Forward and inverse design of kirigami via supervised autoencoder

Paul Z. Hanakata,[1,*] Ekin D. Cubuk,[2] David K. Campbell [⬤],[3] and Harold S. Park[4]

[1]*Department of Physics, Harvard University, Cambridge, Massachusetts 02138, USA*
[2]*Google Brain, Mountain View, California 94043, USA*
[3]*Department of Physics, Boston University, Boston, Massachusetts 02215, USA*
[4]*Department of Mechanical Engineering, Boston University, Boston, Massachusetts 02215, USA*

Machine learning (ML) methods have recently been used as forward solvers to predict the mechanical properties of composite materials. Here, we use a supervised autoencoder (SAE) to perform the inverse design of graphene kirigami, where predicting the ultimate stress or strain under tensile loading is known to be difficult due to nonlinear effects arising from the out-of-plane buckling. Unlike the standard autoencoder, our SAE is able not only to reconstruct cut configurations but also to predict the mechanical properties of graphene kirigami and classify the kirigami with either parallel or orthogonal cuts. By interpolating in the latent space of kirigami structures, the SAE is able to generate designs that mix parallel *and* orthogonal cuts, despite being trained independently on parallel or orthogonal cuts. Our method allows us to both identify alternate designs and predict, with reasonable accuracy, their mechanical properties, which is crucial for expanding the search space for materials design.

Introduction. Recently, there has been growing interest in investigating the nonlinear mechanics of perforated thin sheets across length scales ranging from the macroscale [1–4] down to nanoscale systems [5–8]. The cuts in a thin sheet—known as kirigami cuts—induce buckling and other motions (e.g., rotations). These mechanisms result in new properties, such as enhanced ductility [7] and auxeticity [9] that are different from the pristine (cut-free) counterpart. This simple strategy has led to programable kirigami actuators which are the building blocks of soft robots [1,10]. While many analytic descriptions have been developed to understand the changes in mechanical behavior due to the cuts [1,2,4], these analytic approaches are used to describe systems with repeating and uniform cut patterns or to optimize a specific target property. An analytical model that can describe how the mechanical properties of kirigami sheets depend on the interaction of *different* types of cuts still needs to be developed.

For the *inverse design* problem, one ongoing challenge for kirigami structures is in designing them to achieve specific properties. Most current machine learning (ML) techniques rely on applying ML to select top candidates from a *fixed* library [8,11,12]. The usual approach is to perform "active learning" where the model is trained incrementally with data proposed by the ML [8,11], or by training the model with a significant amount of data to predict top candidates [12]. For

----

*paul.hanakata@gmail.com

both approaches, ML (the "forward solver") must be applied to the *entire* library. Even when the computational cost of the ML approach is much lower than the ground truth data generator (physics-based simulations or experimental data), in a highly complex system with many degrees of freedom, it is not practical to use ML to calculate the properties of all candidates to find the best candidates.

In computer vision problems, *generative models* have shown to be successful in generating realistic synthetic samples [13]. Unlike supervised learning, the generative models are trained to capture an underlying data distribution and to learn important features. For instance, variational autoencoders have been used to capture the important information from a high-dimensional space of the real representation (e.g., image) within a lower-dimensional space, known as the *latent space*. The latent vectors capture important features, for instance, smiles in facial images, and thus can be used for interpolation which is useful for generating new synthetic samples.

In optimizing material properties, we often have some key observable properties, such as ultimate stress and fracture strain. The goal is to make the learned hidden (latent) variables correlated to the key properties, so that we can perform optimization in the *latent space*, which has a significantly reduced dimensionality compared to that of the discrete (original) representation of the structures. This strategy gives a large advantage over performing optimization in the original representation space, and has recently been applied for designing materials with a large design space, such as drugs, organic molecules, and optical metamaterials [14–17].

In this Rapid Communication we propose a *supervised autoencoder* (SAE) for inverse structural design. We set up our training such that we can evaluate the effectiveness of

interpolation (generating different designs) within and outside the training domain. First, we find that the SAE is able to generate designs consisting of mixed cuts even though the SAE is trained with kirigami structures with only parallel and orthogonal cuts, which shows the ability of SAE to perform interpolation in the latent space. Moreover, in the latent space, the SAE captures similarities and differences between distinct structures with different cut types whereas the information about cut types is not provided during the training. As generalization requires diversity in the training set, we can leverage the ability of the SAE to distinguish different structures in the latent space to use it as an exploration strategy to propose designs that lie outside the training data.

