Extreme Mechanics Letters 57 (2022) 101895

Contents lists available at ScienceDirect

Extreme Mechanics Letters

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

How to see hidden patterns in metamaterials with interpretable machine learning

Zhi Chen [a,*], Alexander Ogren [b], Chiara Daraio [b], L. Catherine Brinson [a], Cynthia Rudin [a]

[a] Duke University, NC 27708, USA
[b] California Institute of Technology, CA 91125, USA

ARTICLE INFO

Article history:
Received 14 July 2022
Received in revised form 29 August 2022
Accepted 12 September 2022
Available online 19 September 2022

Dataset link: https://github.com/zhiCHEN96/interpretable_ml_metamaterials.git

Keywords:
Phononic materials
Frequency band gaps
Interpretable machine learning
Rule-based models

ABSTRACT

Machine learning models can assist with metamaterials design by approximating computationally expensive simulators or solving inverse design problems. However, past work has usually relied on black box deep neural networks, whose reasoning processes are opaque and require enormous datasets that are expensive to obtain. In this work, we develop two novel machine learning approaches to metamaterials discovery that have neither of these disadvantages. These approaches, called *shape-frequency features* and *unit-cell templates*, can discover 2D metamaterials with user-specified frequency band gaps. Our approaches provide logical rule-based conditions on metamaterial unit-cells that allow for interpretable reasoning processes, and generalize well across design spaces of different resolutions. The templates also provide design flexibility where users can almost freely design the fine resolution features of a unit-cell without affecting the user's desired band gap.

© 2022 Elsevier Ltd. All rights reserved.

1. Introduction

Metamaterials are traditionally designed through empirical trial-and-error or intuition [1] or computationally expensive topology optimization [1–5] which often produces designs out of predetermined geometrical building blocks. Recently, machine learning methods (ML) have gain popularity for metamaterial design. For example, machine learning models can be trained to predict material properties from unit cells defined by a finite set of pixels/voxels, and used as faster surrogate models for computationally expensive simulations [6–10]. Using deep generative models and neural network inversion techniques, some recent works [11–17] aim to directly solve the inverse design problem for metamaterials, i.e., generating the designs given the target property. However, the ML models used in these work are usually gigantic black boxes, whose decision processes are hard to understand. This is undesirable for scientific discovery purposes because scientists may also want to gain insights into what geometric features are important for a given target property, such a particular frequency band gap. In addition, these massive models are data hungry and not robust to distributional shift — they usually require a huge simulated dataset that covers most of the design space. These models also tend to perform poorly on datasets they have not seen before, such as unit-cells in a finer resolution space.

In this paper, instead of relying on existing black box approaches, we propose two novel rule-based ML approaches for metamaterial design that have major advantages:

- ***Interpretability:*** The approach allows us to discover interpretable key patterns within unit-cells that are related to a physical property of interest (see Fig. 1**a** and 1**b**). We consider two types of patterns: (i) local patterns called *shape frequency features*, which calculate the occurrence frequency of certain shapes in the unit-cell; (ii) global patterns, called *unit-cell templates*, which look for arrangements of constituent materials in specific regions of the metamaterials' unit-cells. The unit-cell templates are optimized with binary integer programming to find global patterns within unit cells that give the metamaterial a desired property.
- ***Leverages Multi-resolution Properties:*** An important observation underpinning our methodology is that a pattern in the coarser resolution design space also exists in finer resolution design space, with one coarse pixel replaced by many finer pixels. As a result, *if a pattern can robustly characterize the target property at the coarse resolution design space, it will also be predictive at the finer resolution design space*. This leads to computationally-efficient discovery of many valuable metamaterial designs possessing the desired properties. In particular, our method allows us to construct a *scaffold of patterns* that allows interpretable coarse scale information discovered at low resolutions to be reliably transferred to make accurate predictions for high resolution designs (see Fig. 1**c**).

* Corresponding author.
E-mail address: zhi.chen1@duke.edu (Z. Chen).

https://doi.org/10.1016/j.eml.2022.101895
2352-4316/© 2022 Elsevier Ltd. All rights reserved.

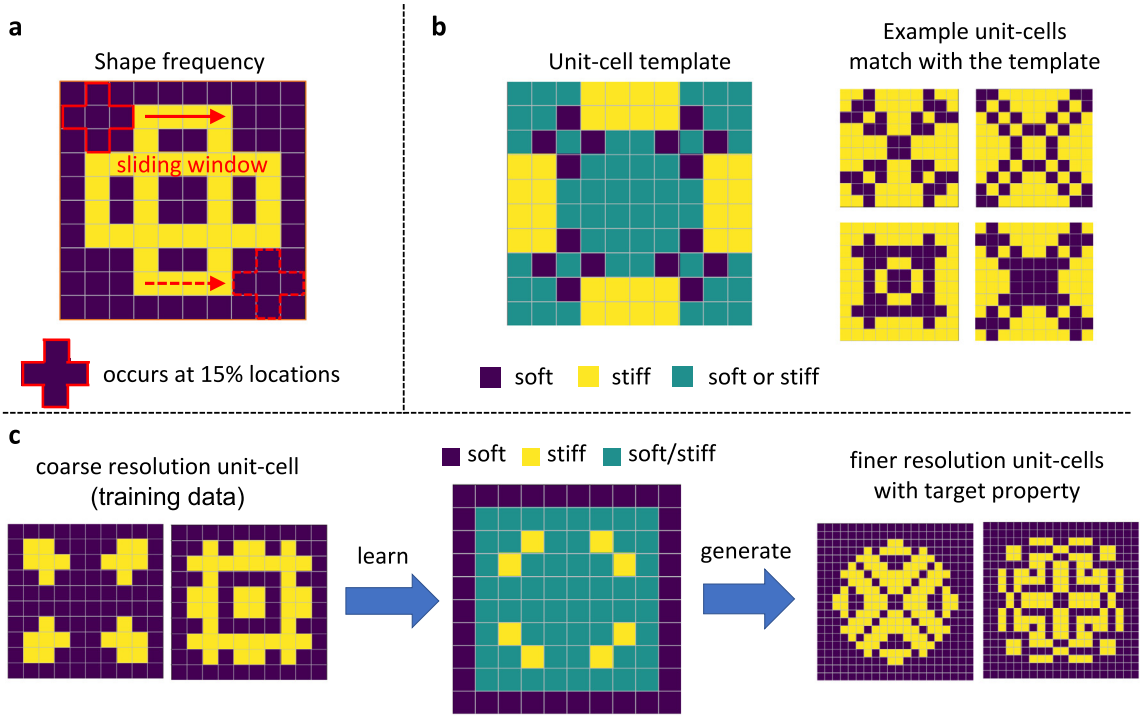

Fig. 1. Examples of interpretable key patterns discovered by the proposed method. **a**. A shape frequency feature (this one is shaped like a "+"). How frequently this shape appears in the unit-cell is a useful predictor of a band gap. **b**. A unit-cell template, which considers specific global patterns in the unit-cell. Here, regardless of whether we place stiff or soft materials at each green pixel in the unit-cell, as long as the stiff and soft materials are in the positions defined by the template in yellow and purple, there will be a band gap within the user's desired range. **c**. The patterns are learned from coarse resolution training data but can be robustly transferred to finer resolution, and generate fine resolution unit-cells with the target property. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

- ***Flexible Metamaterial Designs:*** Our unit-cell templates (e.g., the one in Fig. 1**b**) enables flexibility in unit-cell designs at any resolution. Simply, unit-cell templates specify regions where one can almost freely design unit-cell features without changing the target band gap property (e.g., in Fig. 1**b** constituent phases can be arranged at will in the green regions determined by our algorithm). Such flexibility in design might be useful to satisfy practicality constraints such as connectivity or other design constraints such as overall stiffness.

Section 2 discusses related works on ML approaches for metamaterial designs. In Section 3, we introduce the problem setting. In Section 4, we provide the proposed methods and explain how they deal with the four core objectives of data-driven approaches that are important to materials scientists: (a) design-to-property prediction; (b) property-to-design sampling; (c) identify key patterns; (d) transfer to finer resolution. We then evaluate performance of the proposed methods in Section 5 and test them on practical applications. We discuss and conclude in Section 6.

2. Related works

Metamaterials are architected materials with engineered geometrical micro- and meso-structures that can lead to uncommon physical properties. As mentioned in the introduction, many existing works apply machine learning models for designing metamaterials, i.e., assembling constituent materials into metamaterials that have specific physical properties.

Much past work focuses on structure-to-property prediction. Because of the expensive computational cost of numerical simulations, these works train machine learning models (mostly deep learning models) to approximate the simulation results [6–10]. Using these ML models as a fast replacement of the simulator, materials satisfying the design objective can be found more efficiently using rejection sampling, i.e., randomly picking a structure in the design space until it satisfies the design objective. However, given the immense size of the design space, finding materials through rejection sampling can still be computationally inefficient. Therefore, some recent works apply deep generative models and neural network inverse modeling [11–17] to train a more efficient materials sampler, aiming to solve the inverse design problems for metamaterials, i.e., property-to-structure sampling. See [18–20] for reviews on using deep learning methods for metamaterial designs.

