LEIA: Learned Environment for Interactive Architected Materials

Haiqian Yang[1,2], **Yuan Cao**[1], **Markus J. Buehler**[1,2]
[1]Unreasonable Labs, Mountain View, CA 94043, USA
[2]Massachusetts Institute of Technology, Cambridge, MA 02139, USA
hqyang@mit.edu, realcaoyuan@gmail.com, mbuehler@mit.edu

Abstract

Architected materials offer a vast design space, but the classical numerical methods cannot explore it interactively: resolving a single three-dimensional lattice requires hundreds of thousands of elements, and each nonlinear solve takes minutes per load step. A learned simulator could close this gap, provided it captures the physics of real materials, which exhibit nonlinear constitutive laws, carry history-dependent internal state, undergo inertial dynamics, and may possess hierarchical structures spanning multiple length scales. We present **LEIA** (**L**earned **E**nvironment for **I**nteractive **A**rchitected materials), an action-conditioned learned environment in which engineers apply boundary conditions step by step and observe the resulting deformation and stress fields in real time. LEIA handles large three-dimensional unstructured meshes and generates autoregressive responses to user-specified loading. We introduce MicroPlate, a benchmark of architected plates spanning two regimes of microstructure modeling: architected lattices that resolve microstructure explicitly through three-dimensional geometry, and a homogeneous plate where microstructural change is modeled implicitly through internal degrees of freedom. MicroPlate is used to assess LEIA alongside four baseline methods across both regimes. Finally, we demonstrate that LEIA enables efficient candidate generation and ranking for fast surrogate-guided search for de novo designs of architected materials, with stress-accurate candidate ranking validated by finite element ground truth.

1 Introduction

Hierarchical materials, structures whose properties emerge from geometry organized across multiple length scales, are among the most effective designs in nature [1, 2]. Bone, nacre, and wood achieve combinations of strength, toughness, and light weight that far exceed what their base constituents alone can provide. Inspired by these systems, engineered counterparts, commonly termed architected materials, now replicate and extend these principles across a wide range of applications [3, 4]: lattice-based implants match bone stiffness [5], crash structures absorb energy through controlled strut buckling [6], and metamaterials achieve pro-

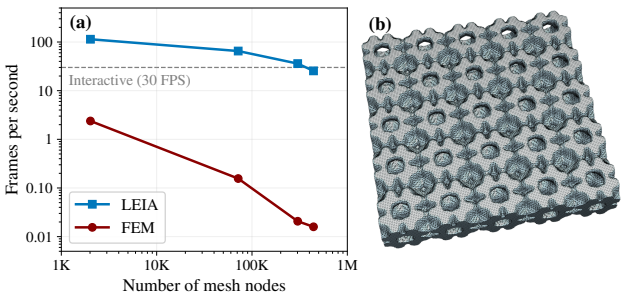

Figure 1: (a) Inference throughput of LEIA and FEM vs. mesh size. LEIA achieves high per-step inference speedup. Dashed line: interactive threshold (30 FPS). (b) A MicroPlate lattice plate (301,565 nodes).

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

grammable stiffness and tunable energy absorption through unit-cell design [7, 8]. Additive manufacturing has made these geometries fabricable, shifting the bottleneck from fabrication to design [9–11]: given the vast space of topologies, strut dimensions, and base materials, how does one efficiently explore the geometry-property landscape to find structures that meet target performance?

The conventional answer is finite element method (FEM), but for architected materials the cost is prohibitive. Resolving detailed features in a single 3D lattice structure requires hundreds of thousands of tetrahedral elements [12, 13], and each nonlinear numerical solve is computationally expensive, even for moderate meshes [14, 15]. Meanwhile, navigating the combinatorial design space demands massive simulation campaigns: recent data-driven studies have required extensive finite element evaluations to map the space of lattice topologies and geometric parameters [16–18]. Neural surrogates, neural network models that approximate complex physics simulations, can accelerate individual evaluations [19, 20], and GPU-accelerated solvers reduce per-solve wall time [21], but interactive exploration of the design space remains an open problem.

We address this problem with a learned environment in which an engineer applies boundary conditions step by step and observes the resulting deformation and stress in real time, accelerating the design loop by orders of magnitude. In robotics and game agents, action-conditioned learned simulators, termed world models, serve the same function by standing in for the ground-truth simulator during policy learning [22–24], whose formulation we adopt here. Building such an environment for architected materials requires a neural surrogate that handles large three-dimensional unstructured meshes, predicts both displacement and stress accurately under complex constitutive laws, and responds autoregressively to user-specified boundary conditions.

Neural surrogates have made substantial progress on mesh-based prediction. The community has converged on an encode-process-decode architecture rooted in Perceiver IO [25, 26]: cross-attention compresses an arbitrary mesh to fixed-size latent tokens, a transformer processes them, and cross-attention decodes back to physical space [27–30]. These models achieve high accuracy on established benchmarks and run much faster than classical numerical solvers. Yet nearly all progress has been concentrated in fluid dynamics and largely confined to two-dimensional settings. Three-dimensional nonlinear solid mechanics, the domain most relevant to architected materials design, remains largely unexplored, with existing benchmarks limited to small meshes or static loading [15, 31, 32].

Beyond mesh handling, an interactive environment for architected materials design must provide accurate stress fields. Engineers rely on stress for failure prediction and force-displacement characterization, and computing stress from displacement requires evaluating a constitutive law that maps deformation history to internal forces. For simple hyperelastic materials, stress can be recovered post-hoc by numerical differentiation of the predicted displacement field [33]. For materials with explicit microstructure resolved in the mesh, this approach becomes expensive at large mesh sizes. For materials with internal state variables that encode deformation history, displacement snapshots alone do not determine stress: identical current displacements can produce different stress fields depending on the loading path. We introduce a *stress head*, a lightweight output branch that predicts stress directly, and evaluate it across both regimes on MicroPlate (Section 4).

An interactive environment must also respond to unseen boundary conditions at each timestep and quickly render the new states of the system. We adopt the action conditioning of game world models [22, 23, 34] for loading actions: at each timestep the user specifies a loading action, and LEIA conditions its dynamics on that action [35, 36].

We present LEIA (**L**earned **E**nvironment for **I**nteractive **A**rchitected materials) [1], an interactive, action-conditioned learned environment for architected materials. LEIA combines Perceiver tokenization for mesh-size invariance, action-conditioned transformer dynamics for boundary condition responsiveness, and the stress head for accurate stress readout. We evaluate LEIA on MicroPlate, a new benchmark covering 63 architected lattices with meshes of 71,000 to 442,000 nodes (two orders of magnitude larger than existing solid mechanics benchmarks) together with a visco-hyperelastic plate that exposes path-dependent stress. Our contributions:

- LEIA, an interactive, action-conditioned learned environment for architected materials that enables real-time design exploration under nonlinear mechanics with user-controlled loading.

[1] Code is available at https://github.com/HaiqianYang-MechE/leia

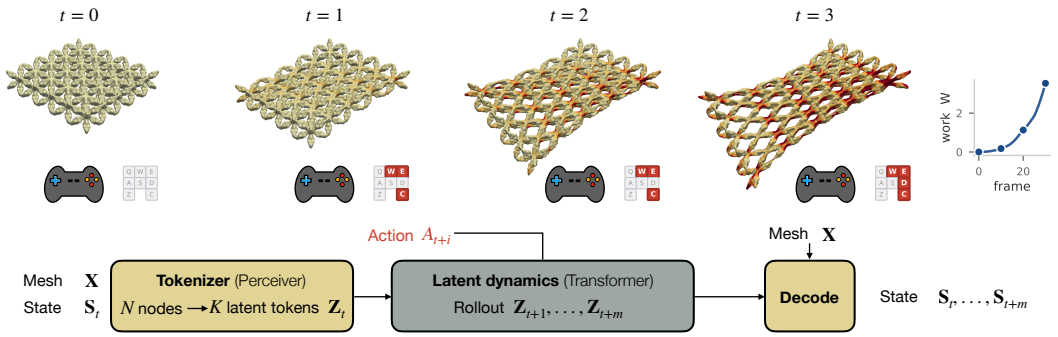

Figure 2: **LEIA architecture.** The tokenizer compresses the physical fields $\mathbf{S}_t$ on an arbitrary mesh $\mathbf{X}$ into K latent tokens $\mathbf{Z}_t$ via Perceiver cross-attention. The dynamics transformer, conditioned on the boundary condition action $\mathbf{A}_t$, predicts the next latent state autoregressively. The decoder reconstructs the physical states at mesh nodes.

- The stress head, a lightweight output branch applicable across architectures that provides accurate stress readout at constant cost regardless of constitutive complexity.
- MicroPlate, a two-regime benchmark of architected plates: 3D architected lattices where microstructure is modeled explicitly through geometry, and a visco-hyperelastic plate where microstructural change is modeled implicitly through internal degrees of freedom.
- Surrogate-guided search for de novo architected materials design, where beam search with LEIA rapidly evaluates candidates and discovers improved designs, validated by FEM.

2 Related Work

Neural operator architectures. Neural surrogates for PDEs have converged on a shared encode-process-decode template rooted in Perceiver IO [25]: cross-attention compresses an arbitrary mesh to fixed-size latent tokens, a transformer processes them, and cross-attention decodes back to the physical domain. LSM [27], UPT [28], AROMA [29], and LNO [30] each instantiate this pattern with different preprocessing and latent-dynamics choices. Physics-attention architectures have scaled to million-node geometries [37]. LEIA adopts the same template with Perceiver tokenization, FiLM-conditioned dynamics for boundary condition control, and a stress head for direct stress readout.

Autoregressive rollout and boundary condition conditioning. Most existing solid-mechanics benchmarks are quasi-static, but real materials exhibit rate-dependent behavior requiring long-horizon autoregressive rollout where error accumulates. Recent works address this via unrolled training [38], Hamiltonian inductive biases [39], and multi-wave architectures [40]. To respond to user-specified boundary conditions, standard surrogates such as MeshGraphNets [32] rely on boundary node clamping, but clamping cannot propagate new loading conditions across the mesh in a single step, particularly when the system starts from rest. Explicit conditioning on the boundary action provides this information directly at every layer.

Derivative accuracy in neural surrogates. Stress is derived from the displacement gradient, so a surrogate that reconstructs displacement accurately does not necessarily produce accurate stress. Sobolev training [41–44] addresses the gap by supervising derivatives alongside function values, and Bhattacharya et al. [45] show that history-dependent constitutive laws require explicit encoding of the deformation path. In practice, DE-DeepONet [46] and gPINNs [47] add derivative losses via automatic differentiation, but autograd incurs substantial overhead for high-dimensional outputs such as the 3D displacement Jacobian, compounding further with pushforward training. An alternative, established in molecular dynamics [48] and neural radiance fields [49], is to predict derivatives directly via a lightweight output head. LEIA's stress head follows this pattern.

