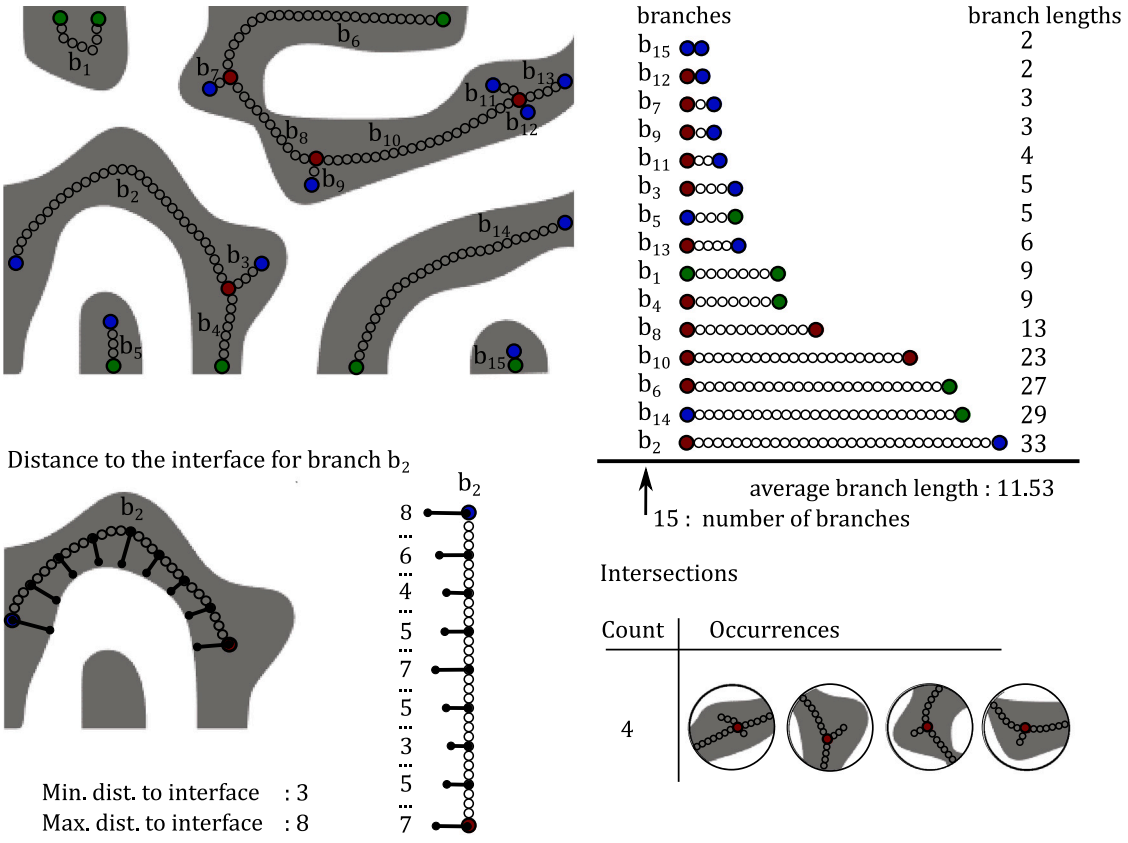


Fig. 3. Illustrations of skeletal segmentation to extract branches, distances to the interface and types of intersections. The top panel (left) is an example of segmentation of skeletal pixels at the intersections (red) and ends (blue and green) into branches (right). The bottom left panel depicts an example to calculate the distance to the interface from branch b_2 and the bottom right corner summarizes the intersections.

the examples of intersection skeletal pixel (marked red), end skeletal pixel (marked blue and green), and a branch on the skeleton (marked with open black circles). Panel C depicts a *bottleneck* in the input microstructure. We provide a few examples of how basic methods are leveraged to compute the descriptors below.

Fig. 3 illustrates examples of a descriptor calculation from the segmented skeleton. In this case, the skeleton of phase A (marked gray) is segmented at the intersections and ends into 15 branches, labeled as $(b_1, b_2, \ldots, b_{15})$. Branches are also sorted (see the top right panel of Fig. 3) based on the branch lengths. Finally, the bottom left panel of that figure demonstrates the bottleneck computation. For each skeletal pixel on the branch (here branch b_2), the distances to the interface are looked up from D_F (see Section 2.1) and stored in an array. The minimum distance is equivalent to the bottleneck along a given branch. This is an example of seamless calculation of the descriptors from our skeletal representation. The bottleneck identification requires combining information from the distance map and the skeleton followed by the filtering operation. Finally, the bottom right panel of Fig. 3 depicts the table with the intersections (including their degree - or the number of branches originating at a given intersection). For the microstructure in Fig. 3, four intersections are identified.

With examples of the descriptors, we now formally define the vector of the skeletal descriptors used in this work. For more involved descriptors, we illustrate the definitions by providing the value of descriptor. We use the italic font and refer to microstructures in Fig. 3 or Table 1.

- d_1 Fraction of skeletal pixels $f_S = |S|/|M_A|$ is the ratio of number of skeletal pixels of a given phase (say, A) $|S|$ to the number of pixels of the same phase of a microstructure, $|M_A|$.
- d_2 Number of skeletal end pixels $|S_E|$ is the number of end/terminating pixels on the skeleton, S_E as defined in Section 2.1. In Fig. 3 the end skeletal pixels are marked with blue and green circles (green circles correspond to the ends that lie on the top and the bottom boundary, and the blue correspond

Table 1
Example microstructures annotated with the descriptors. Top panel illustrates the distribution of phases with corresponding skeleton for each phase: blue skeleton for the black phase (A) and magenta skeleton for the white phase (B). The table contains ten descriptors for each phase A and B - phase is marked in the second column. Descriptors quantifying distances (d_6, d_8, d_9 and d_{10}) are given in unit of pixels.

