PNAS
www.pnas.org

Supplementary Information for

Supervised learning through physical changes in a mechanical system

Menachem Stern, Chukwunonso Arinze, Leron Perez, Stephanie Palmer, Arvind Murugan

Arvind Murugan.
E-mail: amurugan@uchicago.edu

This PDF file includes:

Supplementary text
Figs. S1 to S4
Legends for Dataset S1 to S4
SI References

Other supplementary materials for this manuscript include the following:

Datasets S1 to S4

Supporting Information Text

Supplementary Appendix 1 - Folding origami sheets

Energy of folded structures. The origami sheets used in this work are based on a self-folding origami energy model developed and validated in previous studies (1–3). The effects of stiff creases are modeled by using torsional spring elements on each crease (4, 5). Here we discuss in detail how the energy of a folded structure is computed.

For thin origami sheets with free-folding creases, the primary contribution to the energy of a folded structure is due to bending of the sheet faces. Instead of modelling the faces directly, we look at the mechanical constraints inherent to the geometry of the vertices. An origami vertex is known to apply 3 constraints on the dihedral folding angles of the creases connected to it (due to embedding of the sheet in $3D$-space). The constraints can be derived by noting that the vertex must not tear open when folded. Thus, starting from any crease, alternating rotations about the dihedral and sector angles around the vertex have to result in an identity operation (2, 4, 5).

Suppose there are N creases denoted by an index i, each folded to an angle ρ_i, and N sectors with angles θ_i around the vertex. Rotations about one dihedral angle and one sector would combine to form a rotation matrix

$$R_i = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\rho_i & -\sin\rho_i \\ 0 & \sin\rho_i & \cos\rho_i \end{pmatrix} \begin{pmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{pmatrix}. \quad [1]$$

For the vertex to be closed (i.e. not torn open) in a folded structure, the combination of rotation about all crease dihedral angles and sector angles must be the identity:

$$A \equiv \prod_{i=1}^{N} R_i = I. \quad [2]$$

A folded structure with values ρ_i that do not satisfy Eq. 2 must cause the sheet faces to bend. Mathematically, this effect will manifest in finite off-diagonal values in the matrix $A \equiv \prod_{i=1}^{N} R_i$. As there are 3 independent non-diagonal elements, we say that the vertex imparts 3 mechanical constrains on the dihedral angles ρ_i around it.

At the flat state all $\rho_i = 0$, so that all constraints are trivially satisfied. We can write down an expansion for the 3 off-diagonal terms of A ($T_1 \equiv A_{12}, T_2 \equiv A_{13}, T_3 \equiv A_{23}$) in powers of the folding angles:

$$T_a(\rho_i) = C_a^i \rho_i + D_a^{ij} \rho_i \rho_j + \dots \quad [3]$$

Then, the energy of breaking these constraints is taken as the sum of squares of the residues T_a of the constraint equations $E_{\text{Vertex}} \sim \sum_a T_a(\rho_i)^2$. Summing this vertex energy over all the vertices of the sheet gives rise to the total face bending energy. The energy due to folding of a stiff crease (modeled as a torsional spring with modulus κ_i) is quadratic in the folding angle $E_{\text{Crease},i} = \frac{1}{2} k_i \rho_i^2$. The total energy of a folded sheet with stiff creases is thus computed as

$$E_{\text{sheet}}(\boldsymbol{\rho}) \equiv E_{\text{Face}} + E_{\text{Crease}} = \kappa \sum_{v \in vertices} \sum_{a=1}^{3} T_{va}(\boldsymbol{\rho}_v)^2 + \frac{1}{2} \sum_{i \in \text{creases}} k_i \rho_i^2, \quad [4]$$

With κ the face bending stiffness scale (chosen as $\kappa = 1$ in this work), and k_i the creases stiffness values. The scale of creases stiffness is denoted by $\bar{k}$. The choice of stiffness energy scale plays an important role in our learning protocol. We have previously shown how the face bending energy scales like ρ^4 (5), while the crease stiffness energy scales like $\bar{k}\rho^2$. In turn, this gives rise to a transition folding angle scale in our model $\rho_c = \sqrt{\bar{k}}$. For large folding angles $\rho \gg \rho_c$, sheet bending energy dominates, and the folding landscape is controlled solely by the sheet geometry. At small folding angles $\rho \ll \rho_c$ (close to the flat state), crease stiffness dominates, and it is possible to reshape the force-folding map. The goal of training is to reshape this map close to the flat state, such that the applied forces fold the sheet into desired folded states. Throughout this work, we choose an initial uniform crease stiffness $k_i = 0.02$. We find that trained sheets, though having heterogeneous stiffness profiles, still maintain a dominant stiffness scale at $\bar{k} \sim 0.02$. In our sheets the transition scale is thus given by $\rho_c \sim \sqrt{0.02} \sim 0.14 rad$, a reasonable angle scale close to the flat state. To make learning in sheets feasible, we conclude that a stiffness scale $\bar{k} \sim 10^{-2}$ should be chosen.

The idea that heterogeneous stiffness at creases modifies the folding response of sheets is at the heart of our learning model. This approach was experimentally studied in lattice metamaterials, where stiffness heterogeneities in the form of negative stiffness cells are used to tune the material elastic properties (6, 7). More recently, experimental studies have shown that heterogeneous stiffness can be used to avoid erroneous actuation pathways in metamaterials. Coulais et. al. have shown that a homogeneous stiffness hierarchical structure usually responds to actuation forces in disordered, undesired ways (8). However, when the hierarchical metamaterial is designed rationally such that mechanical elements have different bending stiffness (different thickness), the structure is compactified in steps to obtain the desired final state. It is similarly known that self-folding origami with homogeneous crease stiffness usually folds incorrectly in response to folding forces (4, 9). Biasing the creases to facilitate the correct folding can remove such undesired folding pathways so that the sheet folds correctly. Zhou et. al. considered designed heterogeneous thickness (and hence stiffness) of hydrogels via photolitography to control the buckling of sheets (10). This method was used to remove unwanted pathways in origami, enabling robust folding of sheets into desired states (11).