Almost all modern existing work on this topic uses black box models like deep neural networks. Such approaches are unable to answer key questions such as "What patterns in a material's design would lead to a specific desirable property?" A link between the specific design and the target property could be useful for further research; i.e., to determine whether there is an agreement with domain knowledge, and if not, to potentially discover new knowledge.

Other work also stresses the importance of interpretability for metamaterial design [14,21,22]. These works are very different from ours, and cannot solve the challenge we want to address. Ma et al. [14] try to make the latent space of deep generative models interpretable. Their interpretable features describe general geometrical information such as size and shape of holes in the material but these properties are not associated with the target property. In contrast, our goal is to find key patterns within unit cells that result in the target property (in our case, a band gap). Elzouka et al. [21] and Zhu et al. [22] also build rule-based models, namely decision trees, but the design problems they are solving are different and much simpler than the problem we try to solve. Specifically, their materials are described by several continuous features, e.g., thickness of the materials, while we work

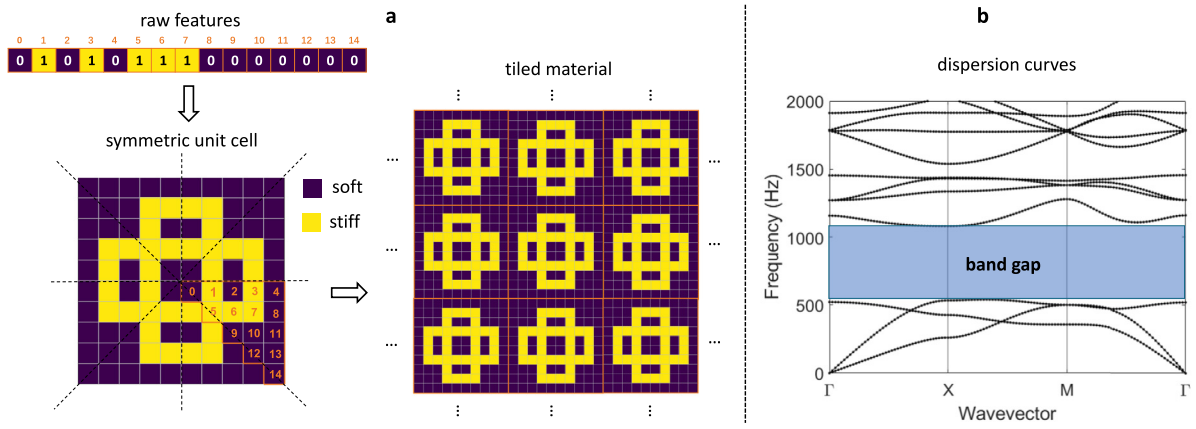

Fig. 2. Input and target of our metamaterial design problem. **a.** 2-D phononic material constructed with the raw features. Upper Left: raw 15 dimensional input feature vector. Lower left: the feature vector defines the triangle in the lower right of the unit-cell. The triangle is copied using lines of symmetry to define the full unit-cell. Right: the unit-cell is tiled to obtain the full material. For the Bloch–Floquet boundary conditions, the tiling is infinite in all dimensions; **b.** The design objective is the frequency band gaps.

on pixelated metamaterials whose features are just raw pixels made of constituent materials. Discovering interpretable patterns directly from raw pixels is a more challenging and fundamental problem.

Another serious issue with using black box models like deep neural networks is that they require large labeled training datasets, and the training and testing data should come from the same distribution so that the model generalizes between training and test. However, constructing these large datasets is extremely expensive because the labels (i.e., material properties) of metamaterials are calculated by simulation, which is computationally expensive. In fact, this whole process could be so expensive that one might find it less expensive to use the simulator to get the results on the test set directly rather than go through the process of collecting a training set at all. Note that deep generative models like GANs [23,24] do not address the simulation bottleneck because training a GAN requires co-training a generator and a discriminator which requires even more training data than just training a structure-to-property predictor. Ma et al. [25] propose a self-supervised learning approach that can utilize randomly generated unlabeled data during training to reduce the amount of training data. However, this work can only handle structure-to-property prediction but not the problem of inverse design considered here. Our method instead alleviates this simulation bottleneck through a *multi-resolution* approach: our unit-cell template models are trained using a (relatively small amount of) *coarse-resolution* data but can extrapolate and generate a (large amount of) *finer-resolution* metamaterial designs. This is helpful because the coarse resolution space is much smaller than the fine-resolution space, and gives us a bird's eye view of what might happen when we sample at the finer scale throughout the space of possible metamaterials.

3. Problem settings

Here, we introduce the settings of the metamaterial design problem we are trying to solve, including the inputs and target of the dataset and their physical meanings.

We aim to design and characterize 2-D pixelated metamaterials made by tiling a 10×10 unit-cell. Such materials can be stacked to form 3D structures, and can direct, reflect or scatter waves, depending on the choice of unit-cell's material selection and geometry. In our framework, the unit-cell is a square with side length $a = 0.1$ m. However, transferring to a different length scale for a different application is easily doable by a simple scaling transformation on the dispersion relations. For a 10×10 unit-cell, the pixel side length is 1 cm; for 20×20 unit-cell, the pixel side length is 0.5 cm. Each unit-cell is made of two constituent materials: one is soft and lightweight, with elastic modulus $E = 2$ GPa[1] and density $\rho = 1{,}000$ kg/m^3, and the other is stiff and heavy with $E = 200$ GPa, and $\rho = 8{,}000$ kg/m^3. These two sets of material properties are representative of a polymer and steel respectively. Our unit-cells are symmetric, with four axes of symmetry (x, y and $\pm 45°$). Under the symmetry constraints, the coarsest resolution (10×10) unit-cell has only 15 irreducible pixels. As a result, the raw input features of a sample in our dataset is a 15-dimensional binary vector: 0 means the soft constituent material in that location, and 1 means stiff constituent material. Thus, the full coarse space can be characterized, having 2^{15} total states. Fig. 2 shows how to construct a material from the representation involving the 15 raw input features.

The material property we desire in our engineered materials is a band gap within a specific frequency range, given by the user. A band gap is a range of frequencies within which elastic waves cannot propagate and are instead reflected.

To identify the existence of a band gap, one can examine the effect of dispersion in metamaterials by calculating dispersion relations. Dispersion relations are functions that relate the wavenumber of a wave to its frequency, and they contain information regarding the frequency dependent propagation and attenuation of waves. Dispersion relations are found by computing elastic wave propagation solutions over a dense grid of wavevectors. A band gap exists when there is a range of frequencies in the dispersion relation for which no wave propagation solutions exist (see Fig. 2**b**).

Dispersion relation computations use Bloch–Floquet periodic boundary conditions, i.e., they assume that a given unit-cell is tiled infinitely in space. The physics revealed in dispersion analysis (infinite-tiling) can be leveraged in more realistic finite-tiling scenarios, as we will demonstrate later, which makes dispersion relation computations very useful for exploration and design of real materials. Our dispersion relation simulations are implemented using the finite element method.

More details about the simulation can be found in the Supplementary Information A.

We are looking for materials with band gaps in a certain frequency range. To define this task as a supervised classification problem, we create a binary label based on existence of a band

[1] GPa is gigapascals, a unit used to quantify elastic modulus.

gap in a given frequency range (e.g. [10, 20] kHz): 1 means one or more band gaps exist, 0 means no band gap exists. In other words, if a band gap range intersects with the target frequency range, the label is 1, otherwise the label is 0. One can also flexibly adjust the band gap label for different practical uses. For example, we can set the label to 1 only when the intersection of the band gap and the target frequency range is above a minimum threshold. We can also set the label to 1 when the band gap covers the entire target range, or even create a label for band gap properties in multiple frequency ranges.

4. Method

This section introduces proposed methods. Section 4.1 explains the shape-frequency features and how they can be used to optimize different objectives. Inspired by the efficiency of shape-frequency features, we then propose unit-cell template sets in Section 4.2.

4.1. Shape-frequency features

We hypothesize that the occurrence of certain local features in the metamaterials might contribute to the formation of band gaps. For example, certain local patterns/shapes in the materials can lead to interference and thus cause band gaps. Because the unit-cells are repeated, the location of such local patterns does not matter, *as long as they occur frequently*, the band gap can be formed. Such physics intuition inspires us to propose *shape-frequency features* which calculates the *number of times a pattern occurs in the unit cell divided by total number of locations*.