Stress prediction in solid mechanics. Engineering design requires accurate stress, not just displacement: stiffness, failure, and energy absorption all depend on quantities derived from the constitutive law and the displacement gradient. A growing body of work predicts stress fields with neural networks: CNN approaches [50, 51], FNO-based methods for composites [52] and viscoplastic materials [53, 54], DeepONet branches for elastoplastic stress [55, 56], LatticeGraphNet for Neo-Hookean

lattice structures [15], GINOT for von Mises stress via transformer decoding [57], and comprehensive 3D benchmarks [58]. PeFNO [59] encodes the divergence-free constraint via a stress potential but is restricted to 2D regular grids. The MeshGraphNet-Transformer [60] handles large-scale plasticity and impact by predicting full internal variable state. At the constitutive modeling level, Constitutive Artificial Neural Networks [61, 62] encode thermodynamic constraints to learn the material law. Concurrent work evaluates stress in crash simulation via geometric pre-training [63] and graph/attention surrogates [64]. Two gaps remain. First, none of these methods combine large 3D unstructured meshes with temporal, action-conditioned rollout. State-of-the-art 3D surrogates [65] achieve impressive scale on fluids, and adaptive mesh quantization [66] has begun incorporating hyperelasticity, yet 3D solid mechanics on unstructured meshes remains unaddressed. Second, none systematically evaluate how stress accuracy degrades as constitutive complexity increases.

3 Methods

In what follows, the *surrogate model* is a learned transition predictor that maps the current state and action to the next state. The *interactive environment* is the action–state interface and low-latency rollout system built around the surrogate, which returns each frame within tens of milliseconds at mesh sizes where a single FEM step takes minutes (Figure 1a); this layer plays the role that a world model plays for robotic and game agents [22], namely an action-conditioned learned simulator, and we refer to it as the environment throughout. The *design loop* is the use of the environment for candidate exploration and selective FEM verification (Sections 4.3 and 4.4).

3.1 Problem formulation

Each loading action prescribes the displacement of the plate's gripped face; the quantities to be predicted are the resulting displacement field and stress field. We consider finite deformation solid mechanics on a body occupying reference configuration Ω_0. The displacement field $\mathbf{u}(\mathbf{X}, t)$ maps material points $\mathbf{X}$ to deformed positions $\mathbf{x} = \mathbf{X} + \mathbf{u}$, with deformation gradient $\mathbf{F} = \mathbf{I} + \nabla\mathbf{u}$ and right Cauchy-Green tensor $\mathbf{C} = \mathbf{F}^\top\mathbf{F}$. The material response is governed by a free energy $\Psi_\mathrm{R} = \Psi_\mathrm{R}^\mathrm{eq}(\bar{\mathbf{C}}) + \sum_i \Psi_\mathrm{R}^\mathrm{neq}(\bar{\mathbf{C}}, \mathbf{A}^{(i)}) + \Psi_\mathrm{R}^\mathrm{vol}(J)$, where $\bar{\mathbf{C}} = J^{-2/3}\mathbf{C}$ is the distortional Cauchy-Green tensor, $J = \det\mathbf{F}$, and $\mathbf{A}^{(i)}$ are symmetric positive-definite tensorial internal variables encoding deformation history across N_b visco-hyperelastic branches. The first Piola stress is $\mathbf{P} = 2\mathbf{F}\,\partial\Psi_\mathrm{R}/\partial\mathbf{C}$ and the Cauchy stress is $\boldsymbol{\sigma} = J^{-1}\mathbf{P}\mathbf{F}^\top$. The displacement field satisfies the balance of linear momentum,

$$\mathrm{Div}\,\mathbf{P} = \rho_0\,\ddot{\mathbf{u}}, \tag{1}$$

with reference density ρ_0. Each internal variable evolves toward thermodynamic equilibrium via $\dot{\mathbf{A}}^{(i)} = {\tau^{(i)}}^{-1}(\bar{\mathbf{C}} - \mathbf{A}^{(i)})$ with $\det\mathbf{A}^{(i)} = 1$. Together these define a coupled initial-boundary-value problem: the displacement $\mathbf{u}$ and internal variables $\{\mathbf{A}^{(i)}\}$ co-evolve, and stress at any point depends on the full deformation history through $\mathbf{A}^{(i)}$. In the elastic limit ($N_\mathrm{b} = 0$, $\rho_0\ddot{\mathbf{u}} \to \mathbf{0}$), the system reduces to quasi-static hyperelasticity, where stress depends only on the current deformation gradient.

Datasets. MicroPlate (a benchmark for architected plates with explicit and implicit microstructure modeling) covers two regimes of modeling microstructural changes, both loaded via boundary condition actions (a mixture of 4 standard plate loading axes, i.e. stretch, twist, shear in two directions) with values drawn from $\{-1, 0, +1\}$ at each timestep for tractable FEM data generation. The lattice regime resolves microstructure explicitly through 3D geometry: 63 architected lattices with meshes of 71,000 to 442,000 nodes (8,836 trajectories of 30 quasi-static steps, Neo-Hookean material), where complex stress concentrations at strut junctions distinguish these plates from the homogeneous plate. Every lattice in MicroPlate is a directly 3D-printable geometry, and its graph parameterization (nodes, beams, radii) mirrors how such lattices are specified in additive-manufacturing practice; the mesh sizes are representative of resolved-lattice finite element studies in manufacturing settings [12, 13] and two orders of magnitude larger than existing solid-mechanics benchmarks (Table 3). The viscoelastic regime models microstructural change implicitly through internal degrees of freedom: a homogeneous plate (363 nodes, 1,000 trajectories of 100 action steps) under finite viscoelasticity with inertia (Arruda-Boyce equilibrium + 3 non-equilibrium branches). Each visco-hyperelastic trajectory applies 50 random loading steps then the exact reverse 50 unloading steps. The protocol is hysteretic by design: because the internal variables lag the applied deformation, the plate passes back through

the same boundary conditions with a different internal state, so the response traces a hysteresis loop rather than retracing the loading path, and predicting stress on the return branch requires a model that carries deformation history. The two regimes vary along complementary axes: geometry varies in the lattice regime with the constitutive law held fixed; constitutive complexity varies in the viscoelastic regime with geometry held fixed. Full constitutive laws, material parameters, and FEM solver details are in Appendix A.

3.2 Model architecture

LEIA combines three design choices that address the requirements identified above (Figure 2). First, Perceiver cross-attention [25] compresses fields on meshes of arbitrary size into a fixed-length latent representation, decoupling downstream model cost from mesh resolution. Second, action-conditioning [36] injects the user's boundary condition action at every transformer layer, ensuring the dynamics model responds to new loading at each step. Third, a stress head, a single linear projection from the shared latent to six Cauchy stress components, provides stress readout at constant cost regardless of constitutive law complexity. Because stress and displacement share the same latent representation, we hypothesize that stress supervision forces the encoder to capture constitutive state that displacement alone cannot encode; Section 4.2 presents evidence consistent with this.

LEIA is trained in two stages: a *tokenizer* is first trained to compress displacement and stress fields defined on arbitrary meshes into a fixed-size latent representation, followed by a second stage which trains a *latent dynamics model* that predicts the next latent state conditioned on the current state and boundary condition action.

Tokenizer. The tokenizer follows the Perceiver IO encode-decode template [25], with encoder $\mathcal{E}_\phi$ and decoder $\mathcal{D}_\psi$:

$$\mathbf{z} = \mathcal{E}_\phi(\mathbf{u}, \mathbf{X}), \qquad (\hat{\mathbf{u}}_j, \hat{\boldsymbol{\sigma}}_{\mathrm{sym},j}) = \mathcal{D}_\psi(\mathbf{z}, \mathbf{X}_j). \tag{2}$$

A learned set of K latent queries cross-attends into the N mesh node features to produce a fixed-size latent representation $\mathbf{z} \in \mathbb{R}^{K\times H}$, refined by L_{enc} self-attention layers. Queries at arbitrary spatial positions $\{\mathbf{X}_j\}$ decode $\mathbf{z}$ back through cross-attention into the latents and two parallel linear heads producing displacement $\hat{\mathbf{u}}_j \in \mathbb{R}^3$ and Cauchy stress $\hat{\boldsymbol{\sigma}}_{\mathrm{sym},j} \in \mathbb{R}^6$. Node features project per-node displacement to dimension H and add a Fourier positional encoding of the reference coordinate. The input displacement can be a single frame $\mathbf{u}^{(t)} \in \mathbb{R}^3$ or a temporal window of F consecutive frames in $\mathbb{R}^{3F}$; multi-frame input provides velocity and acceleration context, beneficial for viscoelastic systems where internal state depends on deformation history. When the full mesh fits in GPU memory during training (e.g., the visco-hyperelastic plate), L_{dec} decoder self-attention layers attend across query positions to improve reconstruction accuracy; for large meshes requiring subsampled training (the lattice regime), we remove these layers to preserve resolution invariance (Appendix C).

Latent dynamics model. A transformer [67] predicts the next latent state from the current state and the boundary condition action:

$$\mathbf{z}_{t+1} = f_\theta(\mathbf{z}_t,\ \mathbf{a}_t). \tag{3}$$

f_θ is a stack of L_{dyn} DiT-style adaLN blocks [35, 36]: each block applies action-conditioned scale and shift to the LayerNorm output before self-attention and again before the pointwise feedforward, with residual connections in the standard pre-norm transformer pattern. The four modulation parameters per block come from a per-block linear projection of a shared action embedding. The action $\mathbf{a}_t \in \mathbb{R}^4$ is the continuous-valued cumulative boundary condition at frame t; it enters at every block, ensuring the dynamics respond to boundary-condition changes immediately.

At inference the model operates autoregressively: starting from $\mathbf{z}_0$ (the encoding of the initial displacement), each predicted latent state is fed back as input, with the user supplying a new action $\mathbf{a}_t$ at each step. The decoder reconstructs displacement and stress from any intermediate $\mathbf{z}_t$.

Training procedure and implementation details. The Perceiver tokenizer is trained with a combined loss to reconstruct both the displacement field and the stress tensor field; the Transformer dynamics model is trained to predict the next latent state conditioned on the action. The training procedure, architecture hyperparameters, optimizer settings, and training schedules for LEIA and all baselines are provided in Appendix C. All models are trained on 8×H100 80 GB GPUs.

4 Results

We organize our experiments around four findings. First, on MicroPlate's lattice regime (meshes up to 442,000 nodes), direct Cauchy stress supervision outperforms all indirect alternatives, and combining it with pushforward dynamics yields the best displacement and stress accuracy (Section 4.1). Second, on MicroPlate's viscoelastic regime where stress depends on deformation history through internal degrees of freedom, the stress head resolves a universal failure across four baseline architectures (Section 4.2). Third, surrogate-guided design search evaluates 553 candidates in 30 minutes (per-sample evaluation $\sim$100–300$\times$ faster than FEM), with the improvement of the optimization goal validated by FEM (Section 4.3). Fourth, a learned confidence head predicts surrogate reliability on out-of-distribution topologies without FEM (Section 4.4).