Folding protocol. Now that the energy of every folded structure ρ_i of a specific sheet is defined. We can use this energy landscape to simulate the folding of the sheet. Experimentally there are multiple different ways to fold origami sheets (12), and we have previously outlined how these methods can be simulated numerically (5).

One way that an origami sheet can be folded is by applying torques directly to the different creases. Suppose a crease i of a flat sheet is subjected to an external torque F_i^{ext}. Such a torque will induce folding in the crease, but the sheet generally resists folding due to the extra energy that might be associated with a folded structure. Assuming that the folding process is over-damped, we may write a dynamical folding equation

$$\tau_{\text{relax}} \frac{d\rho_i}{dt} = -\frac{\partial E_{\text{sheet}}(\boldsymbol{\rho})}{\partial \rho_i} + F_i^{\text{ext}}, \qquad [5]$$

where $\boldsymbol{\rho}$ is the current folded structure, and τ_{relax} a time scale of the over-damped dynamics. In this work we utilize a specific way of folding the origami sheets. Suppose a set of external torques $\boldsymbol{F}^{ext}$ is given (this could be a training or a test example as described in the main text). First, the sheet is folded very fast with a strong external torque $\boldsymbol{F}^{ext}$, until a certain folding magnitude $\rho \equiv ||\boldsymbol{\rho}||$ is reached. For fast folding we can initially disregard the sheet energy and thus get to a state

$$\boldsymbol{\rho}_{\text{fast}} = \rho \frac{\boldsymbol{F}^{ext}}{||\boldsymbol{F}^{ext}||}.$$

Then the sheet is relaxed subject to the constraint that the overall folding magnitude is fixed (i.e. finding an energy minimum on a hyper-sphere of radius ρ in ρ-space):

$$\begin{aligned} \underset{\rho_i}{\text{minimize}} \quad & E_{\text{sheet}}(\boldsymbol{\rho}) \\ \text{subject to} \quad & ||\boldsymbol{\rho}|| = \rho. \end{aligned} \qquad [6]$$

Finding a local minimum on the hyper-sphere guarantees that this folded structure would naturally occur if the sheet is folded with appropriate torques, as any neighboring configuration costs more energy, and the local minimum will attract the folding process. This algorithm is used to mimic experimental fast folding of origami sheets, followed by clamping of a crease at a specific folded dihedral angle. Here we also adjust the clamped angle such that the overall magnitude of folding ρ remains fixed and different (discrete) folded structures may be compared more easily. Such fast folding was tested extensively (5), and found to obtain the same results as numerically solving the ODE of Eq. 5.

Origami sheets and applied force patterns. In this project we use specific self-folding origami sheets. These are triangulated thin sheets, chosen to have the property of self-foldability. As discussed above, a single vertex induces 3 mechanical constrains on the angles of creases surrounding it. Thus each vertex has to connect at least 4 creases or it would be locally rigid. On top of that, for a sheet to self-fold, it needs to have one global degree of freedom, so that the number of creases needs to be one more than the number of constraints.

A simple way of generating patterns meeting these requirements is shown in Fig. S1. These are 4 specific geometries used throughout this work as the sheets to be trained. Note that we label them according to their size, given by the number creases in each sheet. The number of creases in these sheets are $13, 19, 28, 49$ and the numbers of internal vertices are $4, 6, 9, 16$. Subtracting 3 times the number of vertices from the number of creases leaves us with one global degree of freedom for each of these sheets, as required.

The number of supported folded structures for these sheets grows exponentially with the number of internal vertices, such that these sheets can fold in approximately $2^4, 2^6, 2^9, 2^{16}$ distinct ways (4, 9). In fact, any sheet with these topologies (yet different geometries) will have a similar number of distinct folded structures. The exact details of the supported folded structures is dependent on the specific geometry, but we only require the existence of many distinct folded structures for the purpose of training.

These specific sheets, used for training classifiers throughout this work, are definitely not special. We attempted training classifiers using sheets with different geometries and obtained comparable results. In analogy to learning algorithms, the details of the sheet and its supported folded structures correspond to the family of models that the training protocol selects from. For origami, we believe the available classification models are given by merger of attractors of folded structures, supported by the sheet. Since the number of available models to choose from is exponentially large, we reason that the geometry of the sheet should play little role in the success of classification. Therefore, any self-folding origami sheet could be used for training classifiers.

The choice of force patterns applied to the sheets is constrained by the problem definition as training and tests sets. Still, there is usually freedom in how these forces are applied. For example, suppose we wish to train the 13 crease sheet of Fig. S1 on $2d$ force distributions, such as the spherical caps shown in Fig. 3 in the main text. The training and test sets could thus be supplied as pairs of numbers, together with a label (blue\orange). A simple choice for training on such a data set is to pick two creases in the sheet and apply torques directly to these creases, as in Eq. (5). Here we utilize a different approach.

For an untrained sheet with homogeneous stiffness, it is known that all folded structures reside in the linear null space of the vertex constraint matrix C at the flat state (4). Thus, forces applied in a direction within this null space are more 'natural' for the sheet, and in general cost much less energy due to face bending. We compute the span of the null space for each one of

these sheets, and find that the dimension of the null space is $d_{NS} = \#_{\text{creases}} - 2\#_{\text{vertices}}$. Therefore the 13 crease sheet has a $5D$ null space, while the 49 crease sheets has $17D$ null space. Then, the training and test examples are mapped to forces in the null space as follows. For a $n - D$ data set, we choose n random orthonormal vectors in the null space. Each training\test example is mapped to a force pattern by assigning every component to one of the random orthonoraml vectors. Now these forces can be directly applied to the sheet to facilitate the training protocol.