Denote the unit-cell as a $n \times n$ binary matrix $\mathbf{U} \in \{0, 1\}^{n\times n}$, where $\mathbf{U}_{i,j} = 0$ means pixel i, j is assigned to the soft material and $\mathbf{U}_{i,j} = 1$ when the pixel is assigned to the stiff material.

A specific shape s can be represented as a set of location offsets O_s whose elements are coordinates of pixels with respect to a reference pixel. For example, a 2×2 square window can be represented as $\{(0, 0), (1, 0), (0, 1), (1, 1)\}$. A 3×3 plus symbol as in Fig. 1 can be represented as $\{(0, 1), (1, 0), (1, 1), (1, 2), (2, 1)\}$. For a specific unit-cell, the feature value corresponding to that shape is computed by sliding the shape over the unit-cell and calculating the fraction of times that it is entirely contained within the soft material,

$$f_s = \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\mathbb{1}\left[\left(\sum_{(o_r,o_c)\in O_s}\mathbf{U}_{i+o_r,j+o_c}\right) = 0\right], \tag{1}$$

where $\mathbb{1}[\cdot]$ is the indicator function. It equals 1 if and only if all the pixels in the shape are soft material (i.e., $\sum_{o_r,o_c}\mathbf{U}_{i+o_r,j+o_c} = 0$).

We would typically consider a collection of shapes, and have one element in a unit-cell's feature vector per shape. Thus, for unit-cell i, the jth component of its feature vector corresponds to how often the full shape j appears in its soft material.

In more detail, consider the collection of shapes, shown within Fig. 3**a** (left). Note that this collection of shapes is the full set used in the results section, not just examples of them. Again, pixels of the stiff material are in yellow, and the soft material is shown in purple. We slide each of the shapes (sliding windows) over the unit-cell (Fig. 3**a** (middle)) and count the fraction of positions over which the shape is fully contained within pixels of the soft (purple) material. These fractions together form the new representation for the unit-cell (Fig. 3**a** (right)). Note that, to calculate the fraction, we should also consider the situation where the sliding window is across the boundary of two unit-cells, since the entire material is made by tiling the unit-cell. Because the unit-cells are symmetric, the occurrence of the patterns within the unit-cell are also symmetric: if we rotated the patterns by 90°, 180° or 270°, the number of detections of the pattern within the unit-cell would be identical. Note that the shape-frequency features are different from standard convolution filters used in computer vision; details are discussed in the Supplementary Information B. Theoretically, our method can be generalized to nonsquare pixels and unit-cells as well, see Supplementary Information E.

Once we calculate the shape-frequency features of the unit-cells, they can replace the original raw features and be used as the inputs of the machine learning models to predict the band gap output. We show later in Section 5.1 that using the shape-frequency features as inputs, machine learning models can predict the existence of band gaps more accurately than using raw features.

Since shape-frequency features are just new representations of the unit-cells and they are written in vector form, any type of machine learning model can be trained to predict band gap existence, taking these features as inputs. These machine learning models can not only be complex models like neural networks or boosted trees, but can also be interpretable models (e.g., sparse decision trees). Fig. 3**b** shows examples of sparse decision trees that predict the existence of band gaps from shape frequency features. When making predictions, the decision tree starts from its root node, checking if the shape shown on the node appears frequently in the unit-cell (e.g., if a 1×4 soft bar occurs in the unit-cell more than 22% of all possible locations). If so, it goes to the right branch; if not it goes to the left branch. The process is continued until a leaf node (in green or orange) is reached. At that point, it outputs the prediction of whether a band gap exists, based on the majority vote of the training data within that leaf. The paths denoted by red arrows in the trees in Fig. 3**b** show how often the local patterns need to occur in the unit-cells to predict an open band gap. More analysis of these discovered patterns can be found in Section 5.3.

4.1.1. Optimizing precision and support

For regular binary classification problems, we hope the machine learning models have high accuracy for both positive samples (materials with band gaps) and negative samples (materials with no band gap). This is also the objective if our goal is to perform only structure-to-property prediction. However, for the property-to-structure task, where our goal is to produce a number of unit-cell designs with the target band gap property, prediction accuracy is no longer a good metric. Instead, we hope all unit-cells that are predicted to have a band gap actually have a band gap, i.e., we prefer that the model has high *precision*. We also hope the total number of discovered designs, i.e., the *support*, meets the requirement of real applications. Thus, for the property-to-structure task, our objective is a combination of precision and support.

Most machine learning methods cannot directly optimize custom objectives with constraints, such as precision, constrained by support. However, there are new approaches that permit direct optimization of custom discrete objectives. We use GOSDT (Generalized and Scalable Optimal Sparse Decision Trees, [26]) for this task, because it directly optimizes decision trees for customized objectives. Different from traditional decision tree algorithms which use greedy splitting and pruning, GOSDT directly searches through the space of all possible tree structures, uses analytical bounds to reduce a huge amount of search space, and directly outputs a tree that optimizes the customized objective. We programmed it to maximize the following custom objective to optimality:

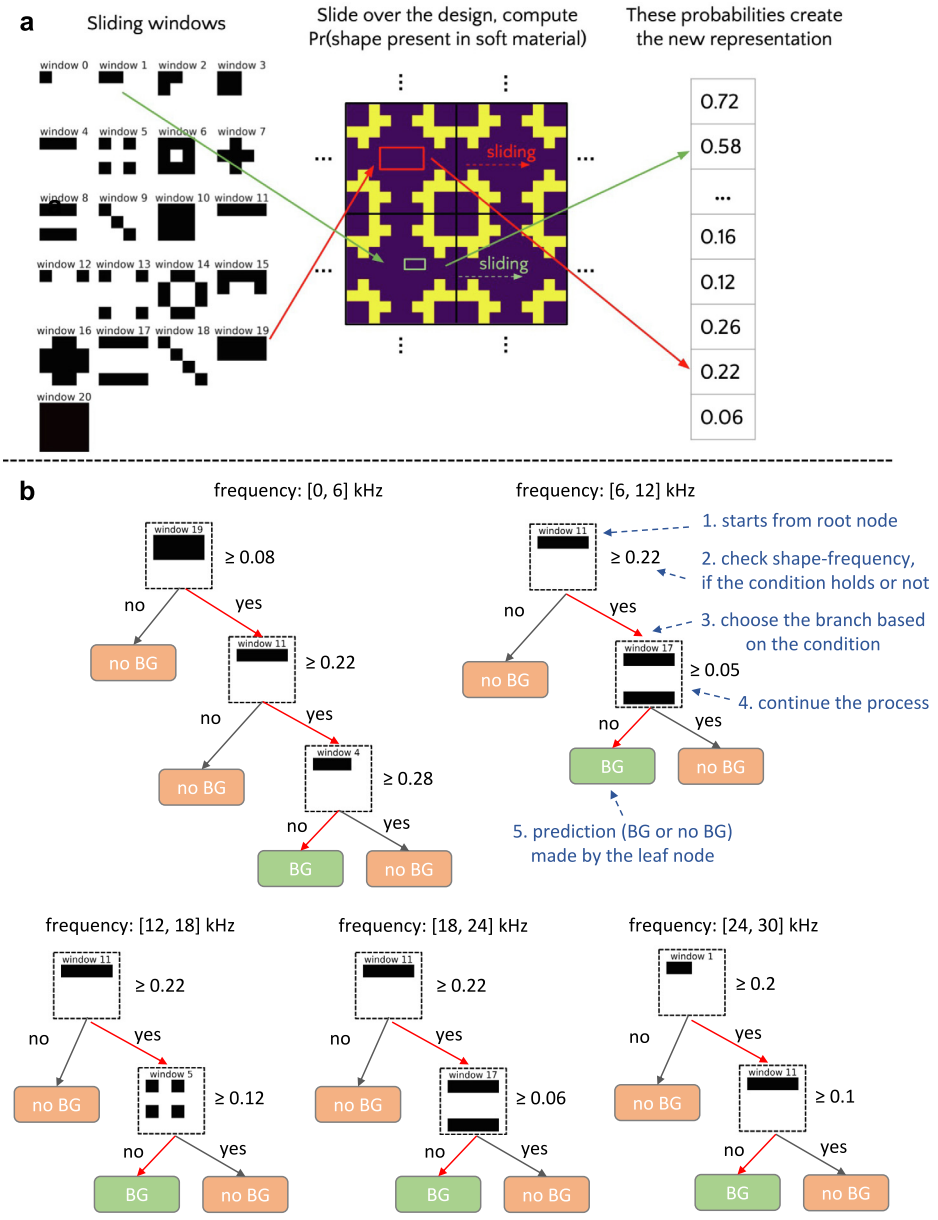

Fig. 3. a. The process of calculating shape-frequency features. *Left*: Collection of shapes (sliding windows) used to create the shape-frequency features. *Middle:* An example 10 × 10 unit-cell design (tiled 4 times for better visualization). *Right:* Shape-frequency features of the unit-cell, which count the fraction of locations in the unit-cell where the shape is present in the soft material; **b.** Optimal sparse decision trees built on shape-frequency features for predicting band gaps in different frequency ranges; red arrows denote the paths to band gap (BG) nodes. Blue text on the top-right tree breaks down how the decision tree predicts whether the band gap exists using shape-frequency features. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

$$\max_{\text{tree}} \left[\frac{TP_{\text{tree}}}{TP_{\text{tree}} + FP_{\text{tree}} + \epsilon} - \frac{K}{TP_{\text{tree}} + \epsilon} \right] \tag{2}$$

$$= \max_{\text{tree}} \left[\frac{P - FN_{\text{tree}}}{P - FN_{\text{tree}} + FP_{\text{tree}} + \epsilon} - \frac{K}{P - FN_{\text{tree}} + \epsilon} \right]. \tag{3}$$