Supervised autoencoder. An autoencoder (AE) consists of two parts: (i) an encoder $\mathcal{E}$ that maps a vector to a reduced representation and (ii) a decoder $\mathcal{D}$ that reconstructs a vector to its original representation from the reduced representation. Let $x \in \mathbb{R}^n = \mathcal{X}$ be the n-dimensional vector, and $z = (z_0, z_1, \ldots, z_{p-1}) \in \mathbb{R}^p = \mathcal{Z}$ be the p-dimensional latent variables. Since the goal is to have a compressed representation, p is chosen to be smaller than n. Mathematically we can write this transformation as $\mathcal{E}: \mathcal{X} \to \mathcal{Z}$, $\mathcal{D}: \mathcal{Z} \to \mathcal{X}$. In the standard AE the mean reconstruction loss is given by

$$\mathcal{L}^{\mathcal{X}}(x, x') = \frac{1}{m} \sum_{i=1}^{m} |x^{(i)} - x'^{(i)}|^2$$
$$= \frac{1}{m} \sum_{i=1}^{m} |x^{(i)} - \mathcal{D}[\mathcal{E}(x^{(i)})]|^2, \quad (1)$$

where i is the ith data point and m is the number of training samples. AEs are widely used for unsupervised learning where only unlabeled data x are provided. In many physical systems, we want to include the known properties to the unsupervised AE such that the supervised AE (SAE) learns new "hidden" features. In this work, we choose p to be 10 and we enforce the first two latent vectors (z_1 and z_0) to learn ultimate (maximum) stress σ^u and the corresponding ultimate strain ϵ^u. We choose a latent dimension that is larger than the number of known properties since kirigami with different cuts can have the same mechanical properties (e.g., due to symmetries). Thus, for this proposed supervised AE (SAE) architecture, we introduce a target property mean-squared error loss function,

$$\mathcal{L}^{\mathcal{Y}}(y, z) = \frac{1}{m} \sum_{i=1}^{m} |z^{(i)} - y^{(i)}|^2 = \frac{1}{m} \sum_{i=1}^{m} \sum_{k=0}^{d-1} |z_k^{(i)} - y_k^{(i)}|^2, \quad (2)$$

where $y \in \mathbb{R}^d = \mathcal{Y}$ is a d-dimensional vector that contains the known properties and d equals the number of observable properties. The total loss function then becomes $\mathcal{L} = \mathcal{L}^{\mathcal{X}} + \eta \mathcal{L}^{\mathcal{Y}}$, where η is a hyperparameter. We standardize y to have zero mean and unit standard deviation [18], which is essential for training a neural network as the optimizer treats all directions uniformly in the parameter space [19,20].

Here, we used the typical AE architecture [21], where a schematic of the SAE is shown in Fig. 1(a). For the encoder, we use a deep neural network (DNN) architecture similar to our previous work [8] with one additional fully connected layer (FCL). The decoder consists of two fully connected layers. More details of the DNN architecture and training
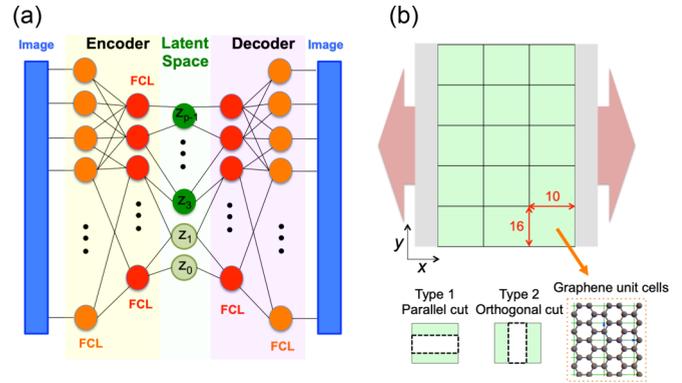
FIG. 1. (a) Schematic of an autoencoder. (b) Schematic of graphene kirigami partitioned into 3×5 grids. The training set contains either parallel or orthogonal cuts (no mixing of cut types). Each grid consists of 10×16 graphene unit cells. Kirigami is stretched in the x direction by moving the edges (gray regions).

procedure can be found in the Supplemental Material (SM) [22].