4.1 Accurate stress rollout on large architected meshes

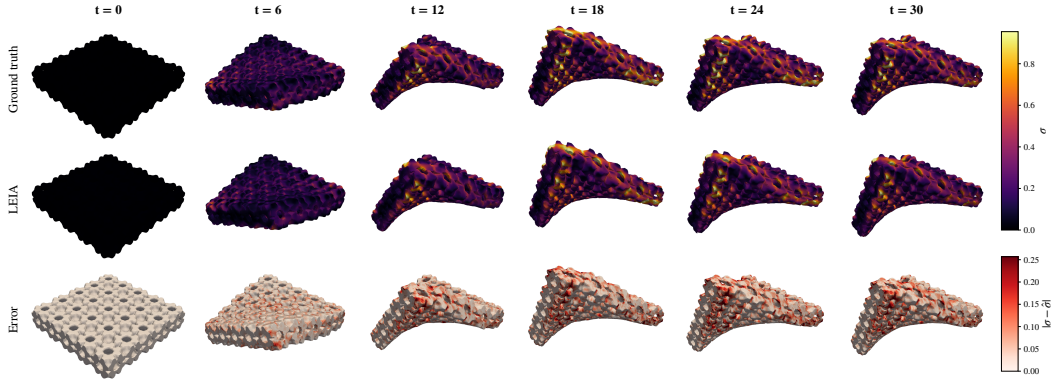

Figure 3: **Autoregressive rollout on a MicroPlate lattice (301,565 nodes, 1,382,904 tetrahedra).** Columns show selected timesteps. Top: ground-truth von Mises stress. Middle: LEIA prediction. Bottom: absolute von Mises stress error.

We evaluate LEIA on MicroPlate's lattice regime: 63 architected lattices with meshes up to 442,000 nodes. We train on 55 lattices and hold out 8 for evaluation on unseen geometries. Fig. 3 shows a representative rollout on an unseen trajectory of a training topology. Fig. 10 compares predicted and ground-truth work versus time on ten shapes.

We evaluate the full pipeline in two modes: *teacher forcing* (ground-truth state at each step, predict one step ahead) and *autoregressive rollout* (initial state only, roll out the full trajectory with user-specified boundary conditions). We report two primary metrics: (1) median per-frame displacement relative L^2 error, $\|\hat{\mathbf{u}} - \mathbf{u}\|/\|\mathbf{u}\|$, and (2) median per-frame Pearson correlation between predicted and ground-truth fields of von Mises stress, the scalar equivalent stress used in yield and failure criteria. Both are computed per trajectory and summarized by the median across test trajectories. For models without a stress head, stress is computed post-hoc from the predicted displacement via tet shape function differentiation and the constitutive law; for models with the stress head, von Mises stress is computed directly from the predicted $\hat{\boldsymbol{\sigma}}_{\mathrm{sym}}$. Full details of the stress calculation procedure are in Appendix A.

We compare five strategies for predicting stress, progressing from no gradient supervision to direct Cauchy stress supervision. (1) No gradient supervision: the tokenizer is trained on displacement reconstruction only; stress is evaluated post-hoc via tet shape function differentiation of the predicted displacement $\hat{\mathbf{u}}$. (2) Tet Sobolev: the displacement gradient is computed via tet shape function differentiation, applied to a subset of tets per training step (a deterministic furthest-point-sampled core plus a random tail, volume-weighted MSE against the per-tet ground-truth gradient). (3) Autograd Sobolev: the displacement Jacobian $\nabla\hat{\mathbf{u}}$ is supervised via automatic differentiation through the decoder, evaluated at the FPS-subsampled query points; this adds roughly $12\times$ training time per step. (4) Gradient head: a learned linear projection from decoder hidden states to $\nabla\mathbf{u}$ components, supervised with precomputed ground-truth gradients, avoiding the autograd cost (+13% overhead). (5) Stress head: a linear projection to six Cauchy stress components, supervised directly with FEM Cauchy stress, with only +6% overhead.

Table 1: Ablation on stress supervision strategy for the LEIA tokenizer on MicroPlate, progressing from no gradient supervision to direct stress supervision. The Overhead column reports additional tokenizer training time per optimizer step at fixed effective batch (128 across 8 H100s) versus the recon-only baseline. All metrics are medians on autoregressive rollout.

	In-distribution		Held-out topology		
	Disp. L^2 (%)↓	VM corr.↑	Disp. L^2 (%)↓	VM corr.↑	Overhead
No gradient supervision (tet eval)	10.83	0.244	14.60	0.210	baseline
Tet Sobolev	7.08	0.410	7.89	0.388	+53%
Autograd Sobolev	9.31	0.327	12.55	0.252	+1160%
Gradient head	17.80	0.854	21.93	0.667	+13%
Stress head	13.21	0.870	17.92	0.654	+6%
Stress head + pushforward	4.68	0.942	5.39	0.662	+6%

Without any gradient supervision, post-hoc tet-shape-function differentiation of the predicted displacement yields VM (von Mises stress) correlation of 0.24 in-distribution and 0.21 on held-out shapes. Supervising the gradient via tet shape functions provides modest gains (VM 0.41 in-dist, 0.39 held-out); autograd through the decoder gives smaller gains (VM 0.33 in-dist, 0.25 held-out), at $+1160\%$ per-step cost. The two learned-head approaches reach a substantially different range: the gradient head attains VM 0.85 in-dist and 0.67 held-out at $+13\%$ overhead, and the stress head attains 0.87 and 0.65 at $+6\%$. Adding pushforward rollout and velocity-input dynamics on top of the stress head yields the lowest displacement error in both splits (4.68% in-dist, 5.39% held-out) and raises in-distribution VM correlation to 0.94, with comparable held-out VM (0.66).

4.2 Accurate stress rollout under homogenized microstructural change

We now test the implicit-microstructure regime, a homogenized representation in which microstructural change is encoded as internal degrees of freedom on a continuum plate. On MicroPlate's visco-hyperelastic plate (3 visco-hyperelastic branches, 100-step load-then-reverse protocol; Appendix A), stress depends on the current deformation and on 18 internal variables carried by the three branches. A model can therefore predict stress in two ways: recover it from the predicted internal variables through the constitutive law, or predict the six independent Cauchy stress components directly with the stress head. Table 2 reports both strategies for the four baselines. With indirect prediction, von Mises correlation stays at or below 0.26 for every baseline, across message-passing (MGN), transformer (UPT, Transolver), and set-encoder (PointNet) architectures. With the direct stress head, all four reach von Mises correlation above 0.94, and displacement accuracy also improves. Two observations follow. First, the high stress accuracy of every baseline is obtained with the head introduced here, attached by replacing each model's output layer; Table 2 ablates the stress-readout strategy and does not indicate that any backbone resolves history-dependent stress on its own. Second, direct stress supervision is inexpensive but not the obvious choice: supervising displacement gradients through automatic differentiation (Autograd Sobolev in Table 1) increases training time per step by 1160% and reaches a von Mises correlation of 0.33, whereas the head adds 6% and reaches 0.87. Under 100-step autoregressive rollout, stress stays correlated above 0.88 for UPT and the per-frame models (PointNet, Transolver), while MGN's stress correlation drops to 0.225 alongside its displacement collapse. PointNet and Transolver are per-frame predictors with no temporal state; their rollout re-applies the model at each step using ground-truth cumulative boundary conditions, so it matches teacher forcing by construction. Because every trajectory applies 50 loading steps followed by the reverse unloading

Table 2: **Stress-prediction strategies on MicroPlate's load-reverse visco-hyperelastic plate.** 3 visco-hyperelastic branches, 100-step load-then-reverse protocol. Both the teacher-forcing and AR (autoregressive) columns report without / with the stress head. The table ablates the stress-readout strategy; the stress head introduced in this work is attached to every baseline for comparison. Medians across 100 test trajectories. The Space column indicates whether the model operates directly on mesh nodes (phys.) or on a compressed latent representation (latent).

		Teacher forcing		AR rollout (100 steps)	
Model	Space	Disp. (%)↓	VM↑	Disp. (%)↓	VM↑
MGN	phys.	3.88 / 3.52	−.009 / .987	241 / 9.94	.012 / .225
UPT	latent	13.0 / 6.09	.257 / .951	80.8 / 31.8	.197 / .882
PointNet	phys.	21.8 / 3.25	.091 / .943	22.8 / 2.87	.092 / .938
Transolver	phys.	33.0 / 2.84	.108 / .941	40.3 / 2.42	.109 / .947
LEIA	latent	**2.84**	.970	5.99	**.986**

path, the AR columns also assess stress prediction on the return branch of the hysteresis loop, where the current displacement alone does not determine the stress. Among the three temporal models, LEIA degrades least: AR displacement rises from 2.84% to 5.99%, and AR von Mises correlation holds at 0.986, the highest value in the AR column.

4.3 Surrogate-guided design search

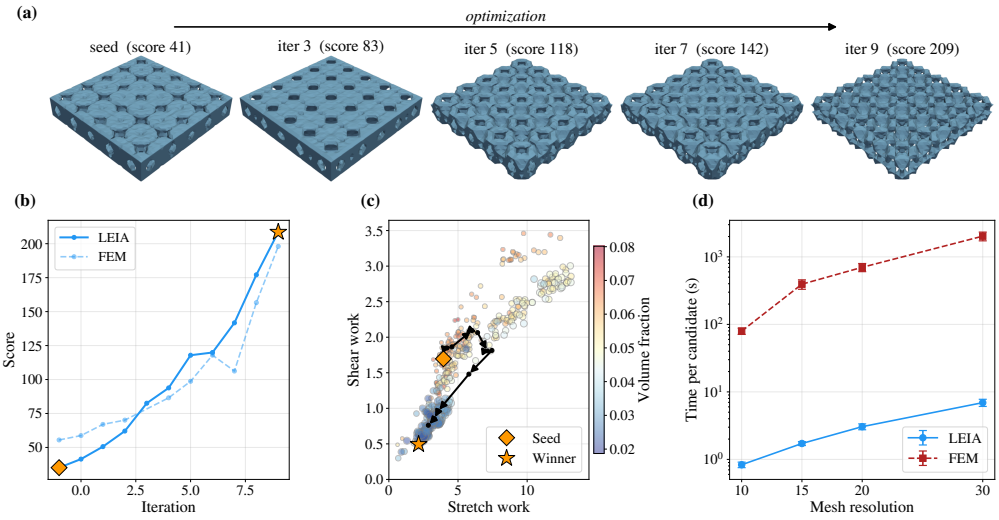

Figure 4: **Surrogate-guided beam search over MicroPlate. (a)** Topology evolution: best design at selected iterations, showing progressive material removal as the search optimizes for high stretch resistance, low shear resistance, and low volume fraction. **(b)** Convergence of the design metric. Solid: LEIA. Dashed: FEM validation at each iteration, confirming that the surrogate ranking is reliable. **(c)** Search landscape: all evaluated candidates in stretch work vs. shear work space, colored by volume fraction. Arrows trace the mean search trajectory across iterations. Diamond: seed; star: winner. **(d)** Per-candidate evaluation cost. LEIA (blue) and FEM (red) measured at four mesh resolutions. Error bars: standard deviation across the 11 best-per-iteration candidates at each mesh resolution.