Descriptor				
d_1: Frac. of skel pixels	A	0.041	0.016	0.07
	B	0.035	0.078	0.07
d_2: Number of ends	A	15	24	30
	B	18	25	31
d_3: Number of intersections	A	7	0	13
	B	2	31	9
d_4: Number of ends on bndr.	A	13	14	16
	B	16	21	18
d_5: Number of branches	A	9	13	26
	B	13	32	36
d_6: Avg. branch length	A	94.11	18.31	58.85
	B	54.85	61.06	39.94
d_7: Number of cycles	A	1	0	1
	B	1	14	0
d_8: Max dist. to interface	A	18.5	15.5	11.36
	B	19.54	17.68	12.5
d_9: Min. dist. to interface	A	1.5	0.5	1.5
	B	5.12	2.5	2.12
d_{10}: Avg. dist. to interface	A	11.19	9.19	6.14
	B	11.77	8.00	6.06

to the bulk ends). *Note that $|S_E|$ for phase A of microstructure in Fig. 3 is* 21.

d_3 Number of skeletal intersection pixels on the skeleton $|S_J|$. The set S_J is defined in Section 2.1 and is marked as red circles in Fig. 3. *Note that $|S_J|$ for the microstructure in Fig. 3 is* 4.

d_4 Number of end skeletal pixels on the boundary is the number of ends that lie on the top and the bottom boundary of the microstructure. In Fig. 3, end skeletal pixels on the boundary are denoted with green circles. *Note that the phase A in Fig. 3 has* 7 *boundary end points on that skeleton.*

d_5 Number of branches $n_B = |S_B|$ is the total number of branches in the skeleton. The branches are extracted after skeletal segmentation at intersections and ends (defined in Section 2.1) procedure. In Fig. 3, the branches are listed according to their lengths in the top right panel. *Note that this descriptor for the microstructure in Fig. 3 is* 15.

d_6 Average branch length $L_f = \sum_i^{n_B} |b_i|/n_B$ is the average of branch lengths in the skeleton.[5] For the branches extracted in the previous descriptor, number of pixels in each branch is calculated as the cardinality of set $|b_i|$ in Eq. (5). In Fig. 3, the branch lengths are listed on the right panel. *Note that this descriptor is the average of that list, which is 11.53 pixels per branch.*

d_7 Number of cycles C is the count of cycles (closed loops) in the skeleton (see Section 2.1). In Table 1, the number of cycles for the second microstructure is listed in row d_7. *Note that phase A in Fig. 3 has no cycles; however, phase B has* 14 *cycles.*

d_8 Maximum distance to the interface D_{max} is the maximum width of the domains along the skeleton of phase A. The bottom left panel of Fig. 3 depicts the process. *Note for branch b_2 in Fig. 3 the maximum distance to the interface is* 8 *pixels.*

d_9 Minimum distance to the interface D_{min} corresponds to the bottleneck or the minimum width of the domains along the skeleton in a microstructure. The value corresponds to half of the width of bottlenecks in the microstructure domain (see panel C in Fig. 2). *Note, for the branch b_2 in Fig. 3, the value of this descriptor is* 3.

d_{10} Average distance to the interface $\langle D \rangle$ is the average domain width of a microstructure. The bottom panel in Fig. 3 depicts the distances to the interface for one branch; however, this value represents the average distance for the entire microstructure.

Using the machine learning language, the vector of the descriptors becomes the featurized microstructures. And, it is compatible with the machine learning methods. The feature vector size is relatively small compared to the typical pixel-based representation of the microstructure (100^2). In this work, the vector captures skeletal information about the microstructures. Moreover, the descriptors are generic with minimal input of the target application. However, the utility of the feature vector needs to be assessed in the context of the targeted application with two questions of importance: is the vector of features sufficient to build the data-driven model with good accuracy? And can the vector be further reduced to distill the key (or salient) features controlling the properties of interest? The following subsection defines the data-driven model of structure–property map and describes the associated feature selection method used to answer the two questions above.

2.3. Feature selection methods and structure–property map

We use a feature selection technique — mRMR (Maximum Relevance–Minimum Redundancy) to identify a small set of *salient* features that capture the most variance in the property under study [19]. This is relatively simple method that is decoupled from the model construction and relies on the correlation between descriptors and property of interest. The method relies on the importance score that is computed for each feature and capture the correlation between the feature under consideration and the property (relevance) as well as the already selected features (redundancy). Therefore, a high importance score indicates high relevance with respect to the property and small redundancy among the selected features. At each iteration, the method selects one feature with maximum relevance with respect to the property and minimum redundancy with respect to the chosen features at previous iterations. As an outcome, the vector of features is reordered based on the importance score, and the gap (or significant decrease of score between consecutive features) is used to choose the features' subset of high importance. Formally, the selected features are defined as $\tilde{d}$, where

[5] The length is given in pixels.

the number of descriptors, $\widetilde{n} \ll 2n$ is determined based on the gap in the importance score — see the results section for the example.

Given the subset of feature, in this work, we define an SP map as a function f that maps $\tilde{d}$ to a property of interest:

$$P = f(\tilde{d}) = \sum_{i=1}^{\widetilde{n}} \omega_i \widetilde{d}_i + \epsilon \tag{9}$$

where ω_i are the weights of each descriptor $\widetilde{d}_i$ in the model, $\widetilde{n}$ is the number of salient features and ϵ is the noise in the map. The map assumes the linear dependence between salient features and our property. Results from other trained other models (e.g., higher order polynomials and random forests) showed similar accuracy as these reported in Section 3. Hence, we use the simplest model to demonstrate the flexibility of generic descriptors to build data-driven models with high accuracy.

3. Results

In this section, we provide the technical details of the framework. Next, we showcase descriptor calculations for a representative set of microstructures and construct an SP map to show the utility of the descriptor set.

3.1. Technical details

The framework for skeletonization and descriptor calculation is implemented in C/C++ using the `boost` library for the graph-based operations [20]. Feature selection is performed on `MatLab R2021` using the Statistics and Machine Learning Toolbox. The regression analysis is performed using `numpy` [21] and `scikit-learn` [22] libraries. The runtime of descriptor calculation for one microstructure (40,501 pixels) is under one second on a desktop computer with a 2.3-GHz dual-core Intel Core i5 processor and 8 GB of RAM. The low computational time affirms that the descriptors are computationally manageable.