Results. We train the SAE with configurations having parallel cuts, i.e., that are parallel to the loading direction (x axis), and orthogonal cuts (y axis), as shown in Fig. 1(b). Each orthogonal cut has a size of 3×16 unit cells (holes), whereas each parallel cut has a size of 3×10 unit cells (holes). We trained the SAE with configurations having between 0 and 15 cuts. Each graphene membrane has 2400 unit cells and we define the density ρ as the number of *holes* divided by the total number of unit cells. This gives a range of density from 1 (0 cuts) to 0.7 (15 cuts or equivalently 720 holes) [23]. We used LAMMPS (Large-scale Atomic/Molecular Massively Parallel Simulator) to simulate graphene kirigami under tension [24]. The molecular dynamics (MD) simulation procedure is similar to our previous work [8] and the simulation details can found in the SM. The SAE takes an image of size 2400 (30×80) and outputs an image with the same size. While we train the SAE with configurations having large cuts ($\sim$30 holes in each grid), in principle, the SAE can generate configurations with any arbitrary cut size, i.e., as small as one hole. We simulated all possible configurations of parallel and orthogonal cuts *without* mixing the two types. As we allow either only 0–15 orthogonal cuts or 0–15 parallel cuts, we obtain a total of 62 558 configurations, of which 29 791 are nondetached configurations with orthogonal cuts while the remaining are the configurations with parallel cuts which have nondetached configurations. The networks were trained with 50% of the data set while the remainder of the data set is used for validation and a test set (25% each).

We first show that the mechanical properties of cut graphene are indeed highly dependent on the material architecture. In some composite designs, the stiffness of materials can be well described by density. In contrast for kirigami, the nonlinear regime becomes important and thus predicting properties beyond the linear regime, such as ultimate stress and yield strain, via density is no longer viable [4,7,8]. For instance, in typical kirigami geometries, the effective stiffness in the postbuckling regime is proportional to bending rigidity as opposed to the Young's modulus [4,25].
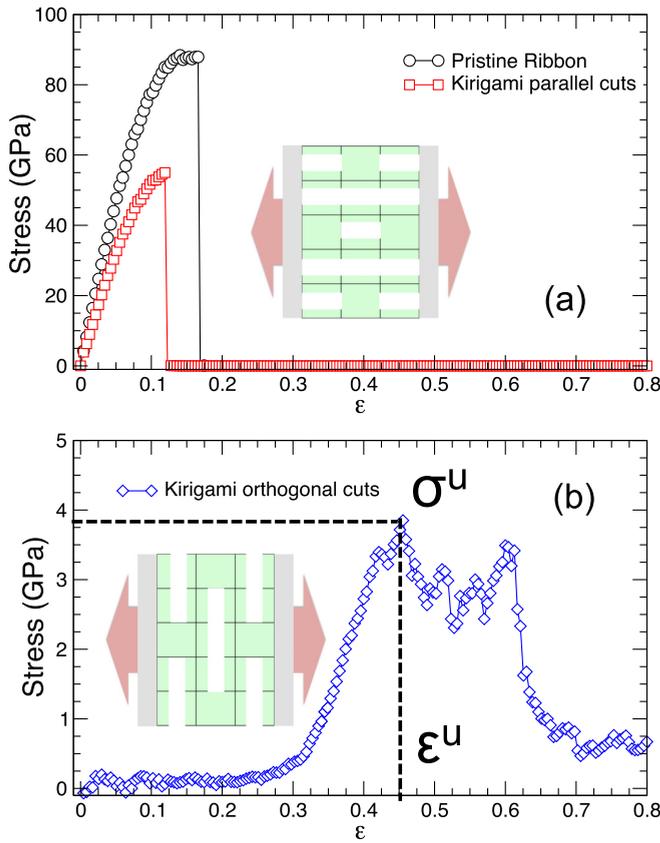
FIG. 2. Stress as a function of strain for pristine graphene ribbon and kirigami with parallel cuts (a) and kirigami with orthogonal cuts (b). The stress-strain curve changes significantly as the orientation of the cuts is changed.