The practical value of a fast, stress-accurate surrogate extends beyond prediction to *design*: an engineer can evaluate thousands of candidate topologies with LEIA in the time FEM would evaluate a handful, then spend the FEM budget only on the most promising candidates.

To demonstrate the practical value of stress-accurate surrogates, we use LEIA to evaluate designs in a beam search over architected-lattice mutations. The design metric is $s = W_{\text{stretch}}/(W_{\text{shear}} \cdot v_f + \varepsilon)$ with $\varepsilon = 10^{-8}$ for numerical stability, where W_{stretch} and W_{shear} are the work (area under the force-displacement curve) for stretch and shear loading, and v_f is the normalized volume fraction. This metric favors lightweight structures that resist axial loads while remaining compliant in transverse directions. Each architected lattice's unit cell is defined by a graph of nodes and beams with associated radii. We define six mutation operators that modify the graph topology and geometry (Table 5 in the Appendix). Each mutation is a random composition of 1–3 operators; invalid mutations are discarded and resampled (Appendix D). At each iteration, the top-B candidates are each expanded by M mutations; all new designs are meshed, evaluated with LEIA to obtain predicted force-displacement curves, and the top-B are retained. In total we evaluate 553 candidate designs across 10 iterations (hyperparameters in Appendix D).

The search evaluates 553 candidate designs in 30 minutes of total wall-clock time. Fig. 4(d) shows the per-candidate LEIA/FEM speedup remains in the $\sim$100–300$\times$ range across mesh resolutions $r \in \{10, 15, 20, 30\}$. Starting from a single seed topology (`mg_046`) with volume fraction 6.6%, the search discovers a design with volume fraction 2.1% and 10 struts (down from 15) that achieves a 3.6$\times$ improvement in the design metric as validated by FEM ground truth. FEM validation at every iteration confirms that the surrogate's ranking tracks the ground truth throughout the search (Fig. 4).

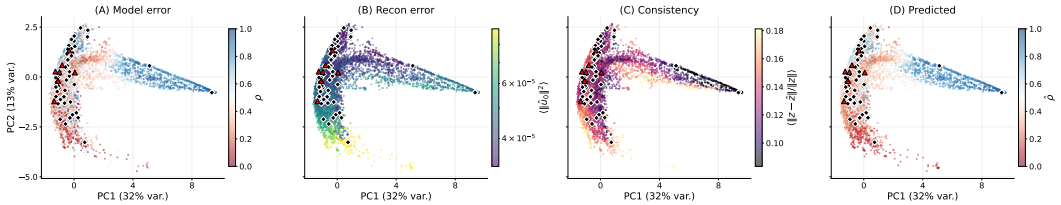

Figure 5: **OOD-detection landscape.** PCA of the tokenizer's mean-pooled latent fit on the training plates and applied to mutated candidates. Black diamonds: training plates. Red triangles: held-out plates. Light grey dots in (A): candidates whose FEM diverged. **(A)** FEM ground-truth model error: ρ is the Pearson correlation between LEIA and FEM-derived von Mises stress fields. **(B)** Tokenizer round-trip error $\langle \|\hat{\mathbf{u}}_0\|^2 \rangle$ with $\hat{\mathbf{u}}_0 = \mathrm{decode}(\mathrm{encode}(\mathbf{0}))$. **(C)** Encoder-cycle inconsistency $\langle \|\mathbf{z} - \hat{\mathbf{z}}\| / \|\mathbf{z}\| \rangle$ with $\hat{\mathbf{z}}_t = \mathrm{encode}(\mathrm{decode}(\mathbf{z}_t))$. **(D)** Predicted $\hat{\rho}$ from a learned confidence head.

4.4 Out-of-distribution design generation and surrogate confidence

Surrogate-guided exploration could produce shapes far from the training distribution, where LEIA's predictions may be unreliable. Detecting when the surrogate enters this regime, without running FEM (which defeats the purpose of the surrogate), is essential for a practical design loop. To test this, we generate a diverse set of 7,981 mutated MicroPlate candidates using random walks and novelty search that deliberately encourage out-of-distribution exploration (details in Appendix E). Fig. 5 projects these candidates onto the tokenizer's mean-pooled latent PCA fit on the 55 training plates.

To predict surrogate reliability without running FEM, we evaluate two FEM-free signals: the tokenizer's frame-0 round-trip reconstruction error and an encoder-cycle inconsistency that measures whether dynamics-predicted latents survive a decode-then-re-encode round trip. We also train a learned confidence head that combines these signals with the tokenizer's mean-pooled latent and 15-dimensional graph statistics to predict LEIA's per-plate von Mises correlation ρ. Signal definitions, head architecture, and training protocol are in Appendix E.

The tokenizer's frame-0 reconstruction error (Fig. 5(B)) flags the lower-left region of the latent cloud where geometries are far from the training distribution, but is otherwise uncorrelated with FEM-derived ρ globally. The encoder-cycle inconsistency over a 10-step uniaxial-stretch rollout (Fig. 5(C)) identifies a different failure mode in the upper region, where rolled-out latents fail to round-trip cleanly through the encoder. Both signals miss the central stripe of the latent cloud where FEM-derived ρ is low despite low reconstruction error and low inconsistency; the dynamics model produces configurations that are wrong but still within the distribution recognizable by the tokenizer, so round-trip checks cannot catch the error.

To close this gap, we train a small learned head on the 6,919 FEM-validated candidates. As shown in Fig. 5(D), the learned head captures the spatial pattern of FEM-derived ρ across the latent cloud, including the central stripe that the free signals miss. Free signals filter obvious geometric outliers cheaply; the learned head is needed for cases that are in-distribution but inaccurate. Together they enable a surrogate-assisted design loop: generate many candidates with mutation-based search, screen with the learned head to rank candidates and flag uncertain ones, and reserve FEM for validating the most promising designs. The confidence signal could also guide active data collection for out-of-distribution topologies, enabling targeted model fine-tuning in future work.

5 Discussion

An interactive environment for engineering design. LEIA provides a real-time interactive loop for architected material design: the user specifies boundary conditions, observes the resulting deformation and stress in real time, and iterates. This enables workflows that are impractical with finite element simulation alone: rapid screening of candidate topologies for target properties, interactive exploration of loading paths, and surrogate-guided search over large design spaces. LEIA can also decode at different resolutions, which could enable fast coarse rendering for exploration and zooming in to finer detail for inspection, a practical feature for interactive engineering workflows. A current limitation is geometric scope: LEIA is evaluated only on cubically symmetric lattice plates with a fixed $5 \times 5 \times 1$ tiling under a 4-degree-of-freedom action space; transfer to arbitrary three-dimensional geometries,

non-lattice microstructures, or larger action spaces is untested. Extending LEIA to these regimes is a natural direction for future work.

Surrogate-assisted design and active learning. Section 4.3 demonstrates surrogate-guided beam search as one instantiation of LEIA's role as a design tool. Several extensions are natural. *Active learning*: the learned confidence head of Section 4.4 can provide a cheap signal for identifying topologies where the surrogate is unreliable; such topologies would be candidates for new FEM data collection, enabling targeted dataset growth that concentrates compute where it is most needed. A limitation is that the current system performs no active learning at inference time; an interactive session could queue on-the-fly FEM validation when the user navigates into regions of boundary-condition space where LEIA is least confident, closing the surrogate-improvement loop during use. The beam search already demonstrates the core loop (surrogate screen $\rightarrow$ FEM validate), and active learning would extend it further. The confidence head is trained on a labeled region of the design space and inherits its coverage, so a loop that ventures well outside that region would still benefit from a signal that requires no labels where it is applied. Variance-based estimates [68, 69] are the natural candidates, at the cost of repeated inference per prediction; combining a label-free estimate outside the labeled region with the FEM-supervised head inside it is a natural direction for future work. *Learned policies*: For future work, it would also be interesting to test neural policies trained against LEIA as a differentiable environment, selecting actions to match engineering objectives, replacing the beam-search controller of Section 4.3 with a fully learned closed loop.

6 Conclusion

Architected materials offer vast design potential, but realizing it requires navigating a combinatorial space of geometries and loading conditions that finite element simulation alone cannot explore at scale. LEIA provides a real-time interactive environment for this exploration: engineers specify boundary conditions, observe deformation and stress, and iterate across candidate topologies at a pace that enables surrogate-guided design search. A learned confidence head predicts where the surrogate is reliable, enabling targeted allocation of FEM compute to the most promising or uncertain candidates. We release LEIA together with MicroPlate and baseline implementations to serve as a foundation for further work. Looking ahead, augmenting LEIA with additional physics, such as fluid-structure coupling, thermal transport, or electrochemical degradation, would broaden its applicability to multi-functional demands placed on hierarchical materials in practice.

Acknowledgments and Disclosure of Funding

This work was funded by Unreasonable Labs. H.Y., Y.C., and M.J.B. are employees of Unreasonable Labs, a company that may have a commercial interest in the methods described in this paper. The authors declare no other competing interests.

References

[1] M. A. Meyers, Po-Yu Chen, A. Lin, and Yasuaki Seki. Biological materials: Structure and mechanical properties. *Progress in Materials Science*, 53:1–206, 2008. doi: 10.1016/j.pmatsci.2007.05.002.

[2] U. Wegst, Hao Bai, E. Saiz, A. Tomsia, and Robert O. Ritchie. Bioinspired structural materials. *Nature Materials*, 14:23–36, 2015. doi: 10.1038/nmat4089.

[3] J. U. Surjadi and C. M. Portela. Enabling three-dimensional architected materials across length scales and timescales. *Nature Materials*, 24:493–505, 2025. doi: 10.1038/s41563-025-02119-8.

[4] X. Xia, C. Spadaccini, and J. Greer. Responsive materials architected in space and time. *Nature Reviews Materials*, 7:683–701, 2022. doi: 10.1038/s41578-022-00450-z.

[5] Xiaojian Wang, Shanqing Xu, Shiwei Zhou, W. Xu, M. Leary, Peter Choong, M. Qian, M. Brandt, and Y. Xie. Topological design and additive manufacturing of porous metals for bone scaffolds and orthopaedic implants: A review. *Biomaterials*, 83:127–141, 2016. doi: 10.1016/j.biomaterials.2016.01.012.

[6] T. Tancogne-Dejean, A. Spierings, and D. Mohr. Additively-manufactured metallic micro-lattice materials for high specific energy absorption under static and dynamic loading. *Acta Materialia*, 116:14–28, 2016. doi: 10.1016/j.actamat.2016.05.054.