Here, K is a parameter that balances the precision and the support; ϵ is a small constant for numerical convenience; TP, FP, TN, FN mean true positives, false positives, true negatives and false negatives. Eq. (2) shows precision (first term) and inverse support (second term); we use inverse support so that if support is large, the term diminishes in importance. The simplification in Eq. (3) shows that the objective is monotonically decreasing with respect to FN_{tree} and FP_{tree}. Theorem B.1 of [26] shows that as long as the objective is decreasing with respect to FN_{tree} and FP_{tree}, we can find an optimal sparse decision tree using GOSDT's branch and bound algorithm.

4.1.2. Property-to-structure sampling

Because we optimize the precision using GOSDT, the false positive rate of our model will be low enough to work with. At this point, we directly do rejection sampling using the decision tree to produce unit-cell designs with the target band gap. Specifically, to produce valid designs, the rejection sampling approach randomly picks structures in the design space, evaluates whether each of them are predicted to have a band gap, and outputs only these relevant designs. After sampling, each accepted sample is evaluated with the physics-based finite-element model to determine whether a band gap is present.

Algorithm 4.1 Sampling Fine resolution Unit-cell Designs via Shape-frequency Features

Input: simulated coarse resolution (10×10) dataset $\mathcal{D} := \{\mathbf{x}_i, y_i\}_{i=1}^{2^{15}}$, $\mathbf{x}_i \in \{0, 1\}^{15}$: raw features; y_i: band gap label
Parameters: set of shapes S; tree sparsity regularization λ (see [26]); K, ϵ (see Section 4.1.1)
Output: raw features of a fine resolution unit-cell $\tilde{\mathbf{x}}$
1: calculate shape-frequency features $\mathbf{x}_i^{\text{SFF}}$ = SFF($\mathbf{x}_i$,S) for all $\mathbf{x}_i$ in the dataset (coarse resolution), see Section 4.1
2: train an optimal sparse decision tree τ = GOSDT($\{\mathbf{x}_i^{\text{SFF}}, y_i\}_{i=1}^{2^{15}}$, λ, K, ϵ), see Section 4.1.2
3: **while**(True):
4: randomly sample a binary vector $\tilde{\mathbf{x}}$ as raw features in the fine resolution space
5: calculate shape-frequency features $\tilde{\mathbf{x}}^{\text{SFF}}$ = SFF($\tilde{\mathbf{x}}$,S) (fine resolution), see Section 4.1.3
6: **if** $\tau(\tilde{\mathbf{x}}^{\text{SFF}}) = 1$:
7: **return** $\tilde{\mathbf{x}}$

4.1.3. Transfer to finer resolution

The raw feature space of finer resolution samples is different from that of coarse resolution samples. Therefore, for the finer resolution data, we need to slightly modify the approach to obtain shape-frequency features that are compatible with the coarse resolution shape-frequency features. Suppose we want to transfer the model from 10×10 to 20×20 space. Then, there are three changes in calculating the shape-frequency features:

- The window size should be doubled when moving from coarse resolution to fine resolution;
- The stride of the sliding window should be 2 instead of 1;
- In counting the shape-frequency values, exact agreement between the window and soft material (purple) should be replaced with near exact agreement. In particular, when less than 2 yellow pixels (stiff material) are found in the window, we can consider this to be an agreement.

When using the shape-frequency features with these modifications, the decision tree model learned on the coarse resolution data can be directly applied to the fine resolution. Thus, we can also do rejection sampling on the fine resolution with the model.

Algorithm 4.1 shows the entire pipeline of using an optimal sparse decision tree built on shape-frequency features to sample fine resolution designs with the target band gap property. Visual illustration of the sampling process can be found in Supplementary Information C.

4.2. Unit-cell template sets

Here, we introduce another interpretable machine learning model, called unit-cell template sets. Different from sparse trees on shape-frequency features, which focus on local patterns, a unit-cell template captures a global pattern for the unit-cell that is related to the target properties. The unit-cell template is a $n \times n$ matrix $\mathbf{T} \in \{0, 1, *\}^{n \times n}$, where $\mathbf{T}_{i,j} = 0$ means the pixel is soft material, $\mathbf{T}_{i,j} = 1$ means the pixel is stiff material, and $\mathbf{T}_{i,j} = *$ means the pixel could be either soft or stiff, i.e., a free pixel.

Definition *(match)*. We say a unit-cell design $\mathbf{U}$ *matches* the unit-cell template if and only if all pixels with value 0 on the template are also 0 on the design, and all pixels with value 1 on the unit-cell template are also 1 on the design. That is, $\forall(i, j)$ such that $\mathbf{T}_{i,j} \neq *$, we have $\mathbf{U}_{i,j} = \mathbf{T}_{i,j}$.

A unit-cell template set contains a set of unit-cell templates, and the sample design is predicted as positive if and only if it matches at least one unit-cell template in the set. Fig. 4**a** shows an example of a unit-cell template set that consists of five different templates. We proposed unit-cell templates because we found that some pixels in the unit-cells are more important for the formation of band gaps than others. For the pixels that are not important, even if they are flipped, the band gaps remain unchanged; we denote these as free pixels in the unit-cell template. The free pixels identify unimportant regions where changes do not affect the target band gaps, while the other pixels form the key global pattern that leads to the band gap. We aim to find a unit-cell template set that captures a diverse set of global patterns related to the target band gap. The relationships between unit-cell template sets and other machine learning methods are discussed in Supplementary Information B. Theoretically, our method can be generalized to nonsquare pixels and unit-cells as well, see Supplementary Information E.

The training objective of the model is to find a small number of unit-cell templates, such that the training precision of the entire model is high enough, and the model covers as many valid designs as possible, i.e., maximizing the support under a minimum precision constraint. The reasons for optimizing precision and support were explained in Section 4.1.1, and we set a limit for the total number of selected unit-cell templates to encourage the unit-cell template sets to contain a diverse set of unit-cell templates. Because the total number of possible templates is extremely large, the training process is divided into two steps, a pre-selection of candidate unit-cell templates that filters out useless templates to reduce the problem size (Section 4.2.1), and a integer linear programming (ILP) formulation to optimally select from the candidates obtained in the first step (Section 4.2.2). Fig. 4**b** shows the unit-cell template sets learned by the proposed algorithm for band gap prediction. More analysis of these discovered patterns can be found in Section 5.3.

4.2.1. Pre-selection of templates

Here we consider symmetric unit-cell templates, because designs in our dataset are all symmetric. By removing all symmetry redundancy, the unit-cell template can be represented by its irreducible pixels, i.e. a 15 dimensional vector $\mathbf{t} \in \{0, 1, *\}^{15}$. The total number of possible templates is 3^{15} which is approximately 14.3 million. This is too large for the ILP in the next step. However, among the 3^{15} possible templates, most of them would never be selected because either their precision or support is not high enough. For example, a unit-cell template with precision 80% is not likely to be used if we want the entire model to have precision above 99%, i.e., we hope the unit-cells are all having the desired band gap properties. Also, if the support of a unit-cell template, i.e., the total number of designs that match it, is very small (e.g., < 10), the model may not generalize well. Therefore, we pre-select the unit-cell templates by setting minimum thresholds of precision and support, which reduces the search space only to promising unit-cell templates. We select the unit-cell template as a candidate only when it meets the minimum thresholds.