Before training, we choose the stiffness values to be uniform. This choice is deliberate, as it allows the training protocol to access the entire set of supported folded structures. We find that initializing the stiffness elements with a substantial poorly chosen heterogeneous profile negatively affects learning. This is expected, as poor initialization can completely eliminate good folded states which could be useful for classification. The training protocol results in *learned* heterogeneous crease stiffness that facilitates the correct classification. We observe that a heterogeneous stiffness changes the geometry of the folded structures, so that they do not strictly reside in the null space of the untrained sheet. Still, for the moderate heterogeneity developed during training, the folded structures are very close to the null space, such that the described mapping is still useful and practical.

Supplementary Appendix 2 - Training origami sheets

Learning rule. As discussed in the main text, self-folding origami sheets naturally give rise to complex mapping of force patterns to folded structures, with exponentially many structures supported by the sheet. The learning rule developed in this work is meant to modify that map by changing crease stiffness coefficients, such that only a small number of folded structures are retained, corresponding to the desired classes. Here we will define precisely how the learning rule is chosen and applied to the sheet in order to develop the desired mapping.

According to the specification of the classification problem, the trainer has no a-priori knowledge of the true underlying force distributions. Instead they are supplied with a list of labeled force patterns ('cats' and 'dogs'). These training examples are used to find a reference folded structure in the following way. We fold an untrained sheet with every 'dog' example in the training set and record the folding angles of the obtained folded structures. Then, a reference 'dog' structure $\hat{\boldsymbol{\rho}}_{\text{dog}}$ is defined as the average of all of these folded structures (normalized appropriately)

$$\hat{\boldsymbol{\rho}}_{\text{dog}} \equiv \frac{\sum_{\boldsymbol{F}\in\mathcal{F}^{\text{dog}}} \boldsymbol{\rho}_U(\boldsymbol{F})}{||\sum_{\boldsymbol{F}\in\mathcal{F}^{\text{dog}}} \boldsymbol{\rho}_U(\boldsymbol{F})||}, \qquad [7]$$

with $\mathcal{F}^{\text{dog}}$ the set of 'dog' training force patterns and $\boldsymbol{\rho}_U(\boldsymbol{F})$ the folded response of the untrained sheet to force pattern $\boldsymbol{F}$. A similar reference state $\hat{\boldsymbol{\rho}}_{\text{cat}}$ is obtained for the 'cat' training examples. Crucially, once the reference structures are set for the untrained sheet, they are kept fixed throughout the training process. These reference structures are used to define the learning rule discussed in the main text. Suppose that during the training protocol, we choose a random 'dog' example $\boldsymbol{F}^{\text{dog}}$ and apply it to the sheet. The normalized resulting folded structure is written as $\boldsymbol{\rho}(\boldsymbol{F}^{\text{dog}})$. The learning rule then compares this folded structure to the reference structures defined above and the stiffness coefficients are modified as follows:

$$\begin{aligned} &\text{if } \boldsymbol{\rho}(\boldsymbol{F}^{\text{dog}}) \cdot \hat{\boldsymbol{\rho}}_{\text{dog}} > \boldsymbol{\rho}(\boldsymbol{F}^{\text{dog}}) \cdot \hat{\boldsymbol{\rho}}_{\text{cat}} : \quad && \frac{dk_i}{dt} = -\alpha\rho_i^r(\boldsymbol{F}^{\text{dog}}) \\ &\text{else} \ : && \frac{dk_i}{dt} = +\alpha\rho_i^r(\boldsymbol{F}^{\text{dog}}), \\ &k_i \geq 0, \ i \in \text{creases} \end{aligned} \qquad [8]$$

where we choose $r = 2$. In essence, the learning rule checks whether the observed folded structure is closer to the 'dog' reference than to the 'cat' reference. If it does, the stiffness of creases that fold considerably in that structure is reduced, effectively reinforcing this force-fold mapping. An opposite modification occurs if the folded structure is far away from the 'dog' reference. A similar training rule is used when 'cat' forces patterns are applied, with the understanding that we wish to compare the resulting folded structure $\boldsymbol{\rho}(\boldsymbol{F}^{\text{cat}})$ to the 'cat' reference $\hat{\boldsymbol{\rho}}_{\text{cat}}$. The intuition for this learning rule is that the softer a crease is, the more it will tend to fold. Thus if the sheet responds to a force in a desired way, making the creases that fold more softer will increase the likelihood that it will continue acting in the right way when subject to that force. Conversely, if the sheet does not respond correctly, stiffening reduces the likelihood it will respond incorrectly to that force in the future. This intuition sheds light on the question of interpretability in our model, as creases that correlate with certain features in the classified data will tend to be softer after training.

As discussed in the main text, our learning modifies stiffness according to the strain energy at each crease $\Delta k \sim \rho^r$, with $r = 2$. We have considered training sheets with other values of r in the range $1 - 5$, as seen in Figure S2. These trials gave rise to qualitatively similar results. We thus elected to use $r = 2$ for our classification problems.

Note that while our learning rule is local in the space of creases, it can still learn non-local correlations in the space of input forces. It is believed that biological systems use local learning rule (13), an idea often stated as 'Hebbian learning' (14). Though such local learning rules are in principle less powerful than arbitrary non-local rules, they can indeed facilitate learning in complex data sets (15, 16).

This learning rule can naturally be generalized to more than two classes. If c classes are to be classified, one could define c reference folded states. Then Eq. 8 could be used for learning from given training examples, with a simple modification; Crease i should be softened in proportion to the folding angle ρ_i^2 if the folded state is closest to the appropriate reference state. Otherwise, the crease should be stiffened.

Assigning labels to folded structures. To begin with, we are given labeled force patterns, and an untrained sheet with many available folded structures. It is important to note that these folded structures are equivalent and not intrinsically labeled. Thus, as part of the learning protocol we must specify how to label these folded structures, and in particular which of them to call 'dog' and 'cat' (or 'blue' and 'orange'). A simple solution would be to choose 2 of the folded structures in advance and assign the classification labels to them. Unfortunately, this turns out to be too restrictive for a couple of reasons. First, the choice may be far from ideal in the sense that these labeled folded structures are very different than the actual folded response of the sheet to the labeled force patterns. Furthermore, as the training process modifies the stiffness of different creases, the folded structures supported by the sheet change as well, either by moving around or disappearing altogether in saddle-node bifurcations (5). We thus take a different approach to labeling folded structures, as detailed below.

Suppose we have trained a sheet for some time, and it now has a particular stiffness profile on its creases k_i. To find a folded structure of this sheet to be labeled 'dog', we apply each of the 'dog' training examples once, and record the discrete resulting folded structures due to all of them $\{\boldsymbol{\rho}(\boldsymbol{F} \in \mathcal{F}^{\text{dog}})\}$. We then count the training force patterns that folded into each one of the structures in this set. The folded structure that resulted from the largest number of training force patterns is chosen to be labeled as 'dog'. In case of a tie, e.g. two or more folded structures folding as a result of the same number of force patterns, one of these structures is randomly chosen to serve as the label. Thus, the labels for 'dog' and 'cat' are decided through simple plurality rules every time we compute the classification accuracy of the sheet. Note that force patterns may also fold the sheet into structures not labeled as either 'cat' or 'dog', in which case they count as failed classification. If both 'dog' and 'cat' labels are chosen to be associated with the same folded structure, a plurality rule between the two classes decides which class is labeled with that structure (i.e. whether more 'cat' or 'dog' force patterns folded into that structure), while the other is assigned with the runner up structure of that labels' plurality vote. Finally, if the sheet is over-trained to the point where only one folded structure remains, that structure is labeled as both 'cat' and 'dog', such that classification fails completely, by definition.

Effective cost function. In this work we have defined our learning rule as a supervised physical process modifying the stiffness coefficients of an origami sheet. It is interesting to compare this kind of learning protocol to more established learning algorithms originating in computer science and statistics. An important difference is that traditional learning algorithms are usually defined as an optimization problem, where the function to be optimized (often called cost or loss function) incorporates the training data.

A simple example of a learning algorithm is linear regression, where the cost function is usually chosen as a least squares form, with differences taken between a linear model $h(x)$ and the observations y:

$$\begin{aligned} \text{Cost} &\equiv \sum_{d\in\text{data}} \left(h(x_d) - y_d\right)^2 \\ h(x) &= a_0 + a_1 x \end{aligned} \quad . \qquad [9]$$

The regression (or learning algorithm) then optimizes the cost function with respect to the model parameters $\boldsymbol{a} \equiv (a_0, a_1)$

$$\underset{\boldsymbol{a}}{\text{minimize}} \quad \text{Cost}(\{x\},\{y\};\boldsymbol{a}).$$

This optimization can be performed in any number of ways, but a practically favored method (at least for more advanced algorithms like deep learning) is mini-batch stochastic gradient descent (SGD) (17). In an extreme case, when the mini-batches are chosen to be of size 1, a single training example (x, y) is chosen at random in each step, and one computes the gradient (with respect to parameters $\boldsymbol{a}$) of the cost function defined with this example alone $\boldsymbol{G} \equiv \nabla_{\boldsymbol{a}}(h(x)-y)^2$. Now, training proceeds by modifying the parameters in proportion to the the gradient of this single example cost function

$$\boldsymbol{a} \to \boldsymbol{a} - \alpha \boldsymbol{G}, \qquad [10]$$

where α is a scalar known as the learning rate. We may compare this single example SGD with our origami training protocol. It is relatively easy to see that our training rule (Eq. 8), once a standard wait time is chosen at the folded state, has the form of SGD, making it similar in essence to other learning algorithms. To find out what effective cost function gives rise to the origami learning rule, we integrate Eq. 8 with respect to the stiffness coefficients

$$\begin{aligned} &\text{cost}_{\text{map}}(\boldsymbol{\rho}(\boldsymbol{F}^{\text{dog}})) = f \sum_{i\in\text{creases}} k_i \rho_i^2(\boldsymbol{F}^{\text{dog}}) \\ &\text{if } \boldsymbol{\rho}(\boldsymbol{F}^{\text{dog}})\cdot\hat{\boldsymbol{\rho}}_{\text{dog}} > \boldsymbol{\rho}(\boldsymbol{F}^{\text{dog}})\cdot\hat{\boldsymbol{\rho}}_{\text{cat}} : \qquad f = +1 \\ &\text{else} \ : \qquad\qquad\qquad\qquad\quad f = -1 \end{aligned} \quad . \qquad [11]$$

Similarly to the linear regression example, our origami training protocol attempts to minimize this derived cost function, one training example at a time. Inspecting this function, note that it is very similar to the energy of the torsional springs in the folded structure $E_{\text{Crease}}(\boldsymbol{\rho}) \sim \sum_i k_i \rho_i^2$. The difference is in the 'supervising factor' f that can be ± 1 whether the folded structure is accepted or not. We conclude that our origami training protocol is attempting to minimize the energy of accepted folded structures, while maximizing the energy of rejected structures. It is however important to note that the origami model does not have a fundamental cost function to optimize, but instead a local learning rule, from which a cost function emerges.

Complexity of origami classification. Self folding origami is often associated with difficult (NP-complete) computational problems. The Classic work of Bern and Hayes has shown that determining whether a sheet is rigidly foldable is NP-complete (18). More recently, it was shown that even folding a given sheet to a desired folded state is NP-complete (4). These NP results apply to a sheet with soft creases; consequently there are many ways (e.g., MV assignments) of incorrectly folding the sheet. In fact, the result can be intuitively understood by a mapping from folding origami to a satisfiability (SAT) problem of a set of equations (one for each vertex) with boolean variables representing the M or V state of each crease at that vertex. Such a SAT problem can also be visualized as a spin glass Hamiltonian with many local minima (19).