[7] Pengcheng Jiao, Jochen Mueller, J. Raney, X. Zheng, and Amir H. Alavi. Mechanical metamaterials and beyond. *Nature Communications*, 14:6004, 2023. doi: 10.1038/s41467-023-41679-8.

[8] A. Shaikeea, Huachen Cui, M. O'Masta, X. Zheng, and V. Deshpande. The toughness of mechanical metamaterials. *Nature Materials*, 21:297–304, 2022. doi: 10.1038/s41563-021-01182-1.

[9] Xiaoyang Zheng, Xubo Zhang, Ta-Te Chen, and I. Watanabe. Deep learning in mechanical metamaterials: From prediction and generation to inverse design. *Advanced Materials*, 35: e2302530, 2023. doi: 10.1002/adma.202302530.

[10] Jan-Hendrik Bastek, WaiChing Sun, and Dennis M. Kochmann. Physics-informed diffusion models. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2403.14404.

[11] Li Zheng, Siddhant Kumar, and Dennis M. Kochmann. Algebraic language models for inverse design of metamaterials via diffusion transformers. *Nature Machine Intelligence*, 8(4):628–640, 2026. URL https://doi.org/10.1038/s42256-026-01218-8.

[12] T. Hirschler, R. Bouclier, P. Antolin, and A. Buffa. Reduced order modeling based inexact FETI-DP solver for lattice structures. *International Journal for Numerical Methods in Engineering*, 125, 2024. doi: 10.1002/nme.7419.

[13] Clément Guillet, Thibaut Hirschler, Pierre Jolivet, Pablo Antolin, and Robin Bouclier. Efficient fine-scale simulation of nonlinear hyperelastic lattice structures. *arXiv preprint arXiv:2603.10741*, 2026. URL https://arxiv.org/abs/2603.10741.

[14] Aldair E. Gongora, Caleb Friedman, D. Newton, T. Yee, Zachary Doorenbos, Brian Giera, E. Duoss, T. Yong-Jin Han, Kyle T. Sullivan, and Jennifer N. Rodriguez. Accelerating the design of lattice structures using machine learning. *Scientific Reports*, 14, 2024. doi: 10.1038/s41598-024-63204-7.

[15] Ayush Jain, Ehsan Haghighat, and Sai Nelaturi. LatticeGraphNet: a two-scale graph neural operator for simulating lattice structures. *Engineering with Computers*, 41(2):829–844, 2024. doi: 10.1007/s00366-024-02034-7.

[16] Jan-Hendrik Bastek, Siddhant Kumar, Bastian Telgen, R. Glaesener, and D. Kochmann. Inverting the structure–property map of truss metamaterials by deep learning. *Proceedings of the National Academy of Sciences*, 119, 2022. doi: 10.1073/pnas.2111505119.

[17] Jan-Hendrik Bastek and D. Kochmann. Inverse design of nonlinear mechanical metamaterials via video denoising diffusion models. *Nature Machine Intelligence*, 5:1466–1475, 2023. doi: 10.1038/s42256-023-00762-x.

[18] Li Zheng, K. Karapiperis, Siddhant Kumar, and D. Kochmann. Unifying the design space and optimizing linear and nonlinear truss metamaterials by generative modeling. *Nature Communications*, 14, 2023. doi: 10.1038/s41467-023-42068-x.

[19] Daniel A. White, William J. Arrighi, Jun Kudo, and Seth E. Watts. Multiscale topology optimization using neural network surrogate models. *Computer Methods in Applied Mechanics and Engineering*, 346:1118–1135, 2019. doi: 10.1016/j.cma.2018.09.007.

[20] Bolei Deng, Ahmad Zareei, Xiaoxiao Ding, James C. Weaver, Chris H. Rycroft, and Katia Bertoldi. Inverse design of mechanical metamaterials with target nonlinear response via a neural accelerated evolution strategy. *Advanced Materials*, 34(41), 2022. doi: 10.1002/adma.202206238.

[21] T. Xue, Shuheng Liao, Zhengtao Gan, Chanwook Park, Xiaoyu Xie, Wing Kam Liu, and Jian Cao. JAX-FEM: A differentiable GPU-accelerated 3D finite element solver for automatic inverse design and mechanistic data science. *Computer Physics Communications*, 291:108802, 2023. doi: 10.1016/j.cpc.2023.108802.

[22] David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems 31*, pages 2451–2463, 2018. URL https://arxiv.org/abs/1809.01999.

[23] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy P. Lillicrap. Mastering diverse domains through world models. *CoRR*, abs/2301.04104, 2023. URL https://arxiv.org/abs/2301.04104.

[24] Sherry Yang, Yilun Du, Seyed Kamyar Seyed Ghasemipour, Jonathan Tompson, Leslie Pack Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. In *The Twelfth International Conference on Learning Representations, ICLR 2024*, 2024. URL https://arxiv.org/abs/2310.06114.

[25] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier J. Hénaff, Matthew M. Botvinick, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver IO: A general architecture for structured inputs & outputs. In *The Tenth International Conference on Learning Representations, ICLR 2022*, 2022. URL https://arxiv.org/abs/2107.14795.

[26] Markus J Buehler. Fieldperceiver: Domain agnostic transformer model to predict multiscale physical fields and nonlinear material properties through neural ologs. *Materials Today*, 57: 9–25, 2022. doi: 10.1016/j.mattod.2022.05.020.

[27] Haixu Wu, Tengge Hu, Huakun Luo, Jianmin Wang, and Mingsheng Long. Solving high-dimensional pdes with latent spectral models. In *International Conference on Machine Learning, ICML 2023*, Proceedings of Machine Learning Research, pages 37417–37438. PMLR, 2023. URL https://arxiv.org/abs/2301.12664.

[28] Benedikt Alkin, Andreas Fürst, Simon Schmid, Lukas Gruber, Markus Holzleitner, and Johannes Brandstetter. Universal Physics Transformers: A Framework For Efficiently Scaling Neural Operators. In *Advances in Neural Information Processing Systems 37*, 2024. URL https://arxiv.org/abs/2402.12365.

[29] Louis Serrano, Thomas X. Wang, Etienne Le Naour, Jean-Noël Vittaut, and Patrick Gallinari. AROMA: Preserving spatial structure for latent PDE modeling with local neural fields. In *Advances in Neural Information Processing Systems 37, NeurIPS 2024*, 2024. URL https://arxiv.org/abs/2406.02176.

[30] Tian Wang and Chuang Wang. Latent neural operator for solving forward and inverse PDE problems. In *Advances in Neural Information Processing Systems 37, NeurIPS 2024*, 2024. URL https://arxiv.org/abs/2406.03923.

[31] Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for pdes on general geometries. *Journal of Machine Learning Research*, 24:388:1–388:26, 2023. URL https://jmlr.org/papers/v24/23-0064.html.

[32] Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. Battaglia. Learning mesh-based simulation with graph networks. In *9th International Conference on Learning Representations, ICLR 2021*, 2021. URL https://arxiv.org/abs/2010.03409.

[33] Rishit Dagli, Donglai Xiang, Vismay Modi, Charles Loop, Clement Fuji Tsang, Anka He Chen, Anita Hu, Gavriel State, David I.W. Levin, and Maria Shugrina. VoMP: Predicting volumetric mechanical property fields. *arXiv preprint arXiv:2510.22975*, 2025. URL https://arxiv.org/abs/2510.22975.

[34] Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal Behbahani, Stephanie Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative interactive environments. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 4603–4623. PMLR, 2024. URL https://arxiv.org/abs/2402.15391.

[35] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023*, pages 4195–4205, 2023. doi: 10.1109/ICCV51070.2023.00387.

[36] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 3942–3951, 2018. URL https://arxiv.org/abs/1709.07871.

[37] Huakun Luo, Haixu Wu, Hang Zhou, Lanxiang Xing, Yichen Di, Jianmin Wang, and Mingsheng Long. Transolver++: An accurate neural solver for PDEs on million-scale geometries. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 41432–41449. PMLR, 2025. URL https://arxiv.org/abs/2502.02414.

[38] Bjoern List, Li-Wei Chen, Kartik Bali, and Nils Thuerey. Differentiability in Unrolled Training of Neural Physics Simulators on Transient Dynamics. *Computer Methods in Applied Mechanics and Engineering*, 433:117441, 2025. doi: 10.1016/j.cma.2024.117441.

[39] Tai Hoang, Alessandro Trenta, Alessio Gravina, Niklas Freymuth, Philipp Becker, Davide Bacciu, and Gerhard Neumann. Improving long-range interactions in graph neural simulators via Hamiltonian dynamics. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2511.08185.

[40] Xuan Zhang, Jacob Helwig, Yuchao Lin, Yaochen Xie, Cong Fu, Stephan Wojtowytsch, and Shuiwang Ji. SineNet: Learning temporal dynamics in time-dependent partial differential equations. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2403.19507.

[41] Wojciech M. Czarnecki, Simon Osindero, Max Jaderberg, Grzegorz Swirszcz, and Razvan Pascanu. Sobolev training for neural networks. In *Advances in Neural Information Processing Systems 30*, pages 4278–4287, 2017. URL https://arxiv.org/abs/1706.04859.

[42] Namkyeong Cho, Junseung Ryu, and Hyung Ju Hwang. Sobolev Training for Operator Learning. *Journal of Computational Physics*, 542:114319, 2025. doi: 10.1016/j.jcp.2025.114408.

[43] Jong Kwon Oh, Hanbaek Lyu, and Hwijae Son. Sobolev acceleration for neural networks. 2025. URL https://arxiv.org/abs/2509.19773.

[44] Boyuan Yao, Dingcheng Luo, Lianghao Cao, Nikola Kovachki, Thomas O'Leary-Roseberry, and Omar Ghattas. Derivative-Informed Fourier Neural Operator: Universal Approximation and Applications to PDE-Constrained Optimization. 2025. URL https://arxiv.org/abs/2512.14086.

[45] Kaushik Bhattacharya, Lianghao Cao, George Stepaniants, Andrew Stuart, and Margaret Trautner. Learning Memory and Material Dependent Constitutive Laws. 2025. URL https://arxiv.org/abs/2502.05463.

[46] Yuan Qiu, Nolan Bridges, and Peng Chen. Derivative-enhanced deep operator network. In *Advances in Neural Information Processing Systems 37, NeurIPS 2024*, 2024. URL https://arxiv.org/abs/2402.19242.

[47] Jeremy Yu, Lu Lu, Xuhui Meng, and George Em Karniadakis. Gradient-enhanced physics-informed neural networks for forward and inverse PDE problems. *Computer Methods in Applied Mechanics and Engineering*, 393:114823, 2022. doi: 10.1016/j.cma.2022.114823.

[48] Johannes Gasteiger, Florian Becker, and Stephan Günnemann. Gemnet: Universal directional graph neural networks for molecules. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 6790–6802, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/35cf8659cfcb13224cbd47863a34fc58-Abstract.html.