3.2. Data

The framework is tested on an open-source dataset containing the process–structure–property data of OPVs [23]. The active layer of organic photovoltaics (OPV) is manufactured using thin-film deposition technologies from organic blends. The microstructure of the active layer is known to affect the properties of OPV devices and can be controlled by tailoring the manufacturing settings. Hence, there is a need to establish and optimize PSP maps. In this paper, the focus is on establishing a structure–property relationship using the data-driven approach. The dataset consists of 1708 microstructures generated using a Cahn–Hilliard equation solver [24]. The Cahn–Hilliard solver models the evolution of a binary blend undergoing thermal annealing — one of the manufacturing techniques used in OPV. The binary mixture consists of an electron donor material and an electron acceptor material. Each microstructure is of size 400 nm × 100 nm with the discretization of 400 × 100 elements. Three example microstructures are included on the top panel of Table 1. All microstructures (P3HT:PCBM blend[6] mixture: a well-studied material system) consist of two phases, where one phase serves as an efficient electron-donor and the other phase serves as an efficient electron-acceptor.

Each microstructure in the dataset is annotated with one property (i.e., the target property) - short circuit current, J_{sc}. To predict the property (J_{sc}), a computational framework is employed which models the OPV device physics of the active layer in OPV device [25–27]. The model uses a finite element-based solution strategy to the excitonic drift–diffusion equations. It solves the spatial distribution of excitons, electrons, holes, and the electric potential across the domain. The material parameters corresponding to a P3HT: PCBM system are provided in our prior work [25]. More details on the data generation and the computational model of OPV device physics can be found in our prior paper [28].

Each microstructure is also annotated with twenty skeletal descriptors. For the SP models – Section 3.6 – the dataset is split into training and testing sets of 80% and 20%, respectively. Additionally, the training set is split into five subsets for five-fold cross-validation when specified.

3.3. Descriptor calculation

Descriptors for three example microstructures are listed in Table 1. The microstructures and the skeletons: blue skeleton for phase A (black) and magenta skeleton for phase B (white) are shown in the top panel of Table 1. The descriptors from vector d_M consist of 20 descriptors, 10 per phase. For each descriptor, two values for each phase are provided in sub rows A and B, respectively.

Three diverse microstructures are selected to demonstrate the capabilities of the descriptors. The first microstructure consists of thick domains of both phases with similar topology. In contrast, the phases of the second microstructure have very different topological characteristics. Phase A (black) has dispersed droplets, and phase B (white) has a connected structure. The third microstructure has a connected structure with thin domains for both phases.

The skeletal descriptors capture the above characteristics. For example, d_3 (number of intersections) for phase A in the second microstructure is much smaller ($d_3 = 0$) than that of the third microstructure ($d_3 = 13$). The first microstructure has $d_3 = 7$ for phase A, and the third microstructure has $d_3 = 13$ for phase A. The lower number of intersections and branches is a signature of branch connectivity in a structure. The third microstructure has the highest number of branches ($d_5 = 26$ for phase A and $d_5 = 36$ for phase B). The first microstructure has a balanced number of branches for both the phases; $d_5 = 9$ for phase A, and $d_5 = 13$ for phase B; on the contrary, the d_5 values for the second microstructure are different. Phase A only has 13 branches, while phase B has 32 branches. Combining these descriptors accentuates that only one phase has connected domains (here, phase B), while phase A has many small domains embedded inside phase B.

Descriptor d_7 (number of cycles) captures the topological characteristics of the phases. Among the three example microstructures, the first and the third are similar. Descriptor d_7 reflects these characteristics as the values of the first and the third microstructures are alike (either 0 or 1). For the second microstructure, the value for phase A is 0, and for phase B is 14. A clear difference in the number of cycles is an indication that phase A and phase B of the second microstructure have different topologies. Finally, the last three descriptors provide quantitative means to determine the domain widths and bottlenecks in the microstructure. The first microstructure has the thickest domains, and the third has the thinnest. The d_{10} (average distance to the interface) of the third microstructure is the minimum (6 pixels). Phase B of the second microstructure has the smallest bottleneck (d_9) of width at 0.5 pixels.

Microstructure featurization allows determining the distributions of various descriptors in datasets. The four panels of Fig. 4 depict the distributions of four descriptors for phase B (white): fraction of skeletal pixels (d_1), number of end skeletal pixels (d_2), average distance to the interface (d_{10}), and number of cycles (d_7). The inserts in each of the distributions include two microstructures characterized by extreme values in the histograms. The histograms of descriptors may be considered as a data summary and be used for sampling from the larger data to choose the diverse subset, among other applications.

For example, the top left panel depicts a histogram of descriptor d_1 - a fraction of skeletal pixels. The most common value for a fraction

[6] P3HT:PCBM is poly(3- hexylthiophene) and 1-(3-methoxycarbonyl)-propyl-1-phenyl-[6,6]C_{61}.