Observing that the entries are all binary in the unit-cell designs, we implement the precision calculation via bit operations,

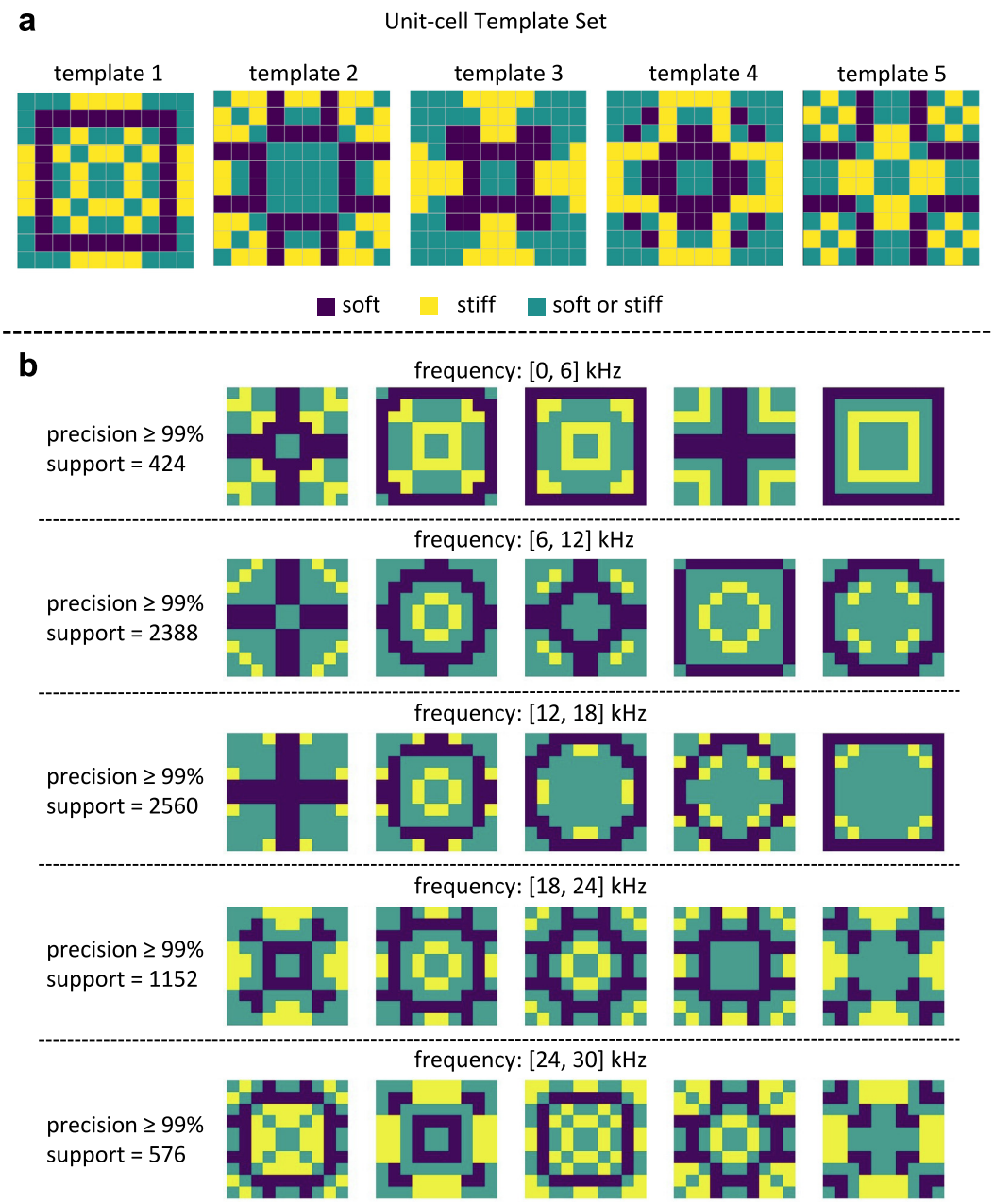

Fig. 4. a. An Example of a unit-cell template set. A unit-cell is predicted as positive as long as it matches at least one unit-cell template in the set; **b**. unit-cell template sets learned for predicting band gaps in different frequency ranges. Note that because unit-cells are tiled, a large cross through the center is identical to a square on the border. (E.g., consider the two upper right unit-cells.).

which significantly improves the speed of the pre-selection step. With the bit-operation implementation, the pre-selection steps of all 3^{15} possible templates finish in 50 s. The number of unit-cell templates that remains is typically in the range of 6000 to 12000, which is now suitable for ILP.

4.2.2. ILP for template selection

After the pre-selection step, we have a set of candidate unit-cell templates. Since the pre-selection step significantly cuts down the space of templates, we can directly formulate the template selection as an optimization problem, and solve it to provable optimality. Specifically, we formulate a ILP to optimally select from the candidates. Suppose we have n designs and m candidate templates. The goal of the ILP is to choose at most s unit-cell templates ($s \ll m$) whose union forms a model, such that the support is maximized and the precision of the model is at least p. Denote the true labels of all designs by a binary vector $\mathbf{y} \in \{0, 1\}^n$, and the predicted labels by $\hat{\mathbf{y}} \in \{0, 1\}^n$. $\mathbf{M} \in \{0, 1\}^{n \times m}$ denotes a matching matrix, where $M_{i,j}$ indicates whether design i matches template j (1 for match, and 0 for not match). Binary vector $\mathbf{c} \in \{0, 1\}^m$ denotes the chosen unit-cell templates, where c_j indicates whether template j is chosen (1 for choose, and 0 for not choose). We solve the following ILP for template selection.

$$\max \sum_{i=1}^{n} \hat{y}_i \quad \text{(optimizing support)} \tag{4}$$

$$\text{s.t.} \sum_{j=1}^{m} c_j \leq s \quad \text{(sparsity constraint)} \tag{5}$$

$$\sum_{i=1}^{n} y_i \cdot \hat{y}_i \geq \left(\sum_{i=1}^{n} \hat{y}_i \right) \cdot p \quad \text{(minimum precision)} \tag{6}$$

$$\sum_{j=1}^{m} M_{i,j} \cdot c_j \geq \hat{y}_i, \quad i = 1, \ldots, n \quad \text{(define } \hat{y}_i\text{)} \tag{7}$$

$$\sum_{j=1}^{m} M_{i,j} \cdot c_j \leq m \cdot \hat{y}_i, \quad i = 1, \ldots, n \quad \text{(define } \hat{y}_i\text{)} \tag{8}$$

$$c_j \in \{0, 1\}, \quad j = 1, \ldots, m \tag{9}$$

$$\hat{y}_i \in \{0, 1\}, \quad i = 1, \ldots, n. \tag{10}$$

In this ILP, the objective (4) means maximizing the total number of designs predicted as positive, which is the same as support. Constraint (5) controls the sparsity, i.e., choose at most s unit-cell templates. Constraint (6) guarantees training precision of the model is at least p. (7) and (8) constraints together define $\hat{y}_i$, where $\hat{y}_i = 1$ if and only if design i matches at least one of the chosen unit-cell templates. In particular, (7) says that if design i does not match any chosen template j (i.e., whenever c_j is 1, $M_{i,j}$ happens to be 0), then $\hat{y}_i$ will be set to 0. (8) will ensure that if there is a match for design i to any of the chosen templates (which all have $c_j = 1$), then this design is assigned $\hat{y}_i$=1. Using a commercial MIP solver, a problem with around 10000 candidate templates can be solved to optimality (when the current best solution meets the upper bound of the best possible solution) or near-optimality in about 10 min (running single-threaded on one core of a 2.66 GHz Intel E5640 Xeon Processor). If $s = 5$, it can be solved to optimality all the time, and when $s = 10$, the optimality gap (difference between current best solution and an upper bound of the best possible solution as the percentage of the upper bound) is always <20% for a run time of 30 min (running under the same environment); it is worthwhile to note that optimal solutions are often attained quickly, but the solvers can take a while to prove that the solution is optimal. Note that, if desired, one can set higher minimum precision and support thresholds for the pre-selection step to make the problem even smaller, so that the ILP can be solved even faster. We choose $s = 5$ for all the experiments in the main paper.

The result of the ILP is our unit-cell template set.

4.2.3. Property-to-structure sampling

After training the structure-to-property model, the resulting unit-cell template set can be directly used to solve the inverse property-to-structure problem. An easy sampling procedure to do this is as follows: first, randomly choose a unit-cell template **t** from the unit-cell template set, where the probability to choose each template is proportional to its support; second, for all entries in **t** that equal *, randomly assign value 0 or 1 to them. These sampled unit-cells are likely to have the desired band gap.

4.2.4. Transfer to finer resolution

The unit-cell template set naturally transfers coarse scale information to finer resolutions. In particular, by subdividing each pixel in the unit-cell template into four sub-pixels, we directly obtain a unit-cell template defined on a finer-resolution space.

Algorithm 4.2 shows the entire pipeline of using unit-cell template set to sample fine resolution designs with the target band gap property. Visual illustration of the sampling process can be found in Supplementary Information C.

5. Results

The results are organized according to four objectives we want to achieve with the proposed methods, including (a) design-to-property prediction; (b) property-to design sampling; (c) identify key patterns; (d) transfer to finer resolution. We evaluate how well the proposed methods can achieve these objectives, followed by several tests involving practical applications in materials discovery.