[49] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd E. Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 5481–5490. IEEE, 2022. doi: 10.1109/CVPR52688.2022.00541.

[50] Zhenguo Nie, Haoliang Jiang, and Levent Burak Kara. Stress field prediction in cantilevered structures using convolutional neural networks. *Journal of Computing and Information Science in Engineering*, 20, 2020. doi: 10.1115/1.4044097.

[51] Jaber Rezaei Mianroodi, Nima H. Siboni, and Dierk Raabe. Teaching solid mechanics to artificial intelligence—a fast solver for heterogeneous materials. *npj Computational Materials*, 7, 2021. doi: 10.1038/s41524-021-00571-z.

[52] Meer Mehran Rashid, Tanu Pittie, Souvik Chakraborty, and N.M. Anoop Krishnan. Learning the stress-strain fields in digital composites using Fourier neural operator. *iScience*, 25:105452, 2022. doi: 10.1016/j.isci.2022.105452.

[53] Mohammad S. Khorrami, Jaber R. Mianroodi, Nima H. Siboni, Pawan Goyal, Bob Svendsen, Peter Benner, and Dierk Raabe. An artificial neural network for surrogate modeling of stress fields in viscoplastic polycrystalline materials. *npj Computational Materials*, 9, 2023. doi: 10.1038/s41524-023-00991-z.

[54] Diab W. Abueidda, Seid Koric, Nahil A. Sobh, and Huseyin Sehitoglu. Deep learning for plasticity and thermo-viscoplasticity. *International Journal of Plasticity*, 136:102852, 2021. doi: 10.1016/j.ijplas.2020.102852.

[55] Junyan He, Seid Koric, Shashank Kushwaha, Jaewan Park, Diab Abueidda, and Iwona Jasiuk. Novel DeepONet architecture to predict stresses in elastoplastic structures with variable complex geometries and loads. *Computer Methods in Applied Mechanics and Engineering*, 415:116277, 2023. doi: 10.1016/j.cma.2023.116277.

[56] Junyan He, Seid Koric, Diab Abueidda, Ali Najafi, and Iwona Jasiuk. Geom-DeepONet: A point-cloud-based deep operator network for field predictions on 3D parameterized geometries. *Computer Methods in Applied Mechanics and Engineering*, 429:117130, 2024. doi: 10.1016/j.cma.2024.117130.

[57] Qibang Liu, Weiheng Zhong, Hadi Meidani, Diab Abueidda, Seid Koric, and Philippe Geubelle. Geometry-Informed Neural Operator Transformer. 2025. URL https://arxiv.org/abs/2504.19452.

[58] Weiheng Zhong, Qibang Liu, Diab Abueidda, Seid Koric, and Hadi Meidani. A comprehensive comparison of neural operators for 3D industry-scale engineering designs, 2025. URL https://arxiv.org/abs/2510.05995.

[59] Mohammad S. Khorrami, Pawan Goyal, Jaber R. Mianroodi, Bob Svendsen, Peter Benner, and Dierk Raabe. A physics-encoded Fourier neural operator approach for surrogate modeling of divergence-free stress fields in solids. 2024. URL https://arxiv.org/abs/2408.15408.

[60] Mikel M. Iparraguirre, Iciar Alfaro, David Gonzalez, and Elias Cueto. MeshGraphNet-Transformer: Scalable mesh-based learned simulation for solid mechanics. 2026. URL https://arxiv.org/abs/2601.23177.

[61] Kevin Linka and Ellen Kuhl. A new family of Constitutive Artificial Neural Networks towards automated model discovery. *Computer Methods in Applied Mechanics and Engineering*, 403: 115731, 2023. doi: 10.1016/j.cma.2022.115731.

[62] Marius Tacke, Matthias Busch, Kian Abdolazizi, Jonas Eichinger, Kevin Linka, Christian Cyron, and Roland Aydin. Automating modeling in mechanics: LLMs as designers of physics-constrained neural networks for constitutive modeling of materials. 2025. URL https://arxiv.org/abs/2512.01735.

[63] Haixu Wu, Minghao Guo, Zongyi Li, Zhiyang Dou, Mingsheng Long, Kaiming He, and Wojciech Matusik. GeoPT: Scaling physics simulation via lifted geometric pre-training. 2026. URL https://arxiv.org/abs/2602.20399.

[64] Mohammad Amin Nabian, Sudeep Chavare, Deepak Akhare, Rishikesh Ranade, Ram Cherukuri, and Srinivas Tadepalli. Automotive Crash Dynamics Modeling Accelerated with Machine Learning. 2025. URL https://arxiv.org/abs/2510.15201.

[65] Benjamin Holzschuh, Georg Kohl, Florian Redinger, and Nils Thuerey. P3D: Highly scalable 3D neural surrogates for physics simulations with global context. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2509.10186.

[66] Winfried van den Dool, Maksim Zhdanov, Yuki M. Asano, and Max Welling. Adaptive mesh-quantization for neural PDE solvers, 2025. URL https://arxiv.org/abs/2511.18474.

[67] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://papers.nips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[68] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017. URL https://arxiv.org/abs/1612.01474.

[69] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016. URL https://arxiv.org/abs/1506.02142.

[70] Eric M. Stewart and Lallit Anand. A large deformation viscoelasticity theory for elastomeric materials and its numerical implementation in the open-source finite element program FEniCSx. *International Journal of Solids and Structures*, 303:113023, 2024. doi: 10.1016/j.ijsolstr.2024.113023.

[71] Johannes Brandstetter, Daniel E. Worrall, and Max Welling. Message passing neural PDE solvers. In *The Tenth International Conference on Learning Representations, ICLR 2022*, 2022. URL https://arxiv.org/abs/2202.03376.

A Constitutive Models and FEM Solver

We consider finite deformation solid mechanics on a body in reference configuration Ω_0. The displacement field $\mathbf{u}(\mathbf{X}, t)$ maps material points $\mathbf{X}$ to deformed positions $\mathbf{x} = \mathbf{X} + \mathbf{u}$, with deformation gradient $\mathbf{F} = \mathbf{I} + \nabla\mathbf{u}$, Jacobian $J = \det \mathbf{F}$, and right Cauchy-Green tensor $\mathbf{C} = \mathbf{F}^\top \mathbf{F}$. The balance of linear momentum in the reference configuration reads

$$\operatorname{Div} \mathbf{P} = \rho_0 \, \ddot{\mathbf{u}}, \tag{A1}$$

where $\mathbf{P}$ is the first Piola-Kirchhoff stress and ρ_0 is the reference density. For hyperelastic materials the stress derives from a free energy density Ψ_{R} per unit reference volume:

$$\mathbf{P} = 2\mathbf{F} \, \frac{\partial \Psi_{\mathrm{R}}}{\partial \mathbf{C}}. \tag{A2}$$

The Cauchy stress is $\boldsymbol{\sigma} = J^{-1} \mathbf{P} \mathbf{F}^\top$.

Neo-Hookean hyperelasticity. A compressible Neo-Hookean free energy is used,

$$\Psi_{\mathrm{R}} = \frac{\mu}{2}(\operatorname{tr} \mathbf{C} - 3) - \mu \ln J + \frac{\lambda}{2}(\ln J)^2, \tag{A3}$$

where the parameters $\mu = 1$, $\lambda = 10$. In the Neo-Hookean setup we solve the systems in the quasi-static limit: the inertial term vanishes ($\rho_0 \ddot{\mathbf{u}} \to \mathbf{0}$), and there are no additional internal variables.

Finite viscoelasticity with inertia. In contrast, the visco-hyperelastic plate has both inertia and rate-dependent internal state, using the constitutive model of Stewart and Anand [70]. The free energy decomposes into equilibrium and non-equilibrium contributions:

$$\Psi_{\mathrm{R}} = \Psi_{\mathrm{R}}^{\mathrm{eq}}(\bar{\mathbf{C}}) + \sum_{i=1}^{N_{\mathrm{b}}} \Psi_{\mathrm{R}}^{\mathrm{neq}(i)}(\bar{\mathbf{C}}, \mathbf{A}^{(i)}) + \frac{\kappa}{2}(J-1)^2, \tag{A4}$$

where $\bar{\mathbf{C}} = J^{-2/3}\mathbf{C}$ is the distortional Cauchy-Green tensor and κ is the bulk modulus enforcing near-incompressibility. The equilibrium term uses the Arruda-Boyce eight-chain model:

$$\Psi_{\mathrm{R}}^{\mathrm{eq}}(\bar{\mathbf{C}}) = G_{\mathrm{eq}} \, \lambda_L^2 \left[\frac{\bar{\lambda}}{\lambda_L} \, \beta_L\!\left(\frac{\bar{\lambda}}{\lambda_L}\right) + \ln \frac{\beta_L(\bar{\lambda}/\lambda_L)}{\sinh \beta_L(\bar{\lambda}/\lambda_L)} - C_0 \right], \tag{A5}$$

where $\bar{\lambda} = \sqrt{\operatorname{tr}(\bar{\mathbf{C}})/3}$ is the chain stretch, λ_L is the locking stretch, $\beta_L(z) = z(3 - z^2)/(1 - z^2)$ is an approximation of the inverse Langevin function, and C_0 is a constant ensuring $\Psi_{\mathrm{R}}^{\mathrm{eq}} = 0$ in the reference configuration. Each non-equilibrium branch uses a neo-Hookean form in the distortional elastic deformation relative to the internal variable $\mathbf{A}^{(i)}$:

$$\Psi_{\mathrm{R}}^{\mathrm{neq}(i)}(\bar{\mathbf{C}}, \mathbf{A}^{(i)}) = \frac{G_{\mathrm{neq}}^{(i)}}{2}\big(\bar{\mathbf{C}} : (\mathbf{A}^{(i)})^{-1} - 3\big). \tag{A6}$$

Each internal variable evolves toward thermodynamic equilibrium:

$$\dot{\mathbf{A}}^{(i)} = \frac{1}{\tau^{(i)}}\big(\bar{\mathbf{C}} - \mathbf{A}^{(i)}\big), \quad i = 1, \ldots, N_{\mathrm{b}}, \quad \det \mathbf{A}^{(i)} = 1, \tag{A7}$$

with relaxation times $\tau^{(i)}$. Material parameters: $G_{\mathrm{eq}} = 200\,\mathrm{kPa}$, $\lambda_L = 10$, $\kappa = 400{,}000\,\mathrm{kPa}$, $\rho_0 = 1.3 \times 10^{-5}\,\mathrm{kg/mm^3}$, $N_{\mathrm{b}} = 3$ branches:

Branch	1	2	3
G_{neq} (kPa)	300	600	150
τ (s)	0.001	0.2	3.0

Initial and boundary conditions. All simulations start from the undeformed reference configuration: $\mathbf{u} = \mathbf{0}$, and for the visco-hyperelastic plate additionally $\dot{\mathbf{u}} = \mathbf{0}$, $\ddot{\mathbf{u}} = \mathbf{0}$, $\mathbf{A}^{(i)} = \mathbf{I}$ (stress-free equilibrium). The left face ($x = x_{\min}$) is fully clamped (zero displacement); the right face ($x = x_{\max}$) is the controlled grip. The remaining faces are traction-free.