Thus, the *architecture* of the materials strongly impacts their properties. To demonstrate this, in Figs. 2(a) and 2(b),

we plot stress-strain curves of kirigami with parallel and orthogonal cuts. Importantly, the orthogonal cuts in Fig. 2(b) represent the same cut pattern with the same number of cuts as shown for the parallel cut kirigami in Fig. 2(a). As can be seen, simply changing the orientation of each cut but fixing the cut configurations results in a completely different stress-strain curve, consistent with MD simulations by Ref. [26]. Furthermore, we can see from Fig. 3(a) that density alone does not correlate to σ^u or ϵ^u. This further suggests that the desired global properties are highly dependent on the structural configuration.

To summarize, the mechanical properties of graphene kirigami depend not only on (i) material density but also on (ii) cut configurations, and on (iii) cut orientations. We will show that despite this complexity our SAE is able to organize the materials based on the structural properties that are not encoded to the latent space in a supervised fashion.

Next, we investigate the learned latent variables. We first turn off the constraint $\mathcal{L}^Y$ ($\eta = 0$) to enable the AE to learn in an unsupervised manner. To better visualize the ten-dimensional (10D) latent space we project the latent vectors to a 2D space using a principal component analysis (PCA). The latent vectors are generated by passing x of the training data through the encoder. From Fig. 3 we see that the 2D projected latent variables (from the training data) clearly separate the two different cut orientations despite the fact that the AE was not provided with the cut orientations. In addition to separating structures based on cut orientation, the AE clusters different structures based on their density. This is similar to how a latent variable found by AE coincides with the net magnetization (the order parameter) in the Ising spin system [27]. However, none of the latent variables found by the AE strongly correlates to either σ^u or ϵ^u in this kirigami problem.

We now include the property predictions into the latent space by setting $\eta = 1.0$ [28]. Similar to the unsupervised AE, as shown in Fig. 3, the SAE clusters the data based on
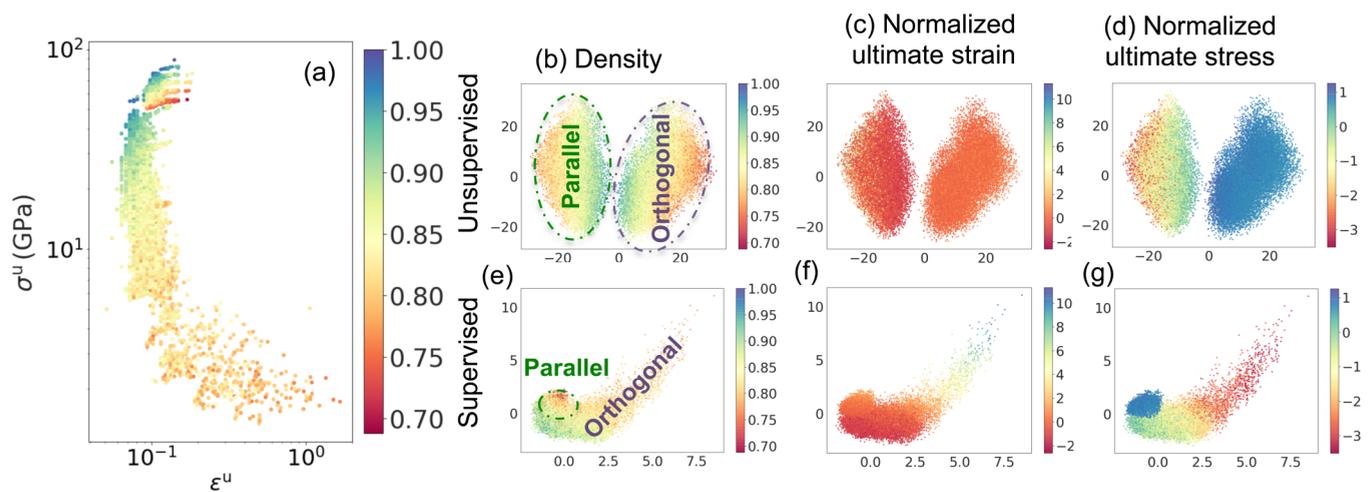
FIG. 3. (a) Log-log plot of ultimate stress as a function of ultimate strain for *all* simulated data. The color bar represents the density, where density alone does not correlate with ultimate strain. Projected latent space for (b)–(d) unsupervised autoencoder and (e)–(g) supervised autoencoder. The latent vectors are generated by passing x of the training data through the encoder. The two axes are found by PCA. The unsupervised autoencoder is able to distinguish different structures based on their cut density and cut orientations. The supervised autoencoder successfully captures not only the structural differences but also mechanical properties such as normalized ultimate strain and normalized ultimate stress.

cut orientation. Furthermore, by color coding the data by the normalized ultimate strain $\overline{\epsilon^{\mathrm{u}}}$, and normalized ultimate stress $\overline{\sigma^{\mathrm{u}}}$, we see that in each phase the SAE organizes structures based on their properties. This shows that the SAE has not only learned to distinguish different structures of the input image in the real representation but also to predict their mechanical properties. We use the R^2 metric to quantify the performance of the model in predicting $\overline{\epsilon^{\mathrm{u}}}$ and $\overline{\sigma^{\mathrm{u}}}$ as we did in our previous work [8]. The R^2 on the training, validation, test sets for σ^{u} (ϵ^{u}) are 0.99 (0.92), 0.99 (0.87), and 0.99 (0.87), respectively. Thus we indeed find that z_0 and z_1 are correlated to the normalized ϵ^{u} and σ^{u}, respectively. Similarly, the SAE is successful in reconstructing the structures x in the real space. We use the fraction of correctly placed graphene unit cells as an accuracy metric and we obtain accuracies of 99.4% for training, validation, and a test set. Details regarding the distribution of all latent variables and the reconstructed structures of the can be found in the SM. For the remainder of the Rapid Communication we will focus on the SAE.

Generating designs via interpolation in the latent space. While the SAE can be used to generate designs by sampling from the latent space, the question remains as to how the latent values $(z_0, \ldots, z_{p-1})$ are set as they all reside in the same space and are interconnected. Another simple approach is to perform interpolation in the latent space. In this section, we introduce metrics to quantify different designs and show that we can generate designs while simultaneously predicting their mechanical properties with reasonable accuracy. The question we want to address here is what objective function should be chosen in order to generate kirigami designs that were not in the training data.

In a p-dimensional space, we can write $z = (z_0, z_1, \ldots, z_{p-1})$ in terms of a radius r and $p - 1$ angles $(\phi_0, \phi_1, \ldots, \phi_{p-2})$. By analogy to a genetic algorithm, designs (children) can be generated by combining two parents and applying a mutation rule. This approach is usually performed in the real representation of the genome. In the current work, we generate designs from the *latent space*, which is much smaller than the real space. The simplest approach to generate designs is by performing linear interpolation between two latent vectors. Here, we use *spherical linear interpolation* (SLERP), which has been used for interpolating images in generative networks [29,30]. Suppose we have two parent vectors $\mathbf{v}_\alpha, \mathbf{v}_\beta \in \mathcal{Z}$, then a new vector can be generated $\mathbf{v}_t = \frac{\sin[(1-t)\Omega_{\alpha\beta}]}{\sin \Omega_{\alpha\beta}}\mathbf{v}_\alpha + \frac{\sin[t\Omega_{\alpha\beta}]}{\sin \Omega_{\alpha\beta}}\mathbf{v}_\beta$, where $0 \leqslant t \leqslant 1$ and $\Omega_{\alpha\beta} = \cos^{-1}\frac{\mathbf{v}_\alpha \cdot \mathbf{v}_\beta}{|\mathbf{v}_\alpha||\mathbf{v}_\beta|}$. With this approach the interpolated vectors then can be decoded into a real structure. Note that in the limit $\Omega_{\alpha\beta} \to 0$ SLERP becomes linear interpolation.

As our goal is to perform inverse design outside the training domain, an important step is to quantify similarity. We use angular distance (Ω_{tk}), and Tanimoto similarity (T_{tk}) to quantify the difference between the interpolated structure and the parent structure,

$$\Omega_{tk} = \cos^{-1}\frac{\mathbf{v}_t \cdot \mathbf{v}_k}{|\mathbf{v}_t||\mathbf{v}_k|}/\Omega_{\alpha\beta}, \qquad (3)$$

$$T_{tk} = \frac{\mathbf{X}_t \cdot \mathbf{X}_k}{|\mathbf{X}_t|^2 + |\mathbf{X}_k|^2 - \mathbf{X}_t \cdot \mathbf{X}_k}, \qquad (4)$$

where t is the interpolation step and $k = \alpha, \beta$. Note that $\Omega_{tk} \sim 0$ indicates two structures that are close in latent space whereas $T_{tk} \sim 1$ indicates structures that are close in real space.

We generated a total of 200 structures from ten pairs of random configurations obtained from the training data set. Each interpolation path contains 20 intermediate structures. We then pass the structures through the encoder and compare the predicted mechanical properties with the MD results. The mechanical predictions of half of the 200 structures are within 15% error relative (in real units) to the MD results. Our discussion will focus on a few representative examples. Details on how configurations were randomly selected and results on all other structures can be found in the SM.

Figure 4(a) shows intermediate structures from interpolating two structures with orthogonal cuts (path 3) and orthogonal and parallel cuts (path 5 and path 8). Figure 4(b) shows the corresponding property predictions and the MD results (path 3 and path 8) in the normalized ultimate stress versus ultimate strain plot (mechanical space). As shown in Fig. 4(a)(p3), the interpolation scheme allows us to generate similar structures in regions that are close to the training domain. It can be seen the MD results are close to the predicted values.

In contrast, as shown in Figs. 4(a)(p5) and 4(a)(p8), by interpolating two configurations that have different cut types, we are able to generate designs consisting of separate parallel and orthogonal cuts as well as overlapping (mixed) cuts, whereas the training data set does not have configurations with two types of cuts. Because the SAE is interpolating two structures that are mechanically and structurally different (*far* in the mechanical space), the predicted mechanical properties are not exact but still in reasonable agreement. The mean absolute ultimate strain relative errors of the three representative structures are 8.5%, 51%, and 15% for p3, p5, and p8, respectively. The mean absolute ultimate stress relative errors of the three representative structures are 10%, 43%, and 11% for p3, p5, and p8, respectively. Several works in computer vision have also shown that ML models do not generalize well to samples that are from a slightly different distribution than the training set [31–33], which means that the ML model can capture only a subset of the underlying physics. We also observed some trends showing the prediction errors of test data points increase with increasing distances from their nearest training point in the latent space (see Figs. 9–12 in the SM). A comparison between MD and ML predictions for all structures as well as details on the increase in error with increasing latent space distance can be found in the SM.

In Figs. 4(c) and 4(d) we plot the similarity metrics for path 8. We found that designs that are different in real space are not necessarily different in the latent space [34]. For instance, in path 8, there are many distinct designs with similar mechanical properties. By comparing the visualization of the structures and their mechanical properties to the similarity metrics, we find that the angular distance performs best in capturing *both* the differences in structures and mechanical properties.

With this in mind, we recommend using the angular distance as an alternate metric to guide searching in the latent space and to generate diverse training data sets or potential
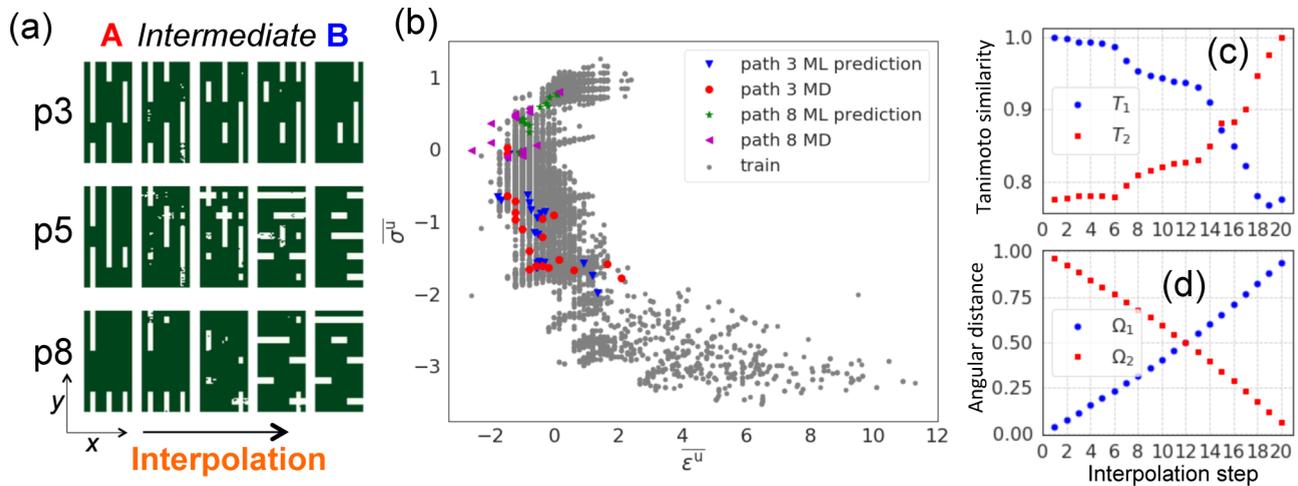
FIG. 4. (a) Three representative designs generated by interpolating from parent A (first column) to parent B (last column). Some structures with mixed cuts are obtained when the two parent structures have distinct cut types. (b) Comparison between ML predictions and MD results plotted in the normalized ultimate stress vs normalized strain plot (mechanical space). Similarity metrics for path 8 measured by (c) Tanimoto similarity and (d) angular distance. Note that the kirigami are stretched in the x direction and the edges are not shown.

designs to obtain nonredundant models. To show how we can utilize our approach to search for designs, we compare two search strategies in generating structures: (i) Select the structure with the highest strain or (ii) select the structure that is the most different, as measured by the angular distance similarity metric. Out of the 200 representative generated designs, we obtained 87 designs with mixed cuts when we used strategy (ii) whereas we only obtained four designs with mixed cuts when we used strategy (i) (see Figs. 10 and 11 in the SM). It is an open question in ML research how to best maximize search diversity, which we hope to further investigate in future studies.

Conclusions. In this Rapid Communication, we have demonstrated the ability of the supervised autoencoder (SAE) to perform both forward and inverse design of graphene kirigami. With regards to forward design, by distinguishing the difference in mechanical properties depending on the cut pattern and orientation, the SAE can overcome the traditional problem of needing to search through the entire design space library to obtain different designs. With regards to inverse design, the SAE enables the generation of structures by passing the latent variables to the decoder. Because the latent space is significantly smaller than the real space, we can perform optimization in the latent space as has previously been done to discover new drugs and chemical compounds [15]. Most importantly, we are able to classify designs that are different from the training data by measuring similarity metrics. While the mechanical property predictions of the sAE for structures that are significantly different (far from the training data) are less accurate, the SAE can still be utilized to propose alternate designs. As online databases for mechanical systems, such as the mechanical MNIST database [35], are developed, our model will be important for learning the underlying physics in a reduced-dimensional space, as well as for proposing designs. Moreover, as the local structures are tightly connected to electronic properties, this method can be extended for learning electronic properties in 2D materials, such as pseudomagnetic and electric polarization, as a function of defects or kirigami cut patterns [36–41].

Codes to generate kirigami structures and TENSORFLOW codes are freely available [42].

[1] M. A. Dias, M. P. McCarron, D. Rayneau-Kirkhope, P. Z. Hanakata, D. K. Campbell, H. S. Park, and D. P. Holmes, Soft Matter **13**, 9087 (2017).

[2] A. Rafsanjani and K. Bertoldi, Phys. Rev. Lett. **118**, 084301 (2017).

[3] Y. Yang, M. A. Dias, and D. P. Holmes, Phys. Rev. Materials **2**, 110601 (2018).

[4] M. Moshe, E. Esposito, S. Shankar, B. Bircan, I. Cohen, D. R. Nelson, and M. J. Bowick, Phys. Rev. Lett. **122**, 048001 (2019).

[5] T. C. Shyu, P. F. Damasceno, P. M. Dodd, A. Lamoureux, L. Xu, M. Shlian, M. Shtein, S. C. Glotzer, and N. A. Kotov, Nat. Mater. **14**, 785 (2015).

[6] M. K. Blees, A. W. Barnard, P. A. Rose, S. P. Roberts, K. L. McGill, P. Y. Huang, A. R. Ruyack, J. W. Kevek,

B. Kobrin, D. A. Muller *et al.*, Nature (London) **524**, 204 (2015).

[7] P. Z. Hanakata, Z. Qi, D. K. Campbell, and H. S. Park, Nanoscale **8**, 458 (2016).

[8] P. Z. Hanakata, E. D. Cubuk, D. K. Campbell, and H. S. Park, Phys. Rev. Lett. **121**, 255304 (2018).

[9] Y. Tang and J. Yin, Extreme Mech. Lett. **12**, 77 (2017).

[10] A. Rafsanjani, Y. Zhang, B. Liu, S. M. Rubinstein, and K. Bertoldi, Sci. Robotics **3**, eaar7555 (2018).

[11] A. Seko, A. Togo, H. Hayashi, K. Tsuda, L. Chaput, and I. Tanaka, Phys. Rev. Lett. **115**, 205901 (2015).

[12] G. X. Gu, C.-T. Chen, and M. J. Buehler, Extreme Mech. Lett. **18**, 19 (2018).

[13] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling, *Advances in Neural Information Processing Systems 27*, edited by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (Curran Associates, Inc., 2014), pp. 3581–3589.

[14] K. Kim, S. Kang, J. Yoo, Y. Kwon, Y. Nam, D. Lee, I. Kim, Y.-S. Choi, Y. Jung, S. Kim *et al.*, npj Comput. Mater. **4**, 67 (2018).

[15] R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, and A. Aspuru-Guzik, ACS Central Sci. **4**, 268 (2018).

[16] J. Noh, J. Kim, H. S. Stein, B. Sanchez-Lengeling, J. M. Gregoire, A. Aspuru-Guzik, and Y. Jung, Matter **1**, 1370 (2019).

[17] W. Ma, F. Cheng, Y. Xu, Q. Wen, and Y. Liu, Adv. Mater. **31**, 1901111 (2019).

[18] To normalize y, we first take the log of ϵ^u and σ^u, then subtract from each value its mean, then divide it by its standard deviation. We normalize these two quantities so that they lie within a similar range.

[19] M. Shanker, M. Y. Hu, and M. S. Hung, Omega **24**, 385 (1996).

[20] P. Mehta, M. Bukov, C.-H. Wang, A. G. Day, C. Richardson, C. K. Fisher, and D. J. Schwab, Phys. Rep. **810**, 1 (2019).

[21] M. A. Kramer, AIChE J. **37**, 233 (1991).

[22] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevResearch.2.042006 for details of simulations, machine learning, distributions of latent variables, interpolated structures, and error predictions.

[23] The maximum cut density of parallel cuts are 15 as there are not detached structures.

[24] LAMMPS, http://lammps.sandia.gov.

[25] M. Moshe, E. Esposito, S. Shankar, B. Bircan, I. Cohen, D. R. Nelson, and M. J. Bowick, Phys. Rev. E **99**, 013002 (2019).

[26] B. Zheng and G. X. Gu, Carbon **155**, 697 (2019).

[27] S. J. Wetzel, Phys. Rev. E **96**, 022140 (2017).

[28] We varied η from 0 to 1.0 with an increment of 0.2 and we found that $\eta = 1$ has the highest R^2 on the validation set. We also found that most of the latent variables are not highly correlated (absolute Pearson correlation coefficient less than 0.4).

[29] T. White, arXiv:1609.04468.

[30] D. Ha and D. Eck, arXiv:1704.03477.

[31] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, arXiv:1312.6199.

[32] S. Dodge and L. Karam, in *2017 26th International Conference on Computer Communication and Networks (ICCCN)* (IEEE, New York, 2017), pp. 1–7.

[33] N. Ford, J. Gilmer, N. Carlini, and D. Cubuk, arXiv:1901.10513.

[34] One example is that two equivalent structures by reflection symmetries (having the same mechanical properties) will be identified as two different structures by the Tanimoto similarity metric; on the other hand, the angular distance metric will measure how two designs are different structurally and mechanically.

[35] E. Lejeune, Extreme Mech. Lett. **36**, 100659 (2020).

[36] Z. Qi, D. K. Campbell, and H. S. Park, Phys. Rev. B **90**, 245437 (2014).

[37] P. Z. Hanakata, A. Carvalho, D. K. Campbell, and H. S. Park, Phys. Rev. B **94**, 035304 (2016).

[38] P. Z. Hanakata, A. S. Rodin, A. Carvalho, H. S. Park, D. K. Campbell, and A. H. Castro Neto, Phys. Rev. B **96**, 161401(R) (2017).

[39] H. Rostami, F. Guinea, M. Polini, and R. Roldán, npj 2D Mater. Appl. **2**, 15 (2018).

[40] P. Z. Hanakata, A. S. Rodin, H. S. Park, D. K. Campbell, and A. H. Castro Neto, Phys. Rev. B **97**, 235312 (2018).

[41] V. Torres, P. Silva, E. A. T. de Souza, L. A. Silva, and D. A. Bahamon, Phys. Rev. B **100**, 205411 (2019).

[42] https://github.com/phanakata/ML_for_kirigami_design.