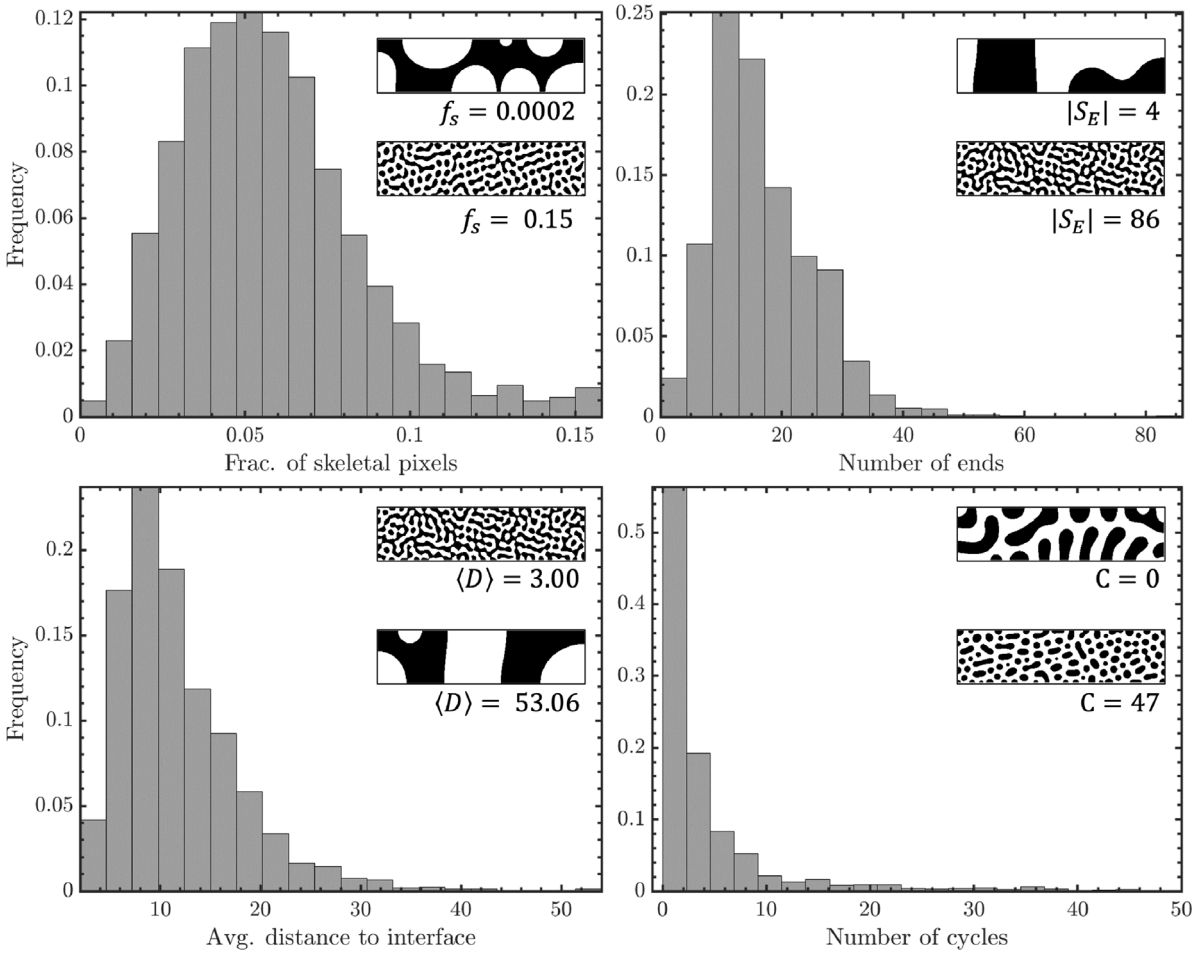


Fig. 4. Four histograms of descriptors for phase B (denoted by white color) in the data set. Four descriptors are: d_1, d_2, d_{10} and d_7. Each panel includes the distribution of descriptor value where frequency is given as a fraction of the number of samples. In each panel two inserts depict microstructures with extreme descriptor values: the lowest and highest values.

of skeletal pixels for phase B is 0.05. However, this descriptor range is from 0.0002 to 0.15. The inserts show that the smallest value of d_1 corresponds to a structure with thick droplet domains, and the largest value corresponds to a structure with connected thin domains.

The top right panel of the figure depicts the histogram of the number of ends on the skeleton. This descriptor informs about the branching and the type of domains in the microstructure. For example, phase B of microstructures with less than 5 skeletal ends corresponds to data points with thick domains connected to the top and bottom of the microstructure. The microstructures with a maximum number of ends are typical of thin domains with a less connected structure. The histogram of the average distance to the interface (d_{10}) is plotted in the bottom left panel of Fig. 4. These values indicate the average domain sizes of phase B in the dataset. The microstructure with the lowest value of this descriptor (3.00) is included in the inserts. As expected, the microstructure is of thin domains, and the microstructure with the highest value (53.06) is of thick domains. Finally, the histogram of number of cycles is included in the bottom right panel of the figure. Most of the microstructures analyzed have less than five cycles. The highest value is 47 (denoted in white in the insert included), the complimentary phase of a droplet microstructure.

3.4. Correlation between descriptors

Given a sufficiently large data set, the correlations between descriptors can be used to understand and remove the redundant descriptors from the initial pool of descriptors. This type of analysis is also of importance if some properties of a material are challenging to quantify. If the correlation is challenging to predict, the descriptor can be inferred from a calculated descriptor combination. For instance, the value of tortuosity may be challenging to determine from two-dimensional imaging. But the correlation with other descriptors exists (for example, porosity), and then the tortuosity can be imputed. In this work, we use the descriptor correlation studies to understand the results of feature selection for SP map construction — see Section 3.5. Specifically, we calculate the Pearson's correlation coefficients (r) [29] within the descriptor set and between the descriptors and the property.

The correlation matrix between the skeletal descriptors (d_M) and the property - J_{SC} is determined and plotted in Fig. 5. The matrix is representated as a heat map and contains the Pearson's correlation coefficient between every pair of descriptors and the property. In the most left panel of Fig. 5, the first row and the first column contain the correlation between the property P and all descriptors. The remaining rows and columns contain the correlations between all 10 descriptors (all-to-all correlation coefficients). Each descriptor is listed twice for each phase in the microstructure. The high correlations are marked with deep red or deep blue for positive and negative correlation, respectively.

For the dataset, as shown in Fig. 5, two groups of correlations with high values can be identified. The first group corresponds to the first five descriptors (d_1, d_2, d_3, d_4 and d_5) and the second group corresponds to the last three descriptors (d_8, d_9 and d_{10}). The first five descriptors are skeletal features. They correspond to the fraction of skeletal pixels, number of ends, number of ends on the boundary, number of intersections, and number of branches. The last three descriptors correspond

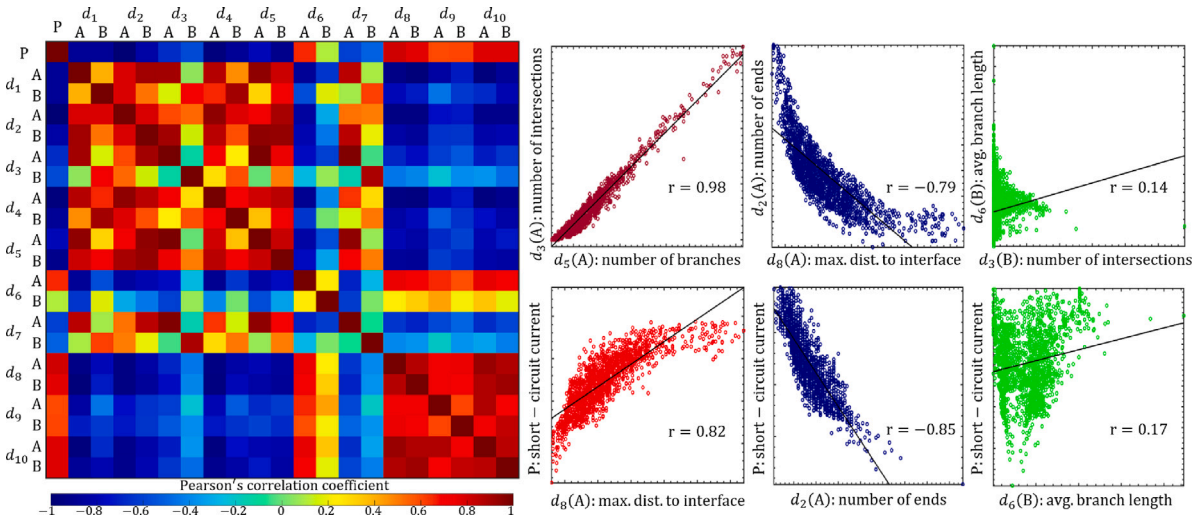


Fig. 5. Correlation matrix for 20 descriptors (10 for each phase) and the property — short circuit current (J_{SC}). The plots on the right are examples of positive, negative, and poor correlations with the property (top row) and descriptors (bottom row).