FEM solver details. Both regimes are solved via the standard Galerkin weak form of the momentum balance. For the visco-hyperelastic plate, the mixed displacement-pressure formulation finds $(\mathbf{u}, p) \in V_h \times Q_h$ such that

$$\int_{\Omega_0} \mathbf{P} : \nabla \mathbf{v} \, \mathrm{d}V + \int_{\Omega_0} \rho_0 \, \ddot{\mathbf{u}} \cdot \mathbf{v} \, \mathrm{d}V = 0 \quad \forall \mathbf{v} \in V_h, \qquad \int_{\Omega_0} \left(J - 1 + \frac{p}{\kappa} \right) q \, \mathrm{d}V = 0 \quad \forall q \in Q_h, \tag{A8}$$

where the first equation is the momentum balance and the second enforces near-incompressibility through the pressure penalty. For the lattice regime, the quasi-static pure displacement formulation finds $\mathbf{u} \in V_h$ such that

$$\int_{\Omega_0} \mathbf{P} : \nabla \mathbf{v} \, \mathrm{d}V = 0 \quad \forall \mathbf{v} \in V_h. \tag{A9}$$

The visco-hyperelastic plate was generated with FEniCSx (DOLFINx 0.8) using Taylor-Hood elements: CG2 (quadratic Lagrange) for displacement and CG1 (linear Lagrange) for pressure. Internal variables $\mathbf{A}^{(i)}$ are stored at quadrature points (degree 2). Nonlinear solver: Newton iteration with MUMPS direct LU ($\mathrm{atol} = \mathrm{rtol} = 10^{-8}$ on the displacement increment norm, max 50 iterations). Time integration: Newmark-beta (trapezoidal rule, $\gamma = 0.5$, $\beta = 0.25$, $\mathrm{d}t = 0.002\,\mathrm{s}$, 5 FEM substeps per action step).

The lattice regime dataset was generated with legacy FEniCS (`dolfin` 2019.1.0) using CG1 (linear Lagrange) displacement elements, pure displacement formulation. Newton iteration with MUMPS ($\mathrm{rtol} = 10^{-3}$, $\mathrm{atol} = 10^{-10}$ on the residual norm, max 15 iterations). Quasi-static incremental loading over 30 steps. Actions use the same per-DOF block sampling as the visco-hyperelastic plate, with per-step magnitudes 0.15 (stretch, shear_y, shear_z) and 0.08 (twist). FEM data generation ran on CPU, with up to 200 parallel FEniCS workers (each trajectory solved on a single worker).

Stress calculation from predicted displacement. For models without a stress head, we compute stress from the predicted displacement field using the standard finite element procedure. Shape function gradients on the tetrahedral mesh yield the deformation gradient $\mathbf{F} = \mathbf{I} + \nabla \hat{\mathbf{u}}$ at element centroids, the constitutive law maps $\mathbf{F}$ to Cauchy stress $\boldsymbol{\sigma}$, and volume-weighted averaging projects element stresses to nodes. The von Mises stress is $\sigma_{\mathrm{VM}} = \sqrt{\frac{3}{2}\,\mathbf{s} : \mathbf{s}}$, where $\mathbf{s} = \boldsymbol{\sigma} - \frac{1}{3}(\mathrm{tr}\,\boldsymbol{\sigma})\,\mathbf{I}$ is the deviatoric stress. For models with the stress head, von Mises stress is computed directly from the predicted $\hat{\boldsymbol{\sigma}}_{\mathrm{sym}}$ without any mesh-based differentiation. For ablation variants with a gradient head, the predicted displacement gradient is mapped to Cauchy stress via the constitutive law.

B MicroPlate Benchmark Details

Graph representation. The unit cell of each architected lattice is defined by a compact graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{\mathbf{v}_i \in \mathbb{R}^3\}$ are node positions and $\mathcal{E} = \{(i, j, r)\}$ are beams connecting pairs of nodes with radius r.

An example graph with 4 nodes and 4 beams:

```
{"nodes": [[0.24, 0.61, 0.46],
           [0.15, 0.18, 0.19],
           [0.44, 0.57, 0.85],
           [0.64, 0.12, 0.51]],
 "beams": [{"idx": [0,1], "r": 0.017},
           {"idx": [1,2], "r": 0.011},
           {"idx": [2,3], "r": 0.016},
           {"idx": [3,0], "r": 0.014}]}
```

Cubic symmetry. To ensure that the resulting microstructure has full cubic symmetry, every beam in the graph is replicated under all 48 operations of the octahedral symmetry group O_h (the 6 coordinate permutations combined with the 8 sign combinations). A graph with $|\mathcal{E}|$ beams thus produces $48\,|\mathcal{E}|$ beams in the expanded unit cell (Fig. 6). This ensures that the mechanical response of each topology is invariant under the full cubic point group.

Mesh generation. The expanded beam geometry is evaluated as a signed distance field (SDF) on a regular voxel grid at 30 voxels per unit cell, using smooth-min blending between beam capsules to ensure connected junctions. The unit cell is tiled $5 \times 5 \times 1$ into a $[10 \times 10 \times 2]$ plate geometry via modular coordinate wrapping: for each evaluation point, the SDF kernel maps the global plate coordinate back to the local unit cell coordinate, so the lattice pattern repeats seamlessly across tile boundaries without explicit stitching. An auto-scaling step ensures the structure fills the unit cell boundary, and a percolation check verifies that the tiled lattice forms a single connected component. The isosurface is extracted via marching cubes, producing a triangular surface mesh that is then volumetrically meshed with TetGen. Mesh sizes range from 71,000 to 442,000 nodes (250,000–2,020,000 tetrahedra). The complex stress concentration patterns at strut junctions distinguish the lattice plates from the homogeneous plate, where stress fields are smooth.

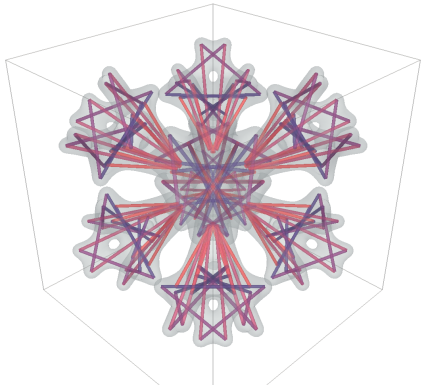

Figure 6: **Unit cell of a MicroPlate lattice.** The compact graph (4 beams, shown as colored rods) is replicated under all 48 operations of the cubic symmetry group O_h, producing $48 \times 4 = 192$ beams whose smooth-min-blended signed distance field yields the final mesh (gray). Each color represents one seed beam and all its symmetric copies.

Dataset statistics. Each topology has multiple boundary condition trajectories (8,836 total across 63 topologies), each 30 frames under 4-DOF random discrete actions simulated using different random seeds. Material: Neo-Hookean, $\mu = 1$, $\lambda = 10$. MicroPlate is split by topology into 55 training and 8 held-out plates. Each of the 55 training plates has more than 100 FEM trajectories; the first 100 trajectories (5,500 in total) are used for training and the 101st trajectory (a different random loading path) is used for evaluation. The 1,000 visco-hyperelastic-plate trajectories are split 800/100/100 (train/val/test) with seed 42. Figure 7 shows all 63 MicroPlate unit cells grouped by split, Fig. 8 shows the PCA distribution of topology features, and Table 3 compares MicroPlate's scale against existing solid mechanics benchmarks.

Table 3: **MicroPlate vs. existing solid mechanics benchmarks.** MicroPlate is two orders of magnitude larger in mesh size, and it is a temporal 3D benchmark with microstructure diversity.

Benchmark	Nodes	Temporal	3D	Nonlinear	History-dep.
Geo-FNO Elasticity [31]	∼1K	No	No	Yes	No
MGN DeformingPlate [32]	∼1K	Yes	Yes	Yes	No
LatticeGraphNet [15]	∼5K	Yes	Yes	Yes	No
MicroPlate (ours)	**71K–442K**	**Yes**	**Yes**	**Yes**	**Yes**

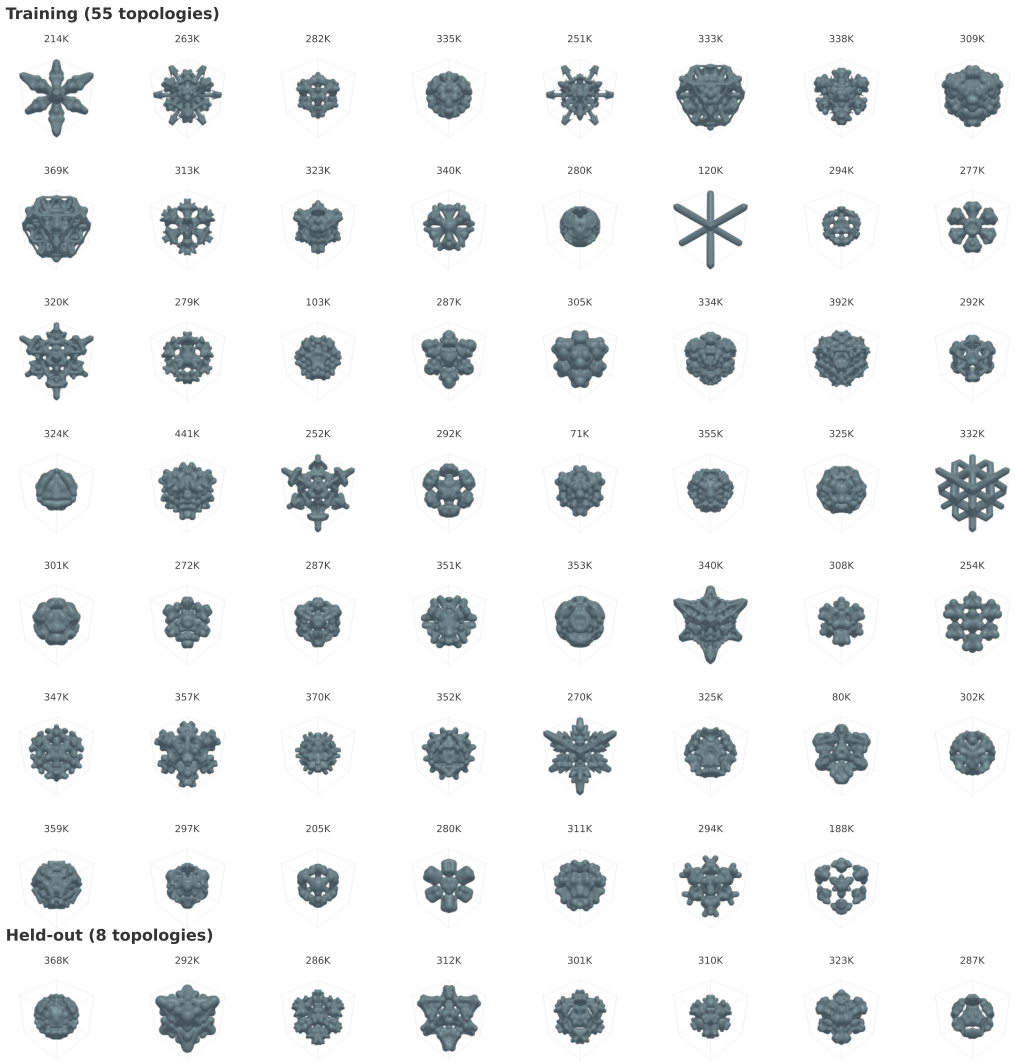

Figure 7: **Library of MicroPlate unit cells.** Unit cells (RVEs) corresponding to the 63 architected lattices, shown after O_h symmetry expansion. Top: 55 training topologies. Bottom: 8 held-out topologies for evaluating generalization to unseen geometries. The number above each cell is the plate mesh size (in thousands of nodes) after $5 \times 5 \times 1$ tiling and TetGen volumetric meshing.