Thus, we expect that no efficient (polynomial time) algorithm can modify *all* origami sheets in a way that makes them easy to fold. Fortunately, computational complexity is a statement about the most difficult instances of a particular problem. It is certainly possible that an efficient algorithm can fold a *typical* sheet in a desired way. We have previously described such an algorithm (5), based on linear or quadratic programming, that selects the correct crease stiffness on creases to support easy folding of the sheet. It was shown that this algorithm can facilitate easy folding of many sheets, which would otherwise require great care to fold correctly. This idea, that the right crease stiffness heterogeneity can be used to make a typical sheet fold in desired ways, is used as the basis of our learning algorithm presented in this work. When crease stiffnesses are introduced, many or all of these incorrect ways of folding can be made energetically unfavorable. In the SAT or spin glass analogy, stiffnesses can be viewed as fields or biases for the variables that lift many of the minima. Thus, the stiffnesses found by our algorithm modify the relevant SAT problem until it is easily solvable. Finally, we must note that results about NP-hardness are worst case results; i.e., there exists at least one sub-class of problems that are exponential time to solve. The supervised learning framework here works with reasonable consistency but any such statistical approach that typically works cannot contradict any NP-hardness results, since it fails on some problems.

Supervised learning, and supervised classification specifically, are NP-complete problems as well (20). Given a large set of constraints (data points) and a certain family of models, we do not know of an efficient algorithm guaranteeing a set accuracy of classification. In this way, machine learning is similar to problems in physics such as spin glasses (19), and self folding origami. With regards to complexity, the learning protocol suggested in this work is similar to machine learning algorithms. Neither our learning rule nor traditional algorithms guarantee an accurate classification for a specific data set and model (in our case, specific sheet). However, it is known experimentally that an accurate solution to a classification problem can be found by using more expressive models (e.g. a deeper neural network). Similarly, we find that larger sheets with more stiff creases provide better classification results, as shown in the main text.

Example classification problem. In this section we provide further detail about the example classification problem discussed at length in the main text, and shown in Figure 3. While it is not a standard benchmark classification problem, we wish to include this extra information here for the sake of future reproduction of these results by other physically motivated learning models. The full data set, including the training forces and the progress of the training protocol, as well as MATLAB codes for training the sheet, are included as supplementary files.

As described in the main text, we use a 13 crease sheet to classify forces drawn from two classes (Figure 3a). The initial sheet has uniform stiffness on all creases ($k_i = 0.02$, in the units where bending stiffness is chosen as 1). We consider the $5d$ null-space of the sheet and classify forces in that space, Force directions F_1 and F_2, defining the distribution to be classified, are two random orthonormal directions in this null-space. We draw 20 training forces from each class, dog and cat, given according to the distribution $S^{\text{dog}} = \{\boldsymbol{F} | \boldsymbol{F} \cdot \boldsymbol{F}_{\text{dog}} \geq D, \boldsymbol{F} \cdot \boldsymbol{F}_1 > \boldsymbol{F} \cdot \boldsymbol{F}_2\}$, and similarly for S^{cat} for a threshold $D = 0.6$. The forces we pick are normalized, so that they live on the surface of a $5d$ sphere, but only 2 of these dimensions are relevant for classification. Therefore, if we sampled forces with small component in the $F_1 - F_2$ plane, they would be extremely hard to classify. For this reason we choose the cutoff $D = 0.6$, ensuring the sampled forces have a significant component in the relevant space.

Once the training forces are picked, we also sample 800 test forces for each class from the same distribution. The training and test forces are randomly ordered. We note that while the order of training examples affects learning, these effects do not change results qualitatively, as long as all training examples are shown. To train the sheet, we go through the training examples, alternating the class at every iteration. We fold the sheet with these training forces and apply the update rule of Eq. 8 given the obtained folded state. We choose the learning rate $\alpha = 10^{-4}$. After exhausting all of the training examples, we say the training has advanced by one epoch. Then, we continue training on the same training set for as long as necessary. Data for this simulation, as well as MATLAB codes for training the sheet, are available as supplements.

Supplementary Appendix 3 - Using origami sheets to define classification problems

The force distributions classified in the main text are relatively simple. Both the spherical cap and the *Iris* data distributions can be well separated by a hyper-plane, a very simple decision boundary. It is interesting to study the type of decision boundaries naturally trainable in origami sheets – and whether they can be used to classify intrinsically high dimensional data.

There are many ways to obtain high dimensional distributions. Here we choose to study distributions derived from the folding maps of origami sheets. Consider a relatively simple sheet with 2 internal vertices (Fig. S3a). It is known that such sheets support 4 discrete folded structures, and that the linearized null space in which they reside is 3-dimensional. Therefore, if we sample random force patterns within this null space, we expect to see the sheet folding into 4 distinct structures (color coded regions in Fig. S3b). The forces F_1, F_2, F_3 are assigned by randomly choosing Euler angles on the 2-sphere, and 3000 data points are sampled on the positive octant. Note that we sample normalized forces on the surface of a $2-sphere$, such that the distribution of force patterns is actually 2-dimensional.

Now, suppose we wish to classify forces to 2 classes ('blue'\'orange'). A simple way to create 2 neighboring sets of points is to take the data of Fig. S3b and merge some attractor regions to create larger groups of points. In Fig. S3c, we merge the 'blue', 'yellow', and 'purple' folded structures to create one region we define as 'blue'. This process yields two distributions that are intrinsically 2-dimensional, and not naturally separable by a hyper-plane. Larger sheets can be similarly used to create force distributions in higher dimensional space.