5.1. Objective 1: Structure-to-property prediction

Here, we test how well the proposed methods can perform structure-to-property prediction. We chose five frequency ranges ([0, 10], [10, 20], [20, 30], [30, 40] and [40, 50] kHz) to predict the existence of band gaps; these frequency ranges correspond to five different binary classification problems. Using balanced accuracy (bacc) as the evaluation metric, we compare the predictive performance of a diverse set of ML models with and without the shape-frequency features. We specifically consider linear models like support vector machines with linear kernels (SVMs) and logistic regression (LR); tree-based models like CART, random forest (RF), and boosted trees (LightGBM [27]), as well as neural networks including the multi-layer perceptron (MLP). We also compare the proposed method with convolutional neural networks (CNNs), since they have been widely used in previous works of ML-based metamaterial design. The baccs of each model trained on raw feature, SFF, and improvements of SFF over raw features, are shown in Table 1 (a). We train each model 5 times and average the accuracy.

Our results show that **using the shape-frequency features, rather than the original raw features, improves the accuracy of classifiers for most machine learning methods**, especially tree-based methods such as boosted trees, but with the exception of MLP (SFF decreases its in [0, 10] kHz, [10, 20] kHz, and [20, 30] kHz).

One might expect CNNs to achieve great success in classifying band gaps for 2-D metamaterials since the unit-cells share many similarities with images. However, LightGBM [27] built on shape-frequency features outperforms ResNet18 [28] in all ranges. In some cases, e.g., within frequency range [40, 50] kHz, simple models like CART outperform CNNs.

More details of the experiment (e.g., hyper-parameter settings) can be found in Supplementary Information D.1.

5.2. Objective 2: Property-to-structure sampling

Using the methods discussed in Sections 4.1.2 and 4.2.3, we are able to solve the inverse design (property-to-structure sampling) problem.

In practice, materials scientists need valid designs with the target property, but they do not require the set of *all* designs with the property. As such, our performance metric is precision, rather than recall. We also calculate the support, which is the total number of testing samples predicted as positive, to ensure the models can generate enough potentially-valid designs. Table 1 (b) lists the precision and support values from different methods. The methods we compared include GOSDT trained on shape-frequency features (denoted SFF) with the objective in Section 4.1.1, the unit-cell template set, and LightGBMs trained on SFF and raw features.

In terms of precision, **SFF+GOSDT and unit-cell template sets significantly outperformed LightGBMs**. This is probably owing to the fact that the proposed methods directly optimize precision. LightGBMs maintain larger support, while the support of SFF+GOSDT and unit-cell template sets is much lower. But for practical use, it is sufficient that the model finds dozens of valid designs. The average sampling time of these methods and the results of unit-cell template sets with different sparsity constraints can be found in Supplementary Information D.2.

5.3. Objective 3: Show key patterns

One advantage of the proposed methods is model interpretability; we aim to explicitly identify the key patterns learned from data that are related to the target property. In this way, domain

Algorithm 4.2 Sampling Fine Resolution Unit-cell Designs via Unit-cell Template Set

Input: simulated coarse resolution (10×10) dataset $\mathcal{D} := \{\mathbf{x}_i, y_i\}_{i=1}^{2^{15}}$, $\mathbf{x}_i \in \{0, 1\}^{15}$: raw features; y_i: band gap label
Parameters: pre-selection support ψ_{pre}, pre-selection precision p_{pre}, sparsity constraint s, minimum precision p
Output: raw features of a fine resolution unit-cell $\tilde{\mathbf{x}}$
1: pre-select candidate template set S_T^{pre} = pre-selecting($\mathcal{D}$, $\{0, 1, *\}^{15}$, ψ_{pre}, p_{pre}), see Section 4.2.1
2: run ILP to find optimal template set S_T^* = ILP($\mathcal{D}$, S_T^{pre}, s, p), see Section 4.2.2
3: randomly pick a template $\mathbf{t} \in S_T^*$
4: expand $\mathbf{t}$ to fine resolution space, get $\tilde{\mathbf{t}}$
5: randomly set to 0 or 1 for all * elements in $\tilde{\mathbf{t}}$, get $\tilde{\mathbf{x}}$
6: **return** $\tilde{\mathbf{x}}$

Table 1
Summary of key quantitative results. The results are organized with respect to different objectives of data-driven metamaterials design. (a) structure-to-property prediction; (b) property-to-structure sampling; (c) transfer to finer resolution.

(a) Structure-to-property prediction: Testing balanced accuracies (baccs) of different methods. We mark the models with the best bacc in each frequency range in bold.

Model		Frequency range				
		[0, 10] kHz	[10, 20] kHz	[20, 30] kHz	[30, 40] kHz	[40, 50] kHz
SVM	Raw	71.77%	73.88%	50.85%	54.93%	49.84%
	SFF (ours)	75.62%	77.35%	67.96%	59.89%	49.93%
	Improvement	+3.85%	+3.47%	+17.11%	+4.96%	0.09%
LR	Raw	78.03%	75.44%	56.31%	76.59%	90.96%
	SFF (ours)	80.53%	79.55%	69.04%	76.9%	91.32%
	Improvement	+2.50%	+4.11%	+12.73%	+0.31%	+0.36%
RF	Raw	85.81%	80.04%	73.00%	77.66%	87.54%
	SFF (ours)	85.52%	81.98%	74.53%	81.26%	**95.35%**
	Improvement	−0.29%	+1.94%	+1.53%	+3.60%	+7.81%
CART	Raw	84.74%	75.68%	63.40%	73.95%	86.82%
	SFF (ours)	83.63%	80.13%	70.72%	79.73%	94.47%
	Improvement	−1.11%	+4.45%	+7.32%	+5.78%	+7.65%
MLP	Raw	90.72%	86.39%	78.15%	77.47%	65.90%
	SFF (ours)	85.90%	82.55%	76.01%	77.69%	66.4%
	Improvement	−4.82%	−3.84%	−2.14%	+0.22%	+0.50%
LightGBM	Raw	91.32%	88.27%	81.11%	82.62%	77.76%
	SFF (ours)	**96.10%**	**90.97%**	**85.57%**	**90.01%**	94.76%
	Improvement	+4.78%	+2.70%	+4.46%	+7.39%	+17.00%
CNN	Raw	93.24%	89.76%	81.65%	79.56%	84.83%

(b) Property-to-structure sampling: Testing precision and support of different methods. Numbers in the table cells are formatted as "precision, support." The testing support here is calculated among 6554 testing samples (20% of the entire dataset).

Frequency range	Raw+LightGBM	SFF+LightGBM	SFF+GOSDT (ours)	Unit-cell template sets (ours)
[0, 10] kHz	80.77%, 2310	88.93%, 2169	95.77%, 89	**98.53%**, 339
[10, 20] kHz	93.52%, 4013	95.62%, 4063	**98.11%**, 423	**98.68%**, 758
[20, 30] kHz	86.94%, 3654	89.56%, 3811	**94.15%**, 205	**94.08%**, 203

(c) Transfer to finer resolution: Transfer precision of different methods.

Frequency range	CNN+resizing	LightGBM+resizing	SFF+GOSDT (ours)	Unit-cell template sets (ours)		
	20×20	20×20	20×20	20×20	40×40	80×80
[0, 10] kHz	18.0%	25.0%	72.5%	**100.0%**	**100.0%**	**100.0%**
[10, 20] kHz	58.0%	68.5%	73.5%	**98.5%**	**99.0%**	**100.0%**
[20, 30] kHz	39.5%	52.5%	25.0%	**91.0%**	**96.0%**	**98.0%**

experts can verify whether the learned rules are aligned with the domain knowledge, or even discover new knowledge.

In Figs. 3**b** and 4**b**, we visualize the GOSDT+SFF and unit-cell template set learned for band gaps in several frequency ranges ([0, 6], [6, 12], [12, 18], [18, 24], [24, 30] kHz).

In Fig. 3**b**, the top splits of each tree trained on shape-frequency features seem to be looking for bars in the soft material. Taking the top-right tree in 3**b** as an example, the root node checks if the 1×4 soft bar occurs in more than 22% of the places in the unit-cell. The unit-cell frequencies need to pass the thresholds to get to the "band gap" node. All band gap nodes are on the left branch of the deepest decision nodes in the trees. For instance, the deepest decision node of the top-right tree checks if shape 17, two 1×4 soft bars 2 pixels away from each other, occurs with more than 5% frequency in the unit-cell. To reach the band gap prediction node, the sample needs to go to the left branch, where shape 17 should occur with less than 5% frequency. This indicates the unit-cells should not have too many soft material patterns. In the field of elastic wave propagation, it is known that the presence of stiff inclusions in a matrix of a softer material may open a band gap due to scattering or resonant dynamics. The trees we found seem to be looking for patterns that fit this

description: the deepest node encourages the existence of stiff inclusion while the first node encourages more soft material.

The unit-cell templates (Fig. 4**b**) contrast with the shape frequency features in that they explicitly identify global (rather than local) patterns. In the unit-cell template sets for different frequency ranges, we can observe the existence of soft circles (closed curves) and stiff inclusions inside the circles, which are responsible for the formation of band gaps. As the frequency range moves higher, the size of the circle decreases. This supports the physical intuition that the smaller the stiff inclusions, the higher the frequency of the band gap.

5.4. Objective 4: Transfer to finer resolution

In Sections 4.1.3 and 4.2.4, we discussed how the proposed methods can transfer coarse scale information to finer resolution design space. To evaluate how well the model can transfer information, we trained the models on coarse resolution (10×10) unit-cells and tested them on finer resolution (20×20, 40×40 and 80×80) unit-cells. Table 1(c) shows the transfer precision of GOSDT+SFF and unit-cell template sets for band gaps in different frequency ranges. Other ML methods are not directly comparable because standard ML models trained on 10×10 data cannot take in 20×20 data. Therefore, we compared the proposed methods with baselines with slight modifications: we resized the fine resolution (20×20) unit-cell to the original size (10×10) and applied two algorithms (CNN or LightGBM) for rejection sampling. As before, if the CNN or LightGBM model predicts that the resized design has a band gap, we accept that sample, otherwise we reject it. The resizing was done via bicubic interpolation. Here, we did not compare with deep generative models such as GANs. Although GANs (which are notoriously hard to train) might generate materials faster than rejection sampling, their precision can only be lower because GANs' discriminators are co-trained with the generator, and thus cannot be more accurate than a CNN directly trained to predict only the target. For each frequency range, we asked the trained models to sample 200 unit-cell designs in finer resolution space, and ran the FEA simulation to obtain the true band gap property for evaluating the transfer precision. For the baseline models and GOSDT+SFF, we show the results for 20×20 design space. As unit-cell template sets performs extremely well on this task, and generates new designs efficiently, we also show its results in 40×40 and 80×80 design space. Please see Supplementary Information C for visual illustrations of how to sample fine resolution designs using each model.

The results in Table 1(c) indicate that **the unit-cell templates, when transferred to all finer resolutions (20×20, 40×40 or 80×80), have very high precision, with almost no precision drop compared to 10×10**. GOSDT+SFF and other baselines do not generalize as well as unit-cell templates to the finer resolution design space. GOSDT+SFF performs better than the resizing baselines in [0, 10] kHz and [10, 20] kHz, but performs worse than baselines in [20, 30] kHz. Interestingly, the transfer precisions, of both GOSDT+SFF and unit-cell template sets, decrease as frequency ranges moves higher, although unit-cell template sets have a much slower precision decrease than GOSDT+SFF. A possible explanation for the decrease of precision is that, in higher frequency ranges, the band gaps are physically more related to finer scale features that are not included in the coarse resolution dataset. Since the models are trained on coarse resolution data, they can only transfer physics that occurs in coarse patterns to finer resolution design space, but cannot discover finer scale physics without supervision. But as shown by the transfer precision results, we should emphasize that our unit-cell template sets method was capable of extracting critical coarse resolution features such that this decrease of precision at finer resolution due to wave physics is minimized (the worst transfer precision is still above 90%). One further potential improvement to this is to add new samples at each finer resolution design space when transferring between extreme scales.

In Supplementary Information D.3, we show additional results on sampling with correlation between green pixels for unit-cell template sets, which demonstrates the surprising flexibility of unit-cell template sets in terms of designing at finer-resolution design space.

5.5. Practicality test

In our method and simulations so far, we assumed the unit-cell is tiled infinitely for computational convenience. However, a unit-cell can only be tiled finitely in practice and the results can differ for infinite and finitely tiled domains due to boundary conditions. To test whether the designs found by our method work in practice, we simulated the dispersion relations of finitely-tiled materials made by unit-cell designs discovered by our method. See Supplementary Information D.4 for results of the finite tiling COMSOL simulation. The results show that our method is robust under finite tiling.

In addition to the finite tiling test, we also tested the practicality of the proposed method on its ability to create a wave demultiplexer (Fig. 5), in which waves with different frequencies travel through the materials in different directions. That is, a signal enters the demultiplexer, and there are three different possible outputs; which one will be non-zero depends on the frequency of the input signal. Specifically, we will build the demultiplexer to route signals from three different frequency ranges in different directions.

We need 5 different materials to build the demultiplexer: one material with band gaps covering all three ranges, one material allowing band pass in all ranges, and three materials allowing band pass in one range while blocking the other two ranges. We use homogeneous stiff unit-cells for the material allowing band pass in all ranges. For other materials, we train a unit-cell template set to find 20×20 unit-cell designs with these properties, and assemble the 5 unit-cells to build the demultiplexer (top row of Fig. 5). The bottom row of Fig. 5 shows the how the demultiplexer successfully guides waves with different frequencies (11.45, 13.97 and 15.55 [kHz]) towards different directions, which was the goal of the experiment.

6. Conclusion and discussion

Our work shows the power of interpretable machine learning tools in material design. The approach has achieved both mechanistic understanding, e.g., physically interpretable rules of patterns that lead to band gaps, and designing new materials with desired functionality. The approach has been demonstrated to be predictive for both infinite domains and realistic finite domains, and it has been able to design the material geometry for a wave demultiplexer. Additionally, our multi-resolution framework, which robustly carries coarse-scale knowledge to finer resolutions, is potentially applicable to a wide range of materials science problems.

Since our method learns robust coarse-scale features that can generalize to finer-resolution design space, it might also be useful in future studies for determining how to collect finer-resolution training data for rapidly capturing fine-scale physics. Using these new data, we might be able to fine tune the model so that it can more efficiently capture physics at multiple scales.

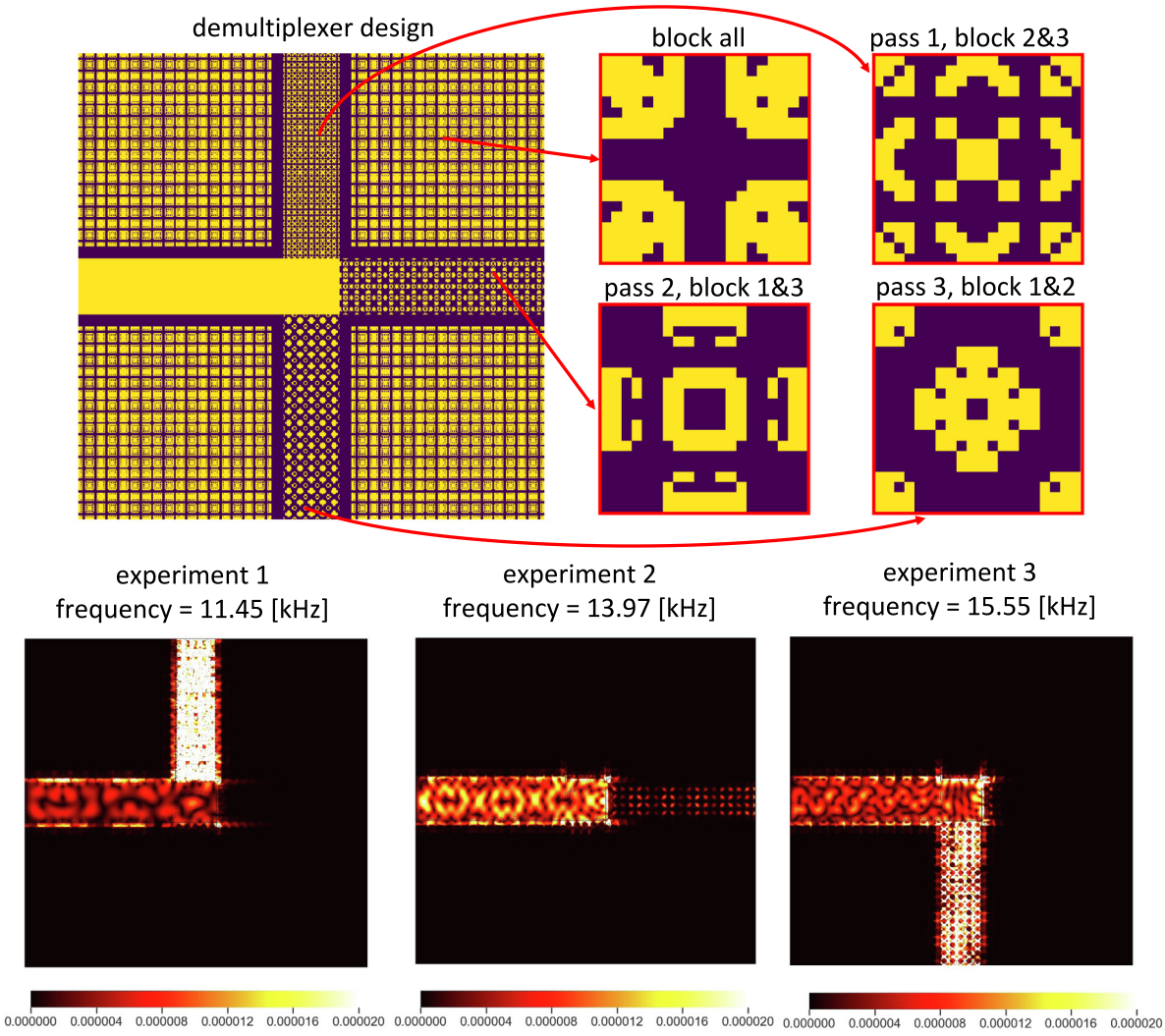

Fig. 5. Mechanical wave demultiplexer and the displacement fields for input signals frequencies. Each part of the demultiplexer was built using our discovered unit-cells that have desired properties. *Top*: The demultiplexer and the unit-cell designs used to make the demultiplexer. *Bottom*: Magnitude of displacement fields when signals of different frequencies (11.45, 13.97 and 15.55 [kHz]) are fed into the left side of the demultiplexer. The displacement values are clipped if they exceed the display range. The signals with different frequencies go through different channels in the demultiplexer: 11.45 [kHz] goes upward, 13.97 [kHz] goes right, and 15.55 [kHz] goes downward, as desired. Note that, although the passing signal of experiment 2 is not as strong as signals in the other experiments, it still pass the channel on the right without fading inside the channel.

CRediT authorship contribution statement

Zhi Chen: Developed the methods, Designed metrics, Designed visualization, Ran experiments related to the ML algorithms, Discussed the results, Contributed to the writing. **Alexander Ogren:** Developed the simulation code, Generated the data, Did the practicality test, Discussed the results, Contributed to the writing. **Chiara Daraio:** Conceived, Supervised the project, Discussed the results, Contributed to the writing. **L. Catherine Brinson:** Conceived, Supervised the project, Discussed the results, Contributed to the writing. **Cynthia Rudin:** Conceived, Supervised the project, Discussed the results, Contributed to the writing.

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.

Code and data availability

The code and data for replicating our experiments are available on https://github.com/zhiCHEN96/interpretable_ml_metamaterials.git.

Acknowledgments

The authors are grateful to M. Bastawrous, A. Lin, K. Liu, C. Zhong, O. Bilal, W. Chen, C. Tomasi, and S. Mukherjee for the feedback and assistance they provided during the development and preparation of this research. The authors acknowledge funding from the National Science Foundation, United States under grant OAC-1835782, Department of Energy, United States under grants DE-SC0021358 and DE-SC0023194, and National Research Traineeship Program under NSF, United States grants DGE-2022040 and CCF-1934964.

Appendix A. Supplementary information

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.eml.2022.101895.

References

[1] Katia Bertoldi, Vincenzo Vitelli, Johan Christensen, Martin Van Hecke, Flexible mechanical metamaterials, Nat. Rev. Mater. 2 (2017) 17066.

[2] Ole Sigmund, Jakob Søndergaard Jensen, Systematic design of phononic band–gap materials and structures by topology optimization, Phil. Trans. R. Soc. A 361 (1806) (2003) 1001–1019.

[3] Ole Sigmund, In systematic design of metamaterials by topology optimization, in: Iutam Symposium (Ed.), On Modelling Nanomaterials and Nanosystems, Dordrecht, 2009; Pyrz, R, Rauhe, J. C. Eds. Springer Netherlands, Dordrecht, 2009, pp. 151–159.

[4] Osama R Bilal, Mahmoud I Hussein, Ultrawide phononic band gap for combined in-plane and out-of-plane waves, Phys. Rev. E 84 (2011) 065701.

[5] Christian Schumacher, Bernd Bickel, Jan Rys, Steve Marschner, Chiara Daraio, Markus Gross, Microstructures to control elasticity in 3D printing, ACM Trans. Graph. 34 (2015) 136.

[6] Christian C Nadell, Bohao Huang, Jordan M Malof, Willie J Padilla, Deep learning for accelerated all-dielectric metasurface design, Opt. Express 27 (20) (2019) 27523–27535.

[7] Ramin Bostanabad, Yu-Chin Chan, Liwei Wang, Ping Zhu, Wei Chen, Globally approximate gaussian processes for big data with application to data-driven metamaterials design, J. Mech. Des. 141 (11) (2019).

[8] Tianshuo Qiu, Xin Shi, Jiafu Wang, Yongfeng Li, Shaobo Qu, Qiang Cheng, Tiejun Cui, Sai Sui, Deep learning: A rapid and efficient route to automatic metasurface design, Adv. Sci. 6 (12) (2019) 1900128.

[9] David Finol, Yan Lu, Vijay Mahadevan, Ankit Srivastava, Deep convolutional neural networks for eigenvalue problems in mechanics, Internat. J. Numer. Methods Engrg. 118 (5) (2019) 258–275.

[10] Peter R. Wiecha, Otto L. Muskens, Deep learning meets nanophotonics: A generalized accurate predictor for near fields and far fields of arbitrary 3D nanostructures, Nano Lett. 20 (1) (2019) 329–338.

[11] Zhaocheng Liu, Dayu Zhu, Sean P Rodrigues, Kyu-Tae Lee, Wenshan Cai, Generative model for the inverse design of metasurfaces, Nano Lett. 18 (10) (2018) 6570–6576.

[12] Wei Ma, Feng Cheng, Yongmin Liu, Deep-learning-enabled on-demand design of chiral metamaterials, ACS Nano 12 (6) (2018) 6326–6334.

[13] Yang Deng, Simiao Ren, Kebin Fan, Jordan M Malof, Willie J Padilla, Neural-adjoint method for the inverse design of all-dielectric metasurfaces, Opt. Express 29 (5) (2021) 7526–7534.

[14] Wei Ma, Feng Cheng, Yihao Xu, Qinlong Wen, Yongmin Liu, Probabilistic representation and inverse design of metamaterials based on a deep generative model with semi-supervised learning strategy, Adv. Mater. 31 (35) (2019) 1901111.

[15] Liwei Wang, Yu-Chin Chan, Faez Ahmed, Zhao Liu, Ping Zhu, Wei Chen, Deep generative modeling for mechanistic-based learning and design of metamaterial systems, Comput. Methods Appl. Mech. Engrg. 372 (2020) 113377.

[16] Haoran Ren, Wei Shao, Yi Li, Flora Salim, Min Gu, Three-dimensional vectorial holography based on machine learning inverse design, Sci. Adv. 6 (16) (2020) eaaz4261.

[17] Zhaocheng Liu, Dayu Zhu, Kyu-Tae Lee, Andrew S Kim, Lakshmi Raju, Wenshan Cai, Compounding meta-atoms into metamolecules with hybrid artificial intelligence techniques, Adv. Mater. 32 (6) (2020) 1904790.

[18] Jiaqi Jiang, Mingkun Chen, Jonathan A. Fan, Deep neural networks for the evaluation and design of photonic devices, Nat. Rev. Mater. (2020) 1–22.

[19] Wei Ma, Zhaocheng Liu, Zhaxylyk A Kudyshev, Alexandra Boltasseva, Wenshan Cai, Yongmin Liu, Deep learning for the design of photonic structures, Nat. Photon. 15 (2) (2021) 77–90.

[20] Yihao Xu, Xianzhe Zhang, Yun Fu, Yongmin Liu, Interfacing photonics with artificial intelligence: An innovative design strategy for photonic structures and devices based on artificial neural networks, Photon. Res. 9 (4) (2021) B135–B152.

[21] Mahmoud Elzouka, Charles Yang, Adrian Albert, Sean Lubner, Ravi S Prasher, Interpretable inverse design of particle spectral emissivity using machine learning, 2020, arXiv preprint arXiv:2002.04223.

[22] Yi Zhu, Evgueni T. Filipov, Harnessing interpretable machine learning for origami feature design and pattern selection, 2022, arXiv preprint arXiv:2204.07235.

[23] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, Generative adversarial nets, Adv. Neural Inf. Process. Syst. 27 (2014).

[24] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, Anil A Bharath, Generative adversarial networks: An overview, IEEE Signal Process. Mag. 35 (1) (2018) 53–65.

[25] Wei Ma, Yongmin Liu, A data-efficient self-supervised deep learning model for design and characterization of nanophotonic structures, Sci. China Phys. Mech. Astron. 63 (8) (2020) 1–8.

[26] Jimmy Lin, Chudi Zhong, Diane Hu, Cynthia Rudin, Margo Selzer, Generalized optimal sparse decision trees, in: Proc. International Conference on Machine Learning, 2020.

[27] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, Tie-Yan Liu, Lightgbm: A highly efficient gradient boosting decision tree, in: Advances in Neural Information Processing Systems, 2017, pp. 3146–3154.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.