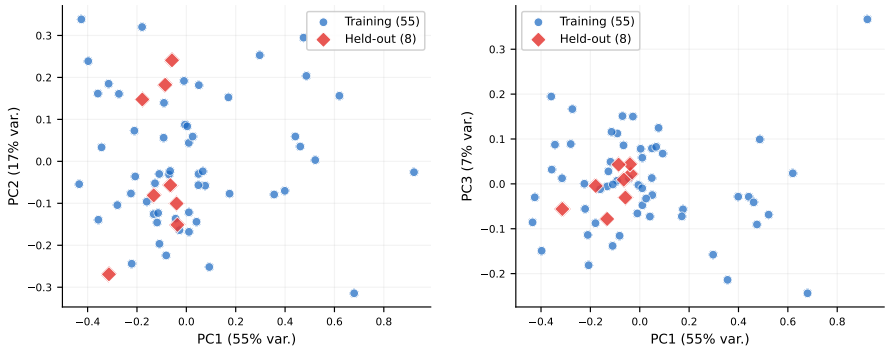

Figure 8: **PCA projection of architected-lattice features.** PCA projection of the 15-dimensional graph feature vectors (node/beam counts, spatial spread, beam lengths, radii, connectivity) for all 63 architected lattices.

C Detailed Training Procedure

LEIA is trained in two stages. First, the tokenizer is trained on individual frames to reconstruct displacement (and, when using the stress head, Cauchy stress) from the latent representation. Second, with the tokenizer frozen, the dynamics transformer is trained to predict the next latent state from the current state and boundary condition action.

Stress supervision. The tokenizer is trained per frame with a combined loss $\mathcal{L}_{\text{tok}} = \|\hat{\mathbf{u}} - \mathbf{u}\|^2 + \beta \|\hat{\boldsymbol{\sigma}}_{\text{sym}} - \boldsymbol{\sigma}_{\text{sym}}^{\text{FEM}}\|^2$, where $\boldsymbol{\sigma}_{\text{sym}}^{\text{FEM}}$ is the Cauchy stress from the finite element solver, stored as 6 symmetric components. Because stress magnitudes can differ from displacements by orders of magnitude, we normalize each output component to zero mean and unit variance before computing the loss. The stress head itself is a single $\text{Linear}(H, 6)$ layer ($< 0.02\%$ of model parameters). The same stress head applies across architectures: we attach it to UPT, Transolver, PointNet, and MeshGraphNets by replacing each model's output layer with one producing $\mathbf{u}(3) + \boldsymbol{\sigma}_{\text{sym}}(6) = 9$ output dimensions. No other architectural changes are made.

The output dimension of the stress head is constant regardless of constitutive law complexity. The Cauchy stress $\boldsymbol{\sigma}$ is always a symmetric 3×3 tensor with 6 independent components, whether the material is Neo-Hookean or viscoelastic. By contrast, predicting all internal variables scales from 3 output dimensions (Neo-Hookean, displacement only) to 22 (3-branch visco-hyperelastic: displacement + 18 $\mathbf{A}^{(i)}$ components + pressure) or higher for more complex constitutive laws.

Furthest point sampling. For the lattice regime's meshes (up to 442,000 nodes), encoding all nodes simultaneously is prohibitive. We subsample S query points via furthest point sampling (FPS), which provides approximately uniform spatial coverage of the mesh. The Perceiver encoder compresses these S sampled node features into K latent tokens. At decoding time, the decoder reconstructs the field at all N mesh nodes by querying the latent tokens from each node's position.

Dynamics training. With the tokenizer frozen, the dynamics transformer is trained to predict the next latent state: $\mathcal{L}_{\text{dyn}} = \|\mathbf{z}_{t+1}^{\text{pred}} - \mathbf{z}_{t+1}^{\text{target}}\|^2$, where $\mathbf{z}_{t+1}^{\text{target}}$ is the encoding of the ground-truth next frame. Optionally, pushforward training [71] unrolls the model for k steps and averages the loss over the rollout, reducing the distribution shift between teacher-forced training and autoregressive inference. For viscoelastic systems where stress depends on deformation history, the dynamics model can additionally be conditioned on the latent velocity $\mathbf{z}_t - \mathbf{z}_{t-1}$, concatenated with the current state before projection. Per-dataset training configurations are reported in Table 4.

Ablations. The first five rows in Table 1 share the same architecture (n_input_frames=1, K=512, H=1024, 6 encoder layers, no decoder layers), the same dynamics recipe (single-step continuous), and the same downstream pipeline; only the tokenizer's stress-supervision strategy differs. The last row applies our production dynamics enhancements (pushforward rollout and velocity-input conditioning) on top of the stress head, isolating the contribution of dynamics-side training from tokenizer supervision. The supervision-loss term and its weight differ per row. The stress head uses $\beta = 1$ on the standardized Cauchy MSE term from the tokenizer training loss, with linear warmup over the first 10K of 30K tokenizer steps. Tet and autograd Sobolev use Sobolev MSE on $\nabla \mathbf{u}$ with $\beta = 3$ ($\beta = 30$ destabilizes reconstruction for these paths). The gradient head uses Sobolev MSE with $\beta = 30$, the larger weight yielding the higher VM correlation reported in Table 1.

Baselines. All four baseline architectures (UPT, Transolver, PointNet, MeshGraphNets) are trained end-to-end on the same data splits. The only modification to each baseline is the output dimension: 3 (displacement only), 9 (displacement + 6 Cauchy stress components with the stress head), or 22 (displacement + 18 internal variable components + pressure, without the stress head on the visco-hyperelastic plate).

Table 4 reports the full architecture and optimizer configurations of LEIA. All models are trained on $8\times$H100 80 GB GPUs unless noted otherwise.

Table 4: **Hyperparameters for LEIA and all baselines.**

Model	Architecture	Training
LEIA tokenizer (lattice regime)	Fourier PE (16 bands), H=1024, K=512 latents, L_{enc}=6, L_{dec}=0, 4 heads, dropout 0.1, input: 3-frame disp(9) + Fourier PE of ref coords, FPS 8192+2048 random nodes $\rightarrow$ K latents, ≈103.5M params	AdamW 10^{-4}, wd 10^{-5}, CosineAnnealingLR, batch 8 ($\times$2 accum), bf16, 100K steps, 8×H100
LEIA dynamics (lattice regime)	H=1024, 12 layers, 8 heads, dropout 0.3, action: 4-dim cumulative BC, FiLM: MLP(4→1024→1024, GELU) + per-layer SiLU proj to $(\gamma, \beta)\times 2$, velocity input ($\mathbf{z}_t - \mathbf{z}_{t-1}$), ≈205.2M params	AdamW 10^{-4}, wd 10^{-4}, push-forward (k=10), noise std 0.02, warmup (10 ep) + CosineAnnealingLR, batch 64, bf16, 300 ep, 8×H100
LEIA tokenizer (visco-hyperelastic plate)	H=256, K=256 latents, L_{enc}=8, L_{dec}=4, 4 heads, dropout 0.1, input: 3-frame disp(9) + Fourier PE, stress head, all 363 nodes (no FPS), ≈11M params	AdamW 3×10^{-4}, wd 10^{-5}, CosineAnnealingLR, batch 64, bf16, 30K steps, 8×H100
LEIA dynamics (visco-hyperelastic plate)	H=256, 12 layers, 8 heads, dropout 0.3, action: 4-dim discrete BC, velocity input ($\mathbf{z}_t - \mathbf{z}_{t-1}$), ≈13M params	AdamW 3×10^{-4}, wd 10^{-4}, push-forward (k=20), CosineAnnealingLR, batch 32, bf16, 300 ep, 8×H100
UPT	CfdPool GNN + Perceiver (256 tokens, H=384, 4 enc + 6 latent + 4 dec layers), AdaLN: timestep(192) + BC(4) $\rightarrow$ cond(768), input: 3-frame disp, ≈49.7M params	AdamW 5×10^{-4}, wd 0.05, warmup (20 ep) + CosineAnnealingLR, bf16, batch 24, 200 ep, 6×H100
Transolver	Physics-attention, 128 hidden, 8 heads, 8 layers, 64 slices, `fun_dim`=7 (BC(4)+vel(3)), `out_dim` $\in \{3, 9, 22\}$, ≈983K params	AdamW 10^{-4}, wd 10^{-5}, CosineAnnealingLR, batch 4, 200 ep, 1×H100
PointNet	Global max-pool MLP, `n_hidden`=64, `fun_dim`=7 (BC(4)+vel(3)), `out_dim` $\in \{3, 9, 22\}$, ≈3.7M params	AdamW 10^{-4}, wd 10^{-5}, CosineAnnealingLR, batch 8, 200 ep, 1×H100
MeshGraphNets	Enc-Proc-Dec, 15 MP layers, hidden 128, node features: type(9)+disp(3)+σ_{sym}(6)+vel(3)+bc(4)=25-dim, ≈2.9M params	Adam 10^{-4}, running normalizer, batch 20/GPU, 100 ep, 6×H100

D Beam Search Mutation Operators

The surrogate-guided design search (Section 4.3) uses beam search with stochastic graph mutations. At each iteration, each of the top-B=5 candidates is expanded by generating $\lfloor M \times 1.5 \rfloor$=12 mutation attempts per parent (over-generating to account for invalid mutations), for a target of M=8 valid candidates per parent. Each mutation randomly composes 1–3 operators (with probabilities 0.6, 0.3, 0.1 for 1, 2, 3 operators respectively), drawn from the distribution in Table 5 and applied sequentially to the graph. After each mutation, the graph is validated: node coordinates must lie in $[-1.5, 1.5]$, beam radii in $(0, 0.5]$, volume fraction in $[0.001, 0.5]$, and the lattice must remain connