Machine learning of implicit combinatorial rules in mechanical metamaterials

Mastrigt, R. van; Dijkstra, M.; Hecke, M.L. van; Coulais, C.

Machine Learning of Implicit Combinatorial Rules in Mechanical Metamaterials

Ryan van Mastrigt[,1,2,*] Marjolein Dijkstra[,3] Martin van Hecke[,2,4] and Corentin Coulais[1]

[1]*Institute of Physics, Universiteit van Amsterdam, Science Park 904, 1098 XH Amsterdam, The Netherlands*
[2]*AMOLF, Science Park 104, 1098 XG Amsterdam, The Netherlands*
[3]*Soft Condensed Matter, Debye Institute for Nanomaterials Science, Department of Physics, Utrecht University,*
Princetonplein 5, 3584 CC Utrecht, The Netherlands
[4]*Huygens-Kamerling Onnes Lab, Universiteit Leiden, Postbus 9504, 2300 RA Leiden, The Netherlands*

Combinatorial problems arising in puzzles, origami, and (meta)material design have rare sets of solutions, which define complex and sharply delineated boundaries in configuration space. These boundaries are difficult to capture with conventional statistical and numerical methods. Here we show that convolutional neural networks can learn to recognize these boundaries for combinatorial mechanical metamaterials, down to finest detail, despite using heavily undersampled training sets, and can successfully generalize. This suggests that the network infers the underlying combinatorial rules from the sparse training set, opening up new possibilities for complex design of (meta)materials.

From proteins and magnets to metamaterials, all around us systems with emergent properties are made from collections of interacting building blocks. Classifying such systems—do they fold, are they magnetized, do they have a target property—normally involves calculating these properties from their structure. This is often straightforward in principle, yet computationally expensive in practice, e.g., requiring the diagonalization of large matrices. Machine learning algorithms such as neural networks (NNs) forgo the need for such calculations by "learning" the classification of structures. In particular, machine learning has proven successful to find patterns in crumpling [1], active matter [2–4] and hydrodynamics [5], photonics [6–8], predict structural defects and plasticity [9,10], design metamaterials [11–18], determine order parameters [19–26], identify phase transitions [27–44], and predict protein structure [45]. In these examples, the relevant property typically varies smoothly and there is no sharp boundary separating classes in configuration space. NNs are thought to be successful because they interpolate these blurred boundaries, even when the configuration space is heavily undersampled.

In contrast, combinatorial problems, viz. those where building blocks have to fit together as in a jigsaw puzzle, feature a sharp boundary between compatible (C) and incompatible (I) configurations. Such problems arise in self-assembly [46,47], folding [48,49], tiling problems [50], and combinatorial mechanical metamaterials [51–54]. The latter are created by tiling different unit cells and are restricted by kinematic compatibility. A simple example is that of structures that can be either floppy (zero mode) or frustrated (no zero mode) [Figs. 1(a) and 1(b)]. The floppy structures require a specific arrangement of building blocks

where all the deformations fit together compatibly (C), and therefore are rare and very sensitive to small perturbations. These perturbations are likely to induce frustrated incompatible (I) configurations [Fig. 1(b)]. The space of C
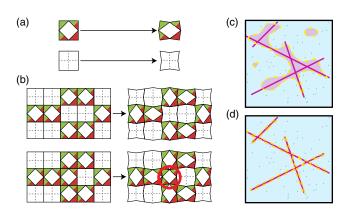

FIG. 1. (a) The building block of [51] can be tiled in two orientations (left) that have a distinct deformation in two dimensions (right). (b) The building blocks of (a) combine into larger designs (structures) that are either C (top) or I (bottom). A change of a single building block can frustrate the deformation (red circle) and change the structure from one that hosts a zero mode (a deformation that costs no energy) (C) to one that does not host a zero mode (I). A set of rules can be formulated for a unit cell design to have a zero mode [51]. (c) and (d) Conceptual configuration spaces of a discrete combinatorial metamaterial problem. Class C (pink lines) exists in a background of class I (blue), is sensitive to perturbations, and has a complex filamentous structure. Distinguishing between a network with a "coarse" decision boundary (purple dashed line) (c) versus a network with a "fine" decision boundary (d) is not possible with the test set (green dots) due to the undersampled C-I boundary.

TABLE I. Confusion matrices of trained CNNs with the lowest validation loss over the test set for the classification problems of Fig. 1(b) (M1), Fig. 2(d) (M2.i), and Fig. 2(e) (M2.ii).