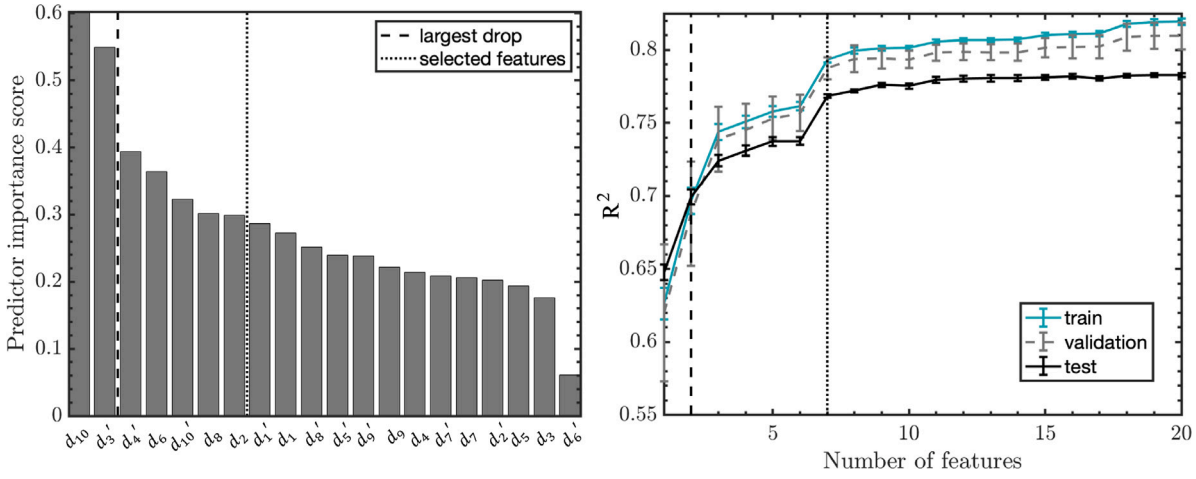


Fig. 6. Feature importance plots for mRMR feature selection method. The left panel shows the rankings of the features, and the right panel depicts the accuracy of the models for increasing number of features. Three curves correspond to the mean accuracy of training, validation, and testing, where error bars represent the standard deviation of R^2 across the five folds.

to the maximum, minimum, and average distance to the interface. To visualize the correlation trends within the descriptor set, we plot the values in the top right panel of the figure. Each point corresponds to one microstructure in the dataset in the scatter plots. For example, descriptors d_5 (number of branches) and d_3 (number of intersections) of phase A are highly correlated with the correlation coefficient of 0.98, as shown in the first scatter plot of Fig. 5. The high correlation is predictable as the intersections determine the branches. An unexpected, correlated descriptor pair is d_8 (number of ends) and d_2 (maximum distance to the interface) of phase A as shown in the second scatter plot of the figure. The correlation coefficient is −0.79, and the scatter plot demonstrates that these descriptors are inversely proportional. This correlation is less apparent and means that microstructures with a high maximum distance to the interface are characterized by the low number of ends for this dataset. Descriptors d_6 (average branch length) and d_7 (number of cycles) are observed to be uncorrelated with other descriptors in the correlation matrix. We use these results in the next section to select/eliminate features for the prediction model.

3.5. Correlations with the property and feature selection

In this section, we report the correlations between the descriptors and the property to aid the development of SP maps. We return to Fig. 5 for the correlation matrix and focus on the first row denoted as P. The property is negatively correlated with descriptors d_1, d_2, d_3, d_4, d_5 and d_7, and positively correlated to d_8, d_9 and d_{10}. Descriptor d_6, especially, phase B is poorly correlated with P. The example scatter plots are depicted in the bottom row of the right panel in Fig. 5. The high correlation coefficient of 0.82 is observed between d_8 (max. distance to the interface) of phase A and property P (J_{SC}) (the left panel in the bottom row). The correlation coefficient captures the correlation between these two variables for microstructure in our dataset. The microstructures with low d_8 exhibit poor performance and low P. The microstructure with high d_8 is more suited for organic electronics. The negative correlation coefficient of −0.85 is observed between d_2 (number of ends) and property P. The microstructures with high performance are characterized by the low number of d_2. Both descriptors are relevant for property predictions.

The concepts of redundancy and relevance are core to the mRMR method of feature selection — explained in Section 2.3. mRMR is performed on the data with 20 descriptors (listed in Table 1) to identify the most relevant, non-redundant features that can explain the property to the maximum extent. The method returns the descriptors/features ranked according to the importance score plotted in the left panel of Fig. 6. Phase A (denoted by d_i) corresponds to the donor, and phase