With this process, we have access to a new variety of 2-way classification problems, on which we can try to train origami sheets using the training protocol described in the main text. Crucially, the sheet used to classify such distributions is different than the sheet used to derive the distribution. In other words, we ask if our training protocol can induce an origami sheet to mimic the force-fold mapping of another sheet.

Suppose we want to classify the distribution seen in Fig. S4a, derived form a 2-vertex sheet as described above. We wish to train a 13 crease sheet to classify this force pattern data. The untrained sheet has 2^4 discrete folded structures that do not align with the target distribution in any representation that we tested (Fig. S4b). The problem of classification here is to train this sheet to have just 2 folded structures with the right force-fold mapping as in the target distribution.

The target distribution is mapped to applied force patterns on the 13 crease sheet by the construction describe in Supplementary Appendix 1: choosing random orthonormal vectors in the null space of the 13 crease sheet and mapping the distribution as components of these vectors. We then randomly sample 20 'blue' and 20 'orange' force patterns, marked as diamonds in Fig. S4, to serve as the training set. As we train the sheet, the classification accuracy improves dramatically and reaches a maximum of 82% (test accuracy) after 23 epochs (Fig. S4c). To qualify the classification better, we look at the classification results corresponding to the maximal accuracy at epoch 23 (Fig. S4d). We observe that the trained decision boundary resembles the desired boundary, so that the training protocol indeed produced a reasonable classification.

Note a few artifacts that still remain in the trained map: 1) there are 3 folded structures left, rather than 2 (a small third color coded region exists, labeled yellow), 2) a second orange region appeared inside the bulk blue region, emphasizing that the decision boundaries between folded structures in sheets are generally *not* hyper-planes. We conclude that origami sheets can be trained to classify distributions derived from other sheets, that are intrinsically higher dimensional than the problems discussed in the main text. Moreover, the decision boundaries are non-linear, so that in principal sheets can classify data that is not linearly separable. We leave questions of the sheet size and the complexity of decision boundaries to future studies.

Supplementary Appendix 4 - Transforming *Iris* data to applied forces on sheets

The *Iris* data set (21) classified in the main text is a classical problem for classification. In this work we are able train an origami sheet to correctly classify two species of Iris (*I. Versicolor*, *I. Virginica*) at an accuracy of 91%. Here we discuss how the *Iris* data is used to generate training and test sets of applied force patterns to be used on origami sheets.

Each *Iris* example in the data set is given as a vector with 4 features (components): sepal length, sepal width, petal length, petal width. These length measurements are all given in *cm*. In addition to these measurements, each *Iris* specimen is labeled as one of the *Iris* species in the study. To generate force pattern sets from this data, we would like the different measurements for each *Iris* specimen to be components of force vectors in the null space of the origami sheet, as described in Supplementary Appendix 1. However, the raw *Iris* data is not suited for this purpose due to two reasons. The dimensionful measurements of lengths, if directly translated to forces, would be far too great for our sheets and will cause it to fold too much and cause the sheet faces to collide. More crucially, sepal and petal lengths tend to be considerably larger than their widths, and the same goes for the variance of these variables. This will causes the width variables to be perceived as less important in the training protocol, and have a negative effect on the classification results.

Fortunately, diverse data like this is an issue regularly faced by learning algorithms, and it is generically solved by applying an invertible transformation to the data. The transformed data is then better suited for the learning algorithm in use. A typical example of such a transformation in data sets is to normalize each feature (divide by the mean of that feature) and translate it such that the mean of the transformed data is 0. This transformation is especially useful for classification algorithms like logistic regression, where the different features have different dimensional units.

In our case however, the standard transformation above is not useful, due to a particular property of origami sheets, namely their Z_2 symmetry. If forces $\boldsymbol{F}$ are applied to the sheet and it folds into a state $\boldsymbol{\rho}$, then folding the same sheet with forces $-\boldsymbol{F}$ will result in a state $-\boldsymbol{\rho}$. This is true for any self-folding origami sheet, regardless of the stiffness profile on its creases. This property cannot be changed by training the sheet. Thus, force patterns of opposite sign and different labels cannot be correctly classified. A simple way to avoid this issue is to limit the force patterns to reside in a restricted part of force space. We choose to limit the distributions such that the transformed *Iris* data will all be in the positive 4-hyperoctant.

In addition, we want the data to span as much as possible of the positive hyperoctant. This will increase the expressive of our training protocol, as more discrete folded structures would become available if the applied force patterns are more diverse. We thus need to transform the *Iris* data to be all positive, and stretch it such that all features have similar variance.

To achieve these goals we apply the following linear (invertible) transformation to the *Iris* data of the *Versicolor* and *Virginica* species. Suppose an *Iris* specimen is given as a vector $\boldsymbol{x}$ (where the components are sepal length, sepal width, petal length, petal width in this order). The vector is transformed by

$$\boldsymbol{x}^* = A\boldsymbol{x} + b$$

$$A = \begin{pmatrix} 0.264 & 0 & 0 & 0 \\ 0 & 0.580 & 0 & 0 \\ 0 & 0 & 0.303 & 0 \\ 0 & 0 & 0 & 0.836 \end{pmatrix} \quad , \quad b = -0.880. \qquad [12]$$

The transformed vector is used to define the force patterns applied to the origami sheet, as described in Supplementary Appendix 1. After training is concluded, the transformation can be inverted to relate the origami classification results with the original *Iris* data, as shown in the main text.

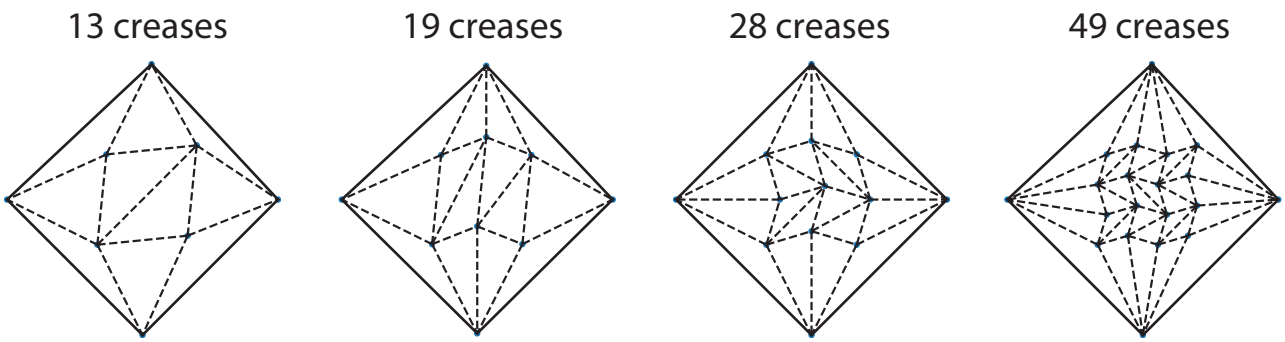

Fig. S1. Origami Sheets used for training. The size of each sheet is determined by the number creases.

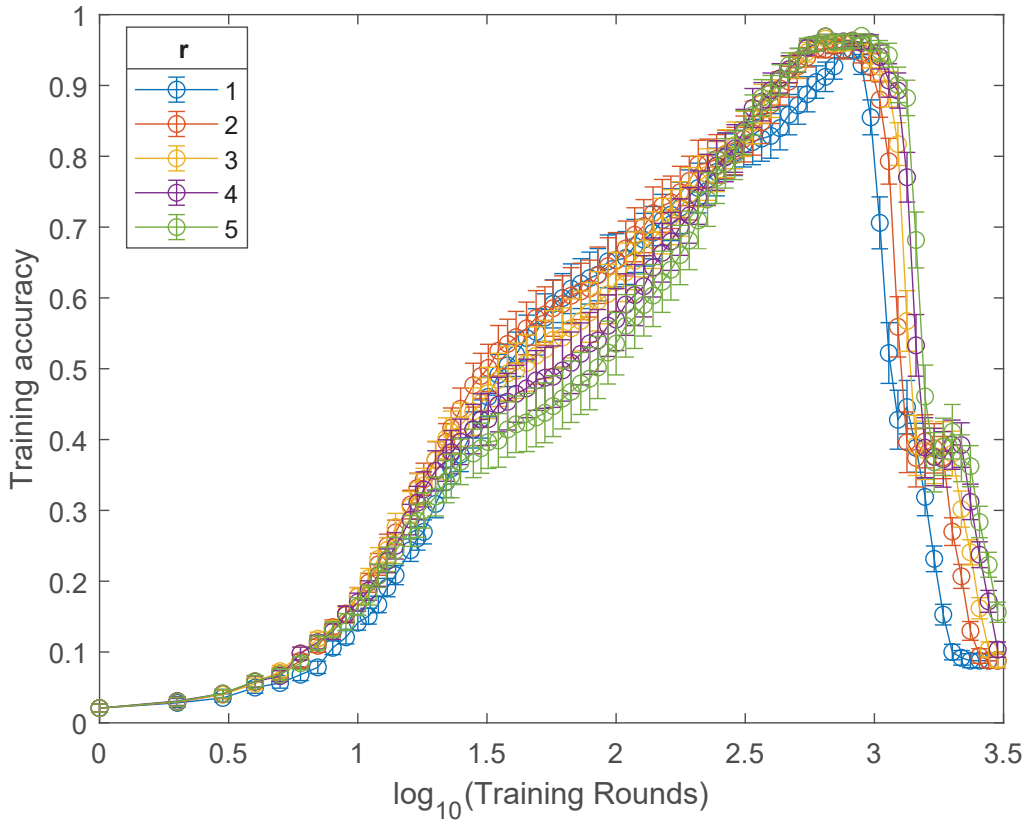

Fig. S2. Training sheets to fold as desired with different values of the power parameter r in Eq. 8. We observe small differences in the accuracy obtained using different values of r. Throughout this work we use $r = 2$, an experimentally viable choice.

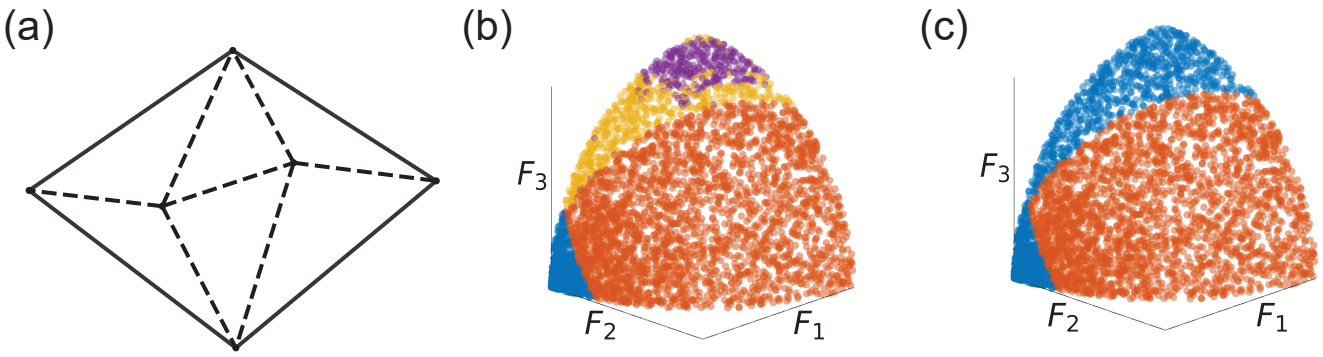

Fig. S3. Defining force distributions using the force-fold mapping of an origami sheet. a) Origami sheets with 2 internal vertices support 4 discrete folded structures. b) Sample force patterns on a 2-sphere show the force-fold mapping (4 color coded regions). c) When some attractor regions are merged (here, blue, yellow and purple are merged), we obtain an intrinsically 2-dimensional separator surface between two classes of force patterns.

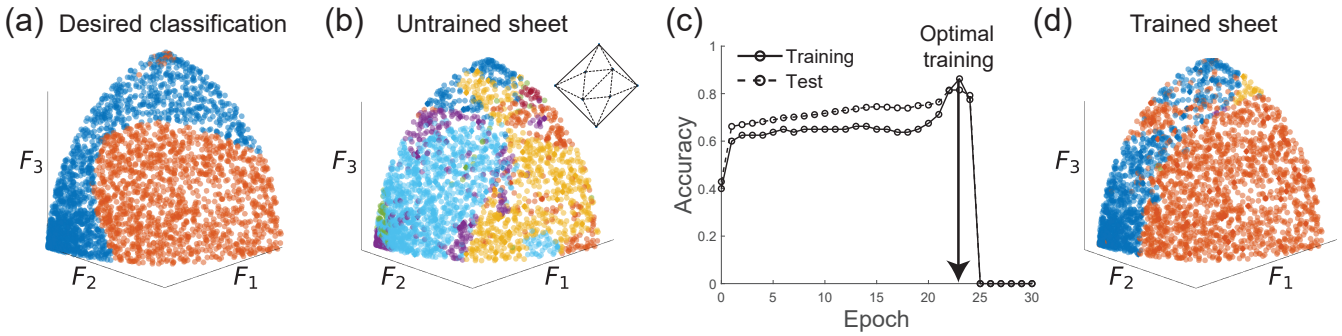

Fig. S4. Training a sheet on a force distribution derived from a different sheet. a) Target classification, a sample distribution derived from a small, 2-vertex origami sheet. b) The force-fold map of an untrained 13 crease sheet is very different from the desired mapping. c) With training, the accuracy of classification improves and peaks at 82%. d) The optimally trained sheet has a complex decision boundary that resembles (but different than) the desired boundary.

SI Dataset S1 (BlueFS.txt)
Training and test forces for the blue class (in the sheet full folding space.

SI Dataset S2 (BlueNS.txt)
Training and test forces for the blue class (in the sheet null space).

SI Dataset S3 (RedFS.txt)
Training and test forces for the Red class (in the sheet full folding space.

SI Dataset S4 (RedNS.txt)
Training and test forces for the Red class (in the sheet null space).

References

1. SM Belcastro, TC Hull, Modelling the folding of paper into three dimensions using affine transformations. *Linear Algebr. its applications* **348**, 273–282 (2002).
2. T Tachi, Geometric considerations for the design of rigid origami structures in *Proceedings of the International Association for Shell and Spatial Structures (IASS) Symposium*. Vol. 12, pp. 458–460 (2010).
3. MB Pinson, et al., Self-folding origami at any energy scale. *Nat. Commun.* **8**, 15477 (2017).
4. M Stern, MB Pinson, A Murugan, The complexity of folding self-folding origami. *Phys. Rev. X* **7**, 041070 (2017).
5. M Stern, V Jayaram, A Murugan, Shaping the topology of folding pathways in mechanical systems. *Nat. Commun.* **9**, 4303 (2018).
6. S Konarski, M Hamilton, M Haberman, Elastic nonlinearities and wave distortion in heterogeneous materials containing constrained negative stiffness inclusions in *2014 8th International Congress on Advanced Electromagnetic Materials in Microwaves and Optics*. (IEEE), pp. 130–132 (2014).
7. BM Goldsberry, MR Haberman, Negative stiffness honeycombs as tunable elastic metamaterials. *J. Appl. Phys.* **123**, 091711 (2018).
8. C Coulais, A Sabbadini, F Vink, M van Hecke, Multi-step self-guided pathways for shape-changing metamaterials. *Nature* **561**, 512–515 (2018).
9. BGg Chen, CD Santangelo, Branches of triangulated origami near the unfolded state. *Phys. Rev. X* **8**, 011034 (2018).
10. Y Zhou, CM Duque, CD Santangelo, RC Hayward, Biasing buckling direction in shape-programmable hydrogel sheets with through-thickness gradients. *Adv. Funct. Mater.* **29**, 1905273 (2019).
11. JH Kang, H Kim, CD Santangelo, RC Hayward, Enabling robust self-folding origami by pre-biasing vertex buckling direction. *Adv. Mater.* **31**, 0193006 (2019).
12. EA Peraza-Hernandez, DJ Hartl, RJ Malak Jr, DC Lagoudas, Origami-inspired active structures: a synthesis and review. *Smart Mater. Struct.* **23**, 094001 (2014).
13. BA Richards, et al., A deep learning framework for neuroscience. *Nat. neuroscience* **22**, 1761–1770 (2019).
14. DO Hebb, *The organization of behavior: A neuropsychological theory*. (Psychology Press), (2005).
15. JR Movellan, Contrastive hebbian learning in the continuous hopfield model in *Connectionist models*. (Elsevier), pp. 10–17 (1991).
16. S Bartunov, et al., Assessing the scalability of biologically-motivated deep learning algorithms and architectures in *Advances in Neural Information Processing Systems*. pp. 9368–9378 (2018).
17. S Ruder, An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747* (2016).
18. M Bern, B Hayes, The complexity of flat origami in *SODA*. (Atlanta, GA), Vol. 96, pp. 175–183 (1996).
19. M Baity-Jesi, et al., Comparing dynamics: Deep neural networks versus glassy systems. *J. Stat. Mech. Theory Exp.* **2019**, 124013 (2019).
20. AL Blum, RL Rivest, Training a 3-node neural network is np-complete. *Neural Networks* **5**, 117–127 (1992).
21. RA Fisher, The use of multiple measurements in taxonomic problems. *Annals eugenics* **7**, 179–188 (1936).