		M1 predicted		M2.i predicted		M2.ii predicted	
		C	I	C	I	C	I
actual	C	19	0	685	1	43418	750
	I	0	4896	29	149265	453	105361

designs can be pictured as needles in a haystack [Figs. 1(c) and 1(d)] and crucially is determined by a set of implicit combinatorial rules. Unless we know these rules, these problems are typically computationally intractable.

Here we show that convolutional neural networks (CNNs) are able to accurately perform three distinct classifications of combinatorial mechanical metamaterials and to generalize to never-before-seen configurations. Crucially, we find that well-trained CNNs can capture the fine structure of the boundary of C, despite being trained on sparse datasets. These results suggest that CNNs implicitly learn the underlying rule-based structure of combinatorial problems. This opens up the possibility for using NNs for efficient exploration of the design space and inverse design when the combinatorial rules are unknown.

Coarse vs fine boundaries.—The boundary between C and I configurations has the shape of needles in a haystack. Therefore, in a randomly sampled training set, this boundary will be typically undersampled, e.g., the training set will contain few I close to C (see Supplemental Material [55]). We argue that a NN simply interpolating the training data will misclassify most I configurations close to C, resulting in a coarse decision boundary around C [Fig. 1(c)]. Instead, an ideal NN should approximate the fine structure of the needles more closely, resulting in a fine decision boundary around C [Fig. 1(d)]. While this may sound impossible, let us recall that this fine structure ultimately arises from combinatorial rules. These rules are in principle much simpler than the myriad of compatible configurations C they can generate. Hence, the question is whether NNs could implicitly learn these rules and finely approximate the fine boundary with great precision. Although a NN can classify perfectly the metamaterial M1 of Figs. 1(a) and 1(b) Table I), this is not sufficient to address this question because the dataset is too small and the C configurations are too rare to consider larger configurations (see Supplemental Material [55]).

Metamaterial classification.—Therefore, to see if NNs are still able to learn the structure of C if the C-I boundary is undersampled, we consider another combinatorial metamaterial M2 [54] [for details on how we define it, see Figs. 2(a) and 2(b)]. While metamaterial M1 had a unit cell of size $k \times k$ with $k = 1$, metamaterial M2 has larger unit cell size—we focus on $k = 5$ in the main text and cover the cases $k = 3$ to 8 in the Supplemental Material. For such a metamaterial, the design space is too large to fully map and
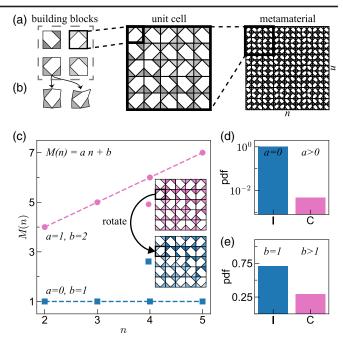

FIG. 2. (a) Four two-dimensional building blocks (left), combined into a square 5×5 unit cell (middle), which is tiled on a $n = 3$ grid, form a combinatorial metamaterial (right). (b) The building blocks feature two zero modes and four orientations with distinct deformations. (c) The number of zero modes $M(n)$ as function of n for two unit cells. The pink unit cell (circles) differs by a point mutation from the blue unit cell (squares), yet the pink unit cell has $a = 1$ and $b = 2$ and the blue unit cell has $a = 0$ and $b = 1$. Thus the pink unit cell is classified as class C for both classification problems while the blue unit cell is classified as class I for both problems. (d) Probability density function (pdf) for classification problem (ii). Class C is more rare than class I. (e) Probability density function for classification problem (i). Class C is much rarer than class I.

class C is rare, yet class C is abundant enough that we can create sufficiently large training sets to train NNs.

The number of zero modes $M(n)$ of a metamaterial consisting of $n \times n$ unit cells depends on the design of the unit cell: when the linear size n is increased, the number of zero modes $M(n)$ either grows linearly with n or saturates at a nonzero value [Fig. 2(c)] as $M(n) = an + b$, where a and b are positive integers. Accordingly, we can now specify two well-defined binary classification problems, which each feature a rare "compatible" (C) class and frequent "incompatible" (I) class [Figs. 2(d) and 2(e)]: (i) $a > 0$ (C) vs $a = 0$ (I). The metamaterial with $a > 0$ hosts zero modes that are organized along strips, for which one can formulate combinatorials rules (see Supplemental Material [55]); (ii) $b > 1$ (C) vs $b = 1$ (I). The metamaterial with $b > 1$ hosts additional zero modes—up to 6—that typically span the full structure and for which the rules still remain unknown despite our best efforts. In both classification problems, a single rotation of one building block in the unit cell can be sufficient to change class

[Fig. 2(c)]. Hence, the boundary between classes C and I is sharp and sensitive to minimal perturbations as in the case of metamaterial M1 [Fig. 1(c)].

If the rules are unknown, the classification of this metamaterial requires the determination of $M(n)$—via rank-revealing QR factorization [61]—as function of the number of unit cells n, which is computationally demanding. For $k \times k$ unit cells, the time it takes to compute this brute-force classification scales nearly cubically with input size k^2. In contrast, classification with NNs scales linearly with input size and is readily parallelizable. In practice this makes NNs invariant to input size due to computational overhead (see Supplemental Material [55]). Hence a trained NN allows for much more time-efficient exploration of the design space.

To train our NNs, we generate labeled data through Monte Carlo sampling the design space to generate 5×5 unit cells designs and explicitly calculate $M(n)$ for $n \in \{2, 3, 4\}$ to determine the classification. We do this for a range of $k \times k$ unit cells with $3 \leq k \leq 8$. We focus on 5×5 but the results hold for other unit cell sizes (see Supplemental Material [55]). The generated data is subsequently split into training (85%) and test (15%) sets. As our designs are spatially structured and local building block interactions drive compatible deformations, we ask whether convolutional neural networks (CNNs) are able to distinguish between class C and I. The input of our CNNs are pixelated representations of our designs. This approach facilitates the identification of neighboring building blocks that are capable of compatible deformations (see Supplemental Material [55]). The CNNs are trained using 10-fold stratified cross validation. Crucially, we use a balanced training set, where the proportion of class I has been randomly undersampled such that classes C and I are equally represented (see Supplemental Material [55]).

Despite the complexity of the classification problems, we find that the CNNs perform very well (Table I). In particular, the CNNs correctly classify most class C unit cells as class C, and most class I unit cells as class I. However, the test set is likely to contain few examples of class I close to the C-I boundary, especially as C becomes more rare (Fig. 1(c), see Supplemental Material [55]). Hence, whether our CNNs capture the complex boundary of C cannot be deduced from the test set alone. In other words, the CNNs find the needles in the haystack but it remains unclear whether the needles are approximated finely [Fig. 1(c)] or coarsely [Fig. 1(d)] [62].

Combinatorial structure.—To probe the shape of both the true set of C configurations and the set of classified C configurations, we start from a true class C configuration, perform random walks in configuration space, and at each step probe the probabilities to be in the set of true class C [Fig. 3(a)]. We randomly change the orientation of a single random building block at each step $s \mapsto s + 1$ and average over 1000 realizations (see Supplemental Material [55])
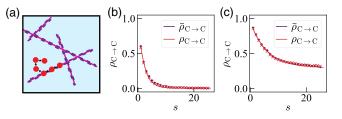

FIG. 3. (a) Example of a six-step random walk through design space (red dots) and sketch of the decision boundary of trained CNNs that has learned the combinatorial rules (purple dashed line). (b) Probabilities to remain in true and predicted class C under random walks of s steps, $\rho_{C \to C}(s)$ (red crosses) and fold-averaged $\langle \bar{\rho}_{C \to C} \rangle(s)$ (purple circles) with standard deviation (purple area), for classification (i) (left) and (ii) (right). The red continuous line is a least-squares fit to $\rho_{C \to C}(s)$ using Eq. (1).

The probability to remain in true class C, $\rho_{C \to C}(s)$, decreases with s and saturates to the class C volume fraction β for classification (i) and (ii) [Fig. 3(b)]. We note that we can fit this decay by a simple model, where we assume that subspace C is highly complex, so that the probabilities to leave it are uncorrelated. For every step, there is a chance α to remain C. Once in class I, we assume any subsequent steps are akin to uniformly probing the design space such that the probability to become C is equal to the C volume fraction β. Thus the probability to become C can be modeled as

$$\rho_{C \to C}(s) = \alpha^s + \beta(1 - \alpha^{s-1}). \tag{1}$$

The uncorrelated nature of the steps are consistent with a random needle structure [Fig. 1(c)], where the coefficient $\alpha \times 4^{5 \times 5}$ corresponds to the average dimensionality of the needles and β corresponds to their volume fraction. We can interpret α as the probability to not break the combinatorial rules when we randomly rotate a building block.