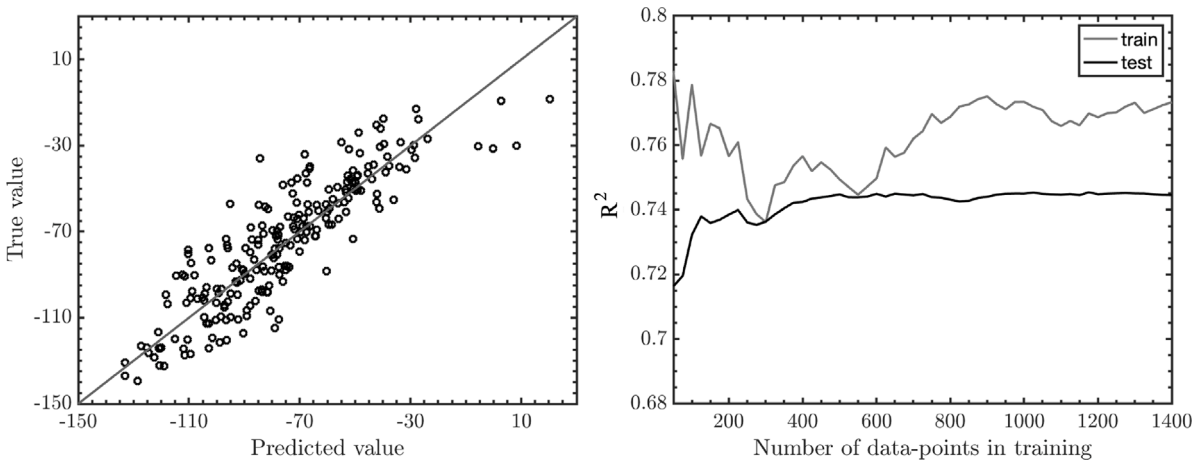


Fig. 7. Left: The accuracy of the model comparing the predicted vs. the true value of J_{SC} for the test data. Each point in this panel correspond to one microstructure in the testing data. Right: learning curve for increasing number of data-points in the training data.

B (denoted by d_i') to the acceptor. The method assigns the highest score to d_{10} (average distance to the interface) of phase A, followed by d_3 (number of intersections) for phase B (d_3'). The descriptor d_{10} displays a strong negative correlation with the property (Fig. 5), and d_3 displays a strong positive correlation. The correlation coefficient of d_{10} for phase A and d_3 for phase B is weak (≈ 0.2). The dip after the second descriptor in Fig. 6 signifies the selection of two descriptors to construct the prediction model.

3.6. Structure–property maps in organic solar cells

In this subsection, we report the accuracy of SP models for selected descriptors and corresponding models. Given the ranking of the features based on the importance score from the mRMR method, data-driven models of SP are built for an increasing number of skeletal features and increasing number of data points.

The right panel of Fig. 6 depicts the accuracy of models with the increasing number of features. The coefficient of determination R^2 [30] is computed as a measure of model accuracy with three types of accuracy reported: accuracy of training, validation, and testing. For each type, the curve depicts the accuracy of the models for an increasing number of features. Each curve includes the mean and standard deviation of R^2 values across the five-folds. The error bars represent the standard deviation of the accuracy across five-folds.

The importance score analysis presented in the previous subsection indicated the first two descriptors to have the highest importance scores. This observation is reflected in the highest increase in the model's accuracy. Further adding up to seven descriptors increases the train, validation, and validation accuracy of the models up to 0.80, after which the curves saturate. The seven features are optimal for SP predictions due to the balance between complexity and accuracy.[7] The seven features include average distance to the interface (donor), number of intersections (acceptor), number of ends on the boundary (acceptor), average branch length (donor), the average distance to the interface(acceptor), maximum distance to the interface (donor), and number of ends (donor). It is important to note that mRMR and model construction is independent; however, they both provide consistent results. In the next section, we built the structure–property maps with seven features and report on its performance.

To additionally evaluate the model performance, we plot the correlation between predicted and true values of the property J_{SC} (left panel) and the learning curve (right panel) of Fig. 7. The left panel depicts a so-called parity plot that compares the predicted value against the true value (J_{SC}). The predicted values of our property are calculated using the model with seven descriptors ranked by mRMR. The true values of our property are computed using the computational model (see Section 3.2) and are considered true values. For reference, we also provide a diagonal line representing the ideal curve where the prediction is equal to the true value (see the gray line in the plot). Points that lie close to that line indicate a low error, with the predicted value very close to the true value. In contrast, the points located further away from the diagonal have a high error. The results included in the parity plot show the good distribution of the error in our model. Most points are distributed near the diagonal without clear outliers affecting the R^2.

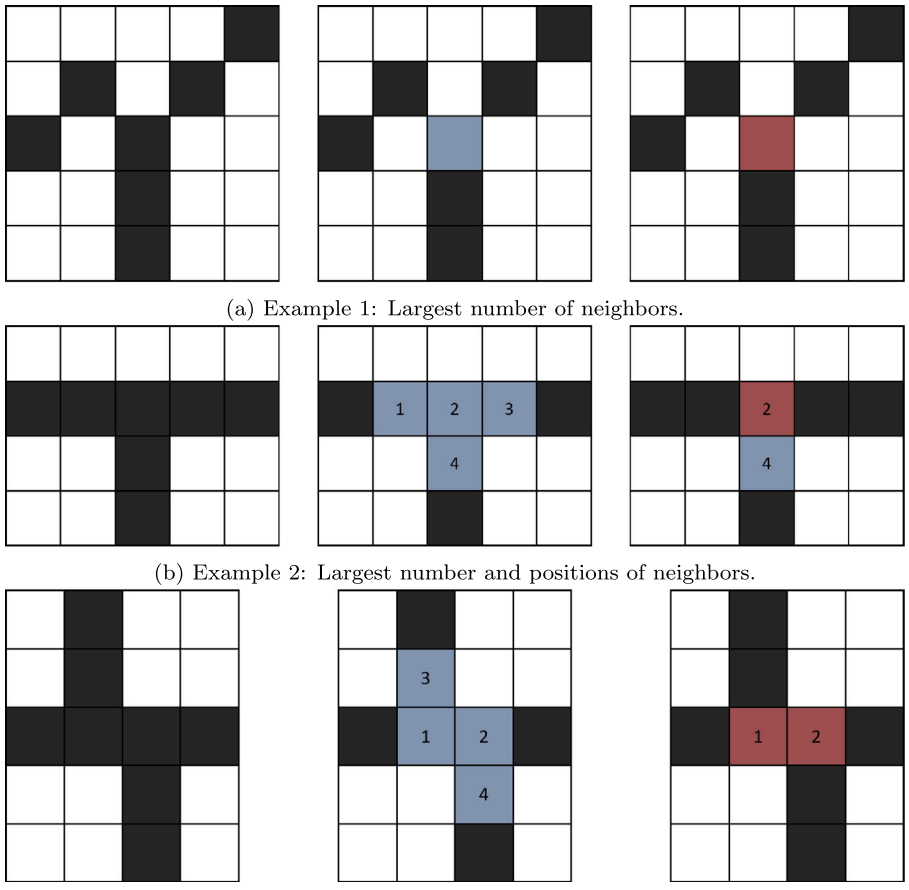
Finally, we plot a learning curve that depicts the model's training and testing accuracy to check the model's generalization. The right panel of Fig. 7 plots the R^2 value for increasing training data size (from 100 points to 1400) and seven features. The size of the testing set is kept constant at 300 points. When the smaller data set is used for training, the R^2 of the model on the training set (gray) is higher than on the testing set (black). As the training size increases, the training R^2 stabilizes at ≈ 0.77, and for test data, the accuracy also increases and stabilizes at 0.74. This indicates that the model is free from high variance and high bias. This demonstrates the efficacy of the proposed featurization framework and the skeletal representation.