To see whether the CNNs are able to capture these key features of space C, we repeat our random walk procedure using the CNNs' classification instead, starting from true and classified C configurations, and obtain the probability $\bar{\rho}_{C \to C}(s)$. The decay of the fold-averaged $\langle \bar{\rho}_{C \to C} \rangle(s)$ closely matches that of the true class C for classification problems (i) and (ii) [Figs. 3(b) and 3(c)]. By fitting the predicted probability $\bar{\rho}_{C \to C}(s)$ for each fold to Eq. (1), using measurements of the CNN's predicted volume fraction $\bar{\beta}$ over the test set to constrain the fit, we obtain the fold-averaged dimensionality $\bar{\alpha}$. For classification (i) we find $\bar{\alpha} \approx 0.632 \pm 0.001$ closely matches the true $\alpha \approx 0.612 \pm 0.001$. In practice, α corresponds to the fraction of building blocks that are outside the relevant combinatorial strip. Using a simple counting argument, we find good agreement with the lower-bound of $\alpha \simeq 3/5$ (see Supplemental Material [63]). Similarly, for classification (ii) we find $\bar{\alpha} \approx 0.8514 \pm 0.0005$ closely matches $\alpha \approx 0.846 \pm 0.002$. Our results thus demonstrate that CNNs successfully capture on average the complex local
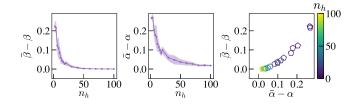
FIG. 4. (a) Difference between predicted class C volume $\bar{\beta}(n_h)$ and true class C volume β as a function of number of hidden neurons n_h shows that $\bar{\beta}(n_h)$ approaches β for increasing n_h. (b) Difference between predicted dimensionality $\bar{\alpha}(n_h)$ and true dimensionality α obtained through least-squares fits to Eq. (1) as a function of the number of hidden neurons n_h shows that $\bar{\alpha}(n_h)$ approaches α for increasing n_h. (c) Scatter plots of class volumes $\bar{\beta}(n_h) - \beta$ versus dimensionality $\bar{\alpha}(n_h) - \alpha$ shows that the latter asymptotes later than the former (n_h indicated by color bar). We use CNNs with a single convolution layer of 20 2×2 filters, which are spatially offset with respect to the unit cell and subsequently flattened. The flattened feature maps are fully connected to a layer of n_h hidden neurons, which itself is fully connected to two output neurons that correspond to class C and I. The CNNs are systematically trained using 10-fold stratified cross validation for varying numbers of hidden neurons n_h.

shape of the combinatorial space C. Even though during learning the algorithm "sees" very few class I unit cells that are close the C-I boundary, the decision boundary still captures on average the sparsity and fine structure of the class C subset. Thus we conclude that the CNNs infer the combinatorial rules [Fig. 1(c)], rather than interpolate the shape in high dimensional design space [Fig. 1(d)]. In other words, CNNs are able not only to capture accurately the volume fraction of the needles, but also to finely distinguish between needle and hay.

Volume before structure.—But what happens with smaller CNNs? We focus on classification (i) and probe how well our CNNs—which consist of a single 20 filters convolution layer, a single n_h neurons hidden layer, and a two neurons output layer—capture the sparsity and structure of class C. First we compare their true and predicted volumes β and $\bar{\beta}(n_h)$ as a function of the number of hidden neurons n_h. The CNNs' predicted class C volume fraction $\bar{\beta}$ approaches the true class C volume fraction β as the number of hidden neurons n_h increases sufficiently, despite their balanced training set [Fig. 4(a)] [55]. Next we compare the true and predicted dimensionality α and $\bar{\alpha}(n_h)$. While for small values of n_h, $\bar{\alpha}$ overestimates α, $\bar{\alpha}$ closely matches α for large n_h [Fig. 4(b)]. For small number of hidden neurons n_h, the CNNs overestimate both the probability to remain in class C and the rarity of class C; in other words, small CNNs coarsen the complex shape of C [Fig. 1(c)]. As seen above, for larger number of hidden neurons n_h both the probability and rarity of C are closely approximated, thus large CNNs finely capture the complex shape of C [Fig. 1(d)].

Strikingly, we observe that the predicted class C volume $\bar{\beta}$ more quickly reaches its asymptotic value than the

dimensionality $\bar{\alpha}$. To see this, we plot $\bar{\beta}(n_h) - \beta$ versus $\bar{\alpha}(n_h) - \alpha$, which demonstrates that after $\bar{\beta}$ closely approximates β, increasing the number of hidden neurons n_h improves $\bar{\alpha}(n_h)$ towards its asymptote α [Fig. 4(c)]—this observation is also present for other unit cell sizes, see Supplemental Material [55]. Thus, further increasing the size of the CNN beyond the point of marginal gain of test set performance results in a significantly more closely captured fine structure of C. In other words, to correctly capture the average dimensionality of the needles requires more neurons than to capture their volume.

Discussion.—NNs are known to be universal approximators [64] and efficient classifiers. They often generalize well when the training data samples representative portions of the input space sufficiently, even for nonsmooth [63] or noisy data [65]. As combinatorial problems are sharply delineated and severely class imbalanced, one expects that the fine details of an undersampled complex boundary would be blurred by NNs. Surprisingly, we have shown that CNNs will closely approximate such a complex combinatorial structure, despite being trained on a sparse training set. We attribute this to the underlying set of rules which govern the complex space of compatible configurations— in simple terms, the CNN learns the combinatorial rules, rather than the geometry of design space, which is the complex result of those rules [66].

Recognizing NNs' ability to learn these rules from a sparse representation of the design space opens new strategies for design. For instance, our CNNs could be readily used as surrogate models within a design algorithm to save computational time. Alternatively, one could instead devise a design algorithm based on generative adversarial NNs [67] or variational auto encoders [68]. It is an open question whether and how such generative models could successfully leverage the learning of combinatorial rules [69].

Our Letter shows that metamaterials provide a compelling avenue for machine learning combinatorial problems, as they are straightforward to simulate yet exhibit complex combinatorial structure [Fig. 1(c)]. More broadly, applying neural networks to combinatorial problems opens many exciting questions. What is the relation between the complexity of the combinatorial rules and that of the networks? Can unsolved combinatorial problems be solved by neural networks? Can neural networks learn size-independent combinatorial rules? Conversely, can these problems help us understand why neural networks work so well [70]? Can they provide insight in how to effectively overcome strong data imbalance [71]? We believe combinatorial metamaterials are well suited to answer such questions.

The code supporting the findings reported in this Letter is publicly available on GitLab [72,73] and the data on Zenodo [74,75].

* r.vanmastrigt@uva.nl

[1] J. Hoffmann, Y. Bar-Sinai, L. M. Lee, J. Andrejevic, S. Mishra, S. M. Rubinstein, and C. H. Rycroft, Machine learning in a data-limited regime: Augmenting experiments with synthetic data uncovers order in crumpled sheets, Sci. Adv. **5**, 6792 (2019).

[2] J. Colen, M. Han, R. Zhang, S. A. Redford, L. M. Lemma, L. Morgan, P. V. Ruijgrok, R. Adkins, Z. Bryant, Z. Dogic *et al.*, Machine learning active-nematic hydrodynamics, Proc. Natl. Acad. Sci. U.S.A. **118**, 10 (2021).

[3] M. J. Falk, V. Alizadehyazdi, H. Jaeger, and A. Murugan, Learning to control active matter, Phys. Rev. Res. **3**, 033291 (2021).

[4] A. R. Dulaney and J. F. Brady, Machine learning for phase behavior in active matter systems, Soft Matter **17**, 6808 (2021).

[5] Y. Bar-Sinai, S. Hoyer, J. Hickey, and M. P. Brenner, Learning data-driven discretizations for partial differential equations, Proc. Natl. Acad. Sci. U.S.A. **116**, 15344 (2019).

[6] P. R. Wiecha, A. Arbouet, C. Girard, and O. L. Muskens, Deep learning in nano-photonics: Inverse design and beyond, Photonics Res. **9**, B182 (2021).

[7] W. Ma, Z. Liu, Z. A. Kudyshev, A. Boltasseva, W. Cai, and Y. Liu, Deep learning for the design of photonic structures, Nat. Photonics **15**, 77 (2021).

[8] Y. Xu, X. Zhang, Y. Fu, and Y. Liu, Interfacing photonics with artificial intelligence: an innovative design strategy for photonic structures and devices based on artificial neural networks, Photonics Res. **9**, B135 (2021).

[9] M. Harrington, A. J. Liu, and D. J. Durian, Machine learning characterization of structural defects in amorphous packings of dimers and ellipses, Phys. Rev. E **99**, 022903 (2019).

[10] M. Mozaffar, R. Bostanabad, W. Chen, K. Ehmann, J. Cao, and M. Bessa, Deep learning predicts path-dependent plasticity, Proc. Natl. Acad. Sci. U.S.A. **116**, 26414 (2019).

[11] M. A. Bessa, P. Glowacki, and M. Houlder, Bayesian machine learning in metamaterial design: Fragile becomes supercompressible, Adv. Mater. **31**, 1904845 (2019).

[12] G. X. Gu, C.-T. Chen, and M. J. Buehler, De novo composite design based on machine learning algorithm, Extreme Mech. Lett. **18**, 19 (2018).

[13] L. Wang, Y.-C. Chan, F. Ahmed, Z. Liu, P. Zhu, and W. Chen, Deep generative modeling for mechanistic-based learning and design of metamaterial systems, Comput. Methods Appl. Mech. Eng. **372**, 113377 (2020).

[14] G. M. Coli, E. Boattini, L. Filion, and M. Dijkstra, Inverse design of soft materials via a deep learning-based evolutionary strategy, Sci. Adv. **8**, eabj6731 (2022).

[15] J.-H. Bastek, S. Kumar, B. Telgen, R. N. Glaesener, and D. M. Kochmann, Inverting the structure-property map of truss metamaterials by deep learning, Proc. Natl. Acad. Sci. U.S.A. **119**, 1 (2022).

[16] D. Shin, A. Cupertino, M. H. de Jong, P. G. Steeneken, M. A. Bessa, and R. A. Norte, Spiderweb nanomechanical resonators via Bayesian optimization: Inspired by nature and guided by machine learning, Adv. Mater. **34**, 2106248 (2022).

[17] P. Z. Hanakata, E. D. Cubuk, D. K. Campbell, and H. S. Park, Forward and inverse design of kirigami via supervised autoencoder, Phys. Rev. Res. **2**, 042006(R) (2020).

[18] A. E. Forte, P. Z. Hanakata, L. Jin, E. Zari, A. Zareei, M. C. Fernandes, L. Sumner, J. Alvarez, and K. Bertoldi, Inverse design of inflatable soft membranes through machine learning, Adv. Funct. Mater. **32**, 2111610 (2022).

[19] E. D. Cubuk, S. S. Schoenholz, J. M. Rieser, B. D. Malone, J. Rottler, D. J. Durian, E. Kaxiras, and A. J. Liu, Identifying Structural Flow Defects in Disordered Solids Using Machine-Learning Methods, Phys. Rev. Lett. **114**, 108001 (2015).

[20] V. Bapst, T. Keck, A. Grabska-Barwińska, C. Donner, E. D. Cubuk, S. S. Schoenholz, A. Obika, A. W. Nelson, T. Back, D. Hassabis *et al.*, Unveiling the predictive power of static structure in glassy systems, Nat. Phys. **16**, 448 (2020).

[21] S. S. Schoenholz, E. D. Cubuk, D. M. Sussman, E. Kaxiras, and A. J. Liu, A structural approach to relaxation in glassy liquids, Nat. Phys. **12**, 469 (2016).

[22] Y.-T. Hsu, X. Li, D.-L. Deng, and S. Das Sarma, Machine Learning Many-Body Localization: Search for the Elusive Nonergodic Metal, Phys. Rev. Lett. **121**, 245701 (2018).

[23] J. Venderley, V. Khemani, and E.-A. Kim, Machine Learning Out-of-Equilibrium Phases of Matter, Phys. Rev. Lett. **120**, 257204 (2018).

[24] K. Swanson, S. Trivedi, J. Lequieu, K. Swanson, and R. Kondor, Deep learning for automated classification and characterization of amorphous materials, Soft Matter **16**, 435 (2020).

[25] R. van Damme, G. M. Coli, R. van Roij, and M. Dijkstra, Classifying crystals of rounded tetrahedra and determining their order parameters using dimensionality reduction, ACS Nano **14**, 15144 (2020).

[26] C. Miles, A. Bohrdt, R. Wu, C. Chiu, M. Xu, G. Ji, M. Greiner, K. Q. Weinberger, E. Demler, and E.-A. Kim, Correlator convolutional neural networks as an interpretable architecture for image-like quantum matter data, Nat. Commun. **12**, 1 (2021).

[27] N. Andrejevic, J. Andrejevic, C. H. Rycroft, and M. Li, Machine learning spectral indicators of topology, arXiv:2003.00994.

[28] J. Carrasquilla and R. G. Melko, Machine learning phases of matter, Nat. Phys. **13**, 431 (2017).

[29] E. P. Van Nieuwenburg, Y.-H. Liu, and S. D. Huber, Learning phase transitions by confusion, Nat. Phys. **13**, 435 (2017).

[30] D.-L. Deng, X. Li, and S. Das Sarma, Machine learning topological states, Phys. Rev. B **96**, 195145 (2017).

[31] Y. Zhang and E.-A. Kim, Quantum Loop Topography for Machine Learning, Phys. Rev. Lett. **118**, 216401 (2017).

[32] P. Zhang, H. Shen, and H. Zhai, Machine Learning Topological Invariants with Neural Networks, Phys. Rev. Lett. 120, 066401 (2018).

[33] Y. Zhang, A. Mesaros, K. Fujita, S. Edkins, M. Hamidian, K. Ch'ng, H. Eisaki, S. Uchida, J. S. Davis, E. Khatami et al., Machine learning in electronic-quantum-matter imaging experiments, Nature (London) 570, 484 (2019).

[34] K. Ch'ng, J. Carrasquilla, R. G. Melko, and E. Khatami, Machine Learning Phases of Strongly Correlated Fermions, Phys. Rev. X 7, 031038 (2017).

[35] B. S. Rem, N. Käming, M. Tarnowski, L. Asteria, N. Fläschner, C. Becker, K. Sengstock, and C. Weitenberg, Identifying quantum phase transitions using artificial neural networks on experimental data, Nat. Phys. 15, 917 (2019).

[36] A. Bohrdt, C. S. Chiu, G. Ji, M. Xu, D. Greif, M. Greiner, E. Demler, F. Grusdt, and M. Knap, Classifying snapshots of the doped hubbard model with machine learning, Nat. Phys. 15, 921 (2019).

[37] J. Carrasquilla, Machine learning for quantum matter, Adv. Phys. 5, 1797528 (2020).

[38] E. van Nieuwenburg, E. Bairey, and G. Refael, Learning phase transitions from dynamics, Phys. Rev. B 98, 060301(R) (2018).

[39] A. Bohrdt, S. Kim, A. Lukin, M. Rispoli, R. Schittko, M. Knap, M. Greiner, and J. Léonard, Analyzing Nonequilibrium Quantum States through Snapshots with Artificial Neural Networks, Phys. Rev. Lett. 127, 150504 (2021).

[40] H. Y. Sigaki, E. K. Lenzi, R. S. Zola, M. Perc, and H. V. Ribeiro, Learning physical properties of liquid crystals with deep convolutional neural networks, Sci. Rep. 10, 7664 (2020).

[41] P. Geiger and C. Dellago, Neural networks for local structure detection in polymorphic systems, J. Chem. Phys. 139, 164105 (2013).

[42] C. Dietz, T. Kretz, and M. H. Thoma, Machine-learning approach for local classification of crystalline structures in multiphase systems, Phys. Rev. E 96, 011301(R) (2017).

[43] L.-F. Zhang, L.-Z. Tang, Z.-H. Huang, G.-Q. Zhang, W. Huang, and D.-W. Zhang, Machine learning topological invariants of non-Hermitian systems, Phys. Rev. A 103, 012419 (2021).

[44] G. M. Coli and M. Dijkstra, An artificial neural network reveals the nucleation mechanism of a binary colloidal ab_{13} crystal, ACS Nano 15, 4335 (2021).

[45] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko et al., Highly accurate protein structure prediction with alphafold, Nature (London) 596, 583 (2021).

[46] E. Gazit, Self-assembled peptide nanostructures: The design of molecular building blocks and their technological utilization, Chem. Soc. Rev. 36, 1263 (2007).

[47] A. Levin, T. A. Hakala, L. Schnaider, G. J. Bernardes, E. Gazit, and T. P. Knowles, Biomimetic peptide self-assembly for functional materials, Nat. Rev. Chem. 4, 615 (2020).

[48] T. Hull, The combinatorics of flat folds: A survey, in Origami3: Proceedings of the 3rd International Meeting of Origami Science, Math, and Education (A. K. Peters, Natick, 2002), pp. 29–38.

[49] P. Dieleman, N. Vasmel, S. Waitukaitis, and M. van Hecke, Jigsaw puzzle design of pluripotent origami, Nat. Phys. 16, 63 (2020).

[50] E. D. Demaine and M. L. Demaine, Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity, Graphs Comb. 23, 195 (2007).

[51] C. Coulais, E. Teomy, K. De Reus, Y. Shokef, and M. Van Hecke, Combinatorial design of textured mechanical metamaterials, Nature (London) 535, 529 (2016).