4. Limitations and potential extensions

Our method is generalizable to other types of microstructures. However, the underlying assumption is that the skeleton can be computed without artifacts and captures some information about the type of microstructure being analyzed. In this work, the focus is on two-phase microstructure. But the multi-phase microstructures can also be featurized using skeletal representation. In such a case, the skeletonization and descriptor definition needs to be done for each phase.

Moreover, the introduced methodology is independent of dimensionality and can be used to quantify both two and three-dimensional microstructures. All descriptors can be expanded for three-dimensional datasets. Nevertheless, on the technical level, the anticipated challenge is related to the skeletonization algorithm in three dimensions. Although several implementations of skeletonization exist [15,31,32], these are known to have limitations in delivering smooth, noise-free, and centered skeletons [33].

[7] These results are consistent with models built using other feature selection methods like Random Forests (not shown here).



(a) Example 1: Largest number of neighbors.

(b) Example 2: Largest number and positions of neighbors.

(c) Example 3: Two true intersections in the same local neighborhood.

Fig. 8. Three examples of identifying intersections on a skeleton. The black squares represent the skeleton of the input microstructure. The potential intersections are marked with blue squares and, the final intersections are marked with red squares.

5. Conclusion

This paper introduced the framework to compute the skeletal descriptors seamlessly computed directly from our skeletal representation. Seamless calculation of descriptors is one of the advantages of the skeletal representation (apart from the compact representation demonstrated before [12]). Here, we defined a suite of skeletal descriptors that captured the topological and morphological characteristics of microstructure. We show that this descriptor-based representation is sufficient to capture the salient features of the microstructure correlated with the short circuit current of OPVs. The simple regression model of the structure–property relationship was built with only seven descriptors with an accuracy of $R^2 = 0.77$. The proposed descriptors are generic, computationally manageable and compatible with machine learning methods.

CRediT authorship contribution statement

Devyani Jivani: Methodology, Investigation, Data curation, Visualization, Methodology, Software, Validation, Writing – review & editing. **Olga Wodo:** Conceptualization, Methodology, Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by National Science Foundation, United States (1906344). Authors also acknowledge the support provided by the Center for Computational Research at the University at Buffalo.

Appendix

Procedure to determine the intersections

As a reminder, an intersection is a skeletal pixel in the set S that lies on at least three branches — see Fig. 2. To determine if a skeletal pixel is an intersection or not, two steps are followed:

1. The first step is to identify the potential intersections. For every pixel $s_i \in S$, N_8 is determined, and the skeletal pixels are counted. If $N_8(s_i)$ has at least 3 other skeletal pixels, s_i is labeled as a potential intersection and added to the set of potential intersections: J.
2. The second step is to identify the actual intersections from the potential intersection set J. For each potential intersection $j_i \in J$, the local neighborhood $N_8(j_i)$ is checked for other potential intersection pixels. If the neighborhood contains more than one potential intersection:

(a) Potential skeletal pixel with the highest number of potential intersection neighbors has the highest precedence.
(b) For potential skeletal pixels with same number of other potential intersection neighbors, the counting is repeated but in the smaller neighborhood — the von Neumann neighborhood. A pixel with the highest number of other potential intersections in $N_4(j_i)$ is marked as the actual intersection.
(c) If more than one pixel has the same number and type of potential intersection neighbors, one among all is selected as the true intersection. However, skeletal segmentation is performed at all the pixels.

In Fig. 8, we provide three examples of typical pixel intersections on a skeleton to illustrate the protocol of intersection determination. In each example, the skeleton is marked with black pixel (the first column). To identify the *potential intersections*, the local neighborhood of each pixel of the skeleton is examined (the second column). The final intersections are marked red (the second column).

The first example is straightforward as only one skeletal pixel has more than two skeletal pixels as neighbors (marked blue). Since there are no other potential intersections in the local neighborhood, in the second step (third column) this pixel is immediately classified as an intersection (marked red).

In the second example, multiple skeletal pixels have more than two skeletal neighbors in their N_8 neighborhoods. These pixels are marked in blue as potential intersections and indexed 1 to 4 (second column). Pixels with index 1 and index 3 have two neighbors marked as potential intersections, while pixels with index 2 and index 4 have three potential intersections. Hence, pixels 1 and 3 are excluded from the further steps. From pixels with index 2 and 4, the neighborhood is then narrowed down to the von Neumann neighborhood and the counting is repeated. Pixel with index 2 has three potential intersections in the von Neumann neighborhood, hence it is selected as a true intersection. For comparison, the pixel with index 4 has only one potential intersection in the von Neumann neighborhood.

The last example illustrates a more complex case with a symmetrical neighborhood and the same number of potential intersections both in Moore's and von Neumann's neighborhood. In Fig. 8(c) pixels with indices 1 and 2 have the same number of neighbors, and hence, both are classified as true intersections (marked red). In this paper, we keep both intersection skeletal pixels to segment the skeleton and ensure the clear segmentation of the skeleton into branches. For this small example, the skeleton will be segmented into 4 branches. However, for counting the number of intersections, we choose one intersection as the representative intersection.

References

[1] O. Wodo, S. Broderick, K. Rajan, Microstructural informatics for accelerating the discovery of processing–microstructure–property relationships, MRS Bull. 41 (8) (2016) 603–609.
[2] H. Xu, R. Liu, A. Choudhary, W. Chen, A machine learning-based design representation method for designing heterogeneous microstructures, J. Mech. Des. (ISSN: 1050-0472) 137 (5) (2015).
[3] P. Fratzl, R. Weinkamer, Nature's hierarchical materials, Prog. Mater. Sci. 52 (8) (2007) 1263–1334.
[4] B.S.S. Pokuri, S. Ghosal, A. Kokate, S. Sarkar, B. Ganapathysubramanian, Interpretable deep learning for guided microstructure-property explorations in photovoltaics, Npj Comput. Mater. 5 (1) (2019) 1–11.
[5] A. Zheng, A. Casari, Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists, " O'Reilly Media, Inc.", 2018.
[6] R. Bostanabad, Y. Zhang, X. Li, T. Kearney, L.C. Brinson, D.W. Apley, W.K. Liu, W. Chen, Computational microstructure characterization and reconstruction: Review of the state-of-the-art techniques, Prog. Mater. Sci. 95 (2018) 1–41.
[7] H. Liu, B. Yucel, D. Wheeler, B. Ganapathysubramanian, S.R. Kalidindi, O. Wodo, How important is microstructural feature selection for data-driven structure-property mapping? MRS Commun. (2022) 1–9.
[8] Y. Bengio, A. Courville, P. Vincent, Representation learning: A review and new perspectives, IEEE Trans. Pattern Anal. Mach. Intell. 35 (8) (2013) 1798–1828.
[9] O. Wodo, S. Tirthapura, S. Chaudhary, B. Ganapathysubramanian, A graph-based formulation for computational characterization of bulk heterojunction morphology, Organ. Electron. 13 (6) (2012) 1105–1113.
[10] B.L. DeCost, E.A. Holm, A computer vision approach for automated analysis and classification of microstructural image data, Comput. Mater. Sci. 110 (2015) 126–133.
[11] S. Yu, Y. Zhang, C. Wang, W.-k. Lee, B. Dong, T.W. Odom, C. Sun, W. Chen, Characterization and design of functional quasi-random nanostructured materials using spectral density function, J. Mech. Des. 139 (7) (2017) 071401.
[12] D. Jivani, R. Rai, O. Wodo, Skeletal-based microstructure representation and convolution reconstruction, Comput. Mater. Sci. 193 (2021) 110409.
[13] R. Kimmel, D. Shaked, N. Kiryati, A.M. Bruckstein, Skeletonization via distance maps and level sets, Comput. Vis. Image Underst. 62 (3) (1995) 382–391.
[14] H.I. Choi, C.Y. Han, Chapter 19 - the medial axis transform, in: G. Farin, J. Hoschek, M.-S. Kim (Eds.), Handbook of Computer Aided Geometric Design, North-Holland, Amsterdam, 2002, pp. 451–471.
[15] T. Lee, R. Kashyap, C. Chu, Building skeleton models via 3-D medial surface axis thinning algorithms, CVGIP: Graph. Models Image Process. 56 (6) (1994) 462–478.
[16] P.K. Saha, G. Borgefors, G.S. di Baja, A survey on skeletonization algorithms and their applications, Pattern Recognit. Lett. 76 (2016) 3–12.
[17] D.A. Zaitsev, A generalized neighborhood for cellular automata, Theoret. Comput. Sci. 666 (2017) 21–35.
[18] T. Toffoli, N. Margolus, Cellular Automata Machines: A New Environment for Modeling, MIT Press, Cambridge, MA, USA, 1987.
[19] Z. Zhao, R. Anand, M. Wang, Maximum relevance and minimum redundancy feature selection methods for a marketing machine learning platform, in: 2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA), 2019, pp. 442–452.
[20] J. Siek, L.-Q. Lee, A. Lumsdaine, Boost library, 2000, http://www.boost.org/libs/graph/.
[21] C.R. Harris, K.J. Millman, S.J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N.J. Smith, R. Kern, M. Picus, S. Hoyer, M.H. van Kerkwijk, M. Brett, A. Haldane, J.F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, T.E. Oliphant, Array programming with NumPy, Nature 585 (7825) (2020) 357–362.
[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[23] O. Wodo, J. Zola, B.S.S. Pokuri, P. Du, B. Ganapathysubramanian, Process-structure-property map for organic solar cells, 2021, http://dx.doi.org/10.5281/zenodo.5061951.
[24] O. Wodo, B. Ganapathysubramanian, Computationally efficient solution to the cahn–hilliard equation: Adaptive implicit time schemes, mesh sensitivity analysis and the 3D isoperimetric problem, J. Comput. Phys. 230 (15) (2011) 6037–6060.
[25] H.K. Kodali, B. Ganapathysubramanian, Computer simulation of heterogeneous polymer photovoltaic devices, Modelling Simulation Mater. Sci. Eng. 20 (3) (2012) 035015.
[26] H.K. Kodali, B. Ganapathysubramanian, A computational framework to investigate charge transport in heterogeneous organic photovoltaic devices, Comput. Methods Appl. Mech. Engrg. 247 (2012) 113–129.
[27] H.K. Kodali, B. Ganapathysubramanian, Sensitivity analysis of current generation in organic solar cells: comparing bilayer, sawtooth, and bulk heterojunction morphologies, Sol. Energy Mater. Sol. Cells 111 (2013) 66–73.
[28] O. Wodo, J. Zola, B.S.S. Pokuri, P. Du, B. Ganapathysubramanian, Automated, high throughput exploration of process–structure–property relationships using the mapreduce paradigm, Mater. Discov. 1 (2015) 21–28.
[29] J. Benesty, J. Chen, Y. Huang, I. Cohen, Pearson correlation coefficient, in: Noise Reduction in Speech Processing, Springer Berlin Heidelberg, 2009, pp. 1–4.
[30] D. Zhang, A coefficient of determination for generalized linear models, Amer. Statist. 71 (4) (2017) 310–316.
[31] Y. Tsao, K. Fu, A parallel thinning algorithm for 3-D pictures, Comput. Graph. Image Process. 17 (4) (1981) 315–331, http://dx.doi.org/10.1016/0146-664X(81)90011-3, URL https://www.sciencedirect.com/science/article/pii/0146664X81900113.
[32] K. Palágyi, A. Kuba, A parallel 3D 12-subiteration thinning algorithm, Graph. Models Image Process. 61 (4) (1999) 199–221, http://dx.doi.org/10.1006/gmip.1999.0498, URL https://www.sciencedirect.com/science/article/pii/S1077316999904987.
[33] A. Tagliasacchi, T. Delame, M. Spagnuolo, N. Amenta, A. Telea, 3D skeletons: A state-of-the-art report, Comput. Graph. Forum 35 (2) (2016) 573–597, http://dx.doi.org/10.1111/cgf.12865, URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12865.