[52] A. S. Meeussen, E. C. Oğuz, Y. Shokef, and M. van Hecke, Topological defects produce exotic mechanics in complex metamaterials, Nat. Phys. 16, 307 (2020).

[53] C. Coulais, A. Sabbadini, F. Vink, and M. van Hecke, Multi-step self-guided pathways for shape-changing metamaterials, Nature (London) 561, 512 (2018).

[54] A. Bossart, D. M. Dykstra, J. van der Laan, and C. Coulais, Oligomodal metamaterials with multifunctional mechanics, Proc. Natl. Acad. Sci. U.S.A. 118, 21 (2021).

[55] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.129.198003 for more details on the undersampled C-I boundary in the training sets; on the design rules and rarity of the metamaterial in Fig. 1; of and numerical evidence for combinatorial rules of classification (i), which includes Refs. [56–58]; of the computational time comparison; for CNN results of more unit cell sizes; of the pixel representation; about the training and test sets for each metamaterial, which includes Refs. [59,60]; of the random walks. To see that $\mathrm{BA}(n_h)$ increases in conjunction with $\bar{\beta}(n_h)$; measurements of $\bar{\alpha}$ and $\bar{\beta}$ of classification problem (i) for more unit cell sizes.

[56] J. C. Maxwell, L. On the calculation of the equilibrium and stiffness of frames, London, Edinburgh, Dublin Philos. Mag. J. Sci., 27, 294 (1864).

[57] J. N. Grima, A. Alderson, and K. Evans, Auxetic behaviour from rotating rigid units, Phys. Status Solidi (b) 242, 561 (2005).

[58] C. Coulais, C. Kettenis, and M. van Hecke, A characteristic length scale causes anomalous size effects and boundary programmability in mechanical metamaterials, Nat. Phys. 14, 40 (2018).

[59] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv:1412.6980.

[60] M. Hossin and M. N. Sulaiman, A review on evaluation metrics for data classification evaluations, Int. J. Data Mining Knowl. Manag. Process 5, 1 (2015).

[61] Y. P. Hong and C.-T. Pan, Rank-revealing QR factorizations and the singular value decomposition, Math. Comput. 58, 213 (1992).

[62] We have observed from qualitative analysis of the 29 falsely classified C unit cells of M2.i that these unit cells can be transformed to true class C by changing a single or a few building blocks, thus they are close to the C-I boundary in design space.

[63] M. Imaizumi and K. Fukumizu, Deep neural networks learn non-smooth functions effectively, in Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics, Proceedings of Machine Learning Research, edited by K. Chaudhuri and M. Sugiyama (PMLR, 2019), Vol. 89, pp. 869–878.

[64] K. Hornik, M. Stinchcombe, and H. White, Multilayer feedforward networks are universal approximators, Neural Netw. 2, 359 (1989).

[65] D. Rolnick, A. Veit, S. Belongie, and N. Shavit, Deep learning is robust to massive label noise, arXiv:1705.10694.

[66] We expect NNs to work beyond combinatorial metamaterials for a wide range of combinatorial problems in physics, such as spin ice. These combinatorial rules in such problems can typically be translated to matrix operations, NNs naturally capture such matrix operations, and therefore we expect them to perform well.

[67] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, Generative adversarial nets, Adv. Neural Inf. Process. Syst. **27**, 2672 (2014).

[68] D. P. Kingma and M. Welling, Auto-encoding variational bayes, arXiv:1312.6114.

[69] Y. Bengio, A. Lodi, and A. Prouvost, Machine learning for combinatorial optimization: a methodological tour d'horizon, Eur. J. Oper. Res. **290**, 405 (2021).

[70] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, Understanding deep learning (still) requires rethinking generalization, Commun. ACM **64**, 107 (2021).

[71] J. M. Johnson and T. M. Khoshgoftaar, Survey on deep learning with class imbalance, J. Big Data **6**, 1 (2019).

[72] See https://uva-hva.gitlab.host/published-projects/Combi-MetaMaterial for code to calculate zero modes and numerically check design rules.

[73] See https://uva-hva.gitlab.host/published-projects/CNN_MetaCombi for code to train and evaluate convolutional neural networks.

[74] R. van Mastrigt, M. Dijkstra, M. van Hecke, and C. Coulais, Zero modes and classification of combinatorial metamaterials (2022), 10.5281/zenodo.7070963.

[75] R. van Mastrigt, M. Dijkstra, M. van Hecke, and C. Coulais, Convolutional neural networks for classifying combinatorial metamaterials (2022), 10.5281/zenodo.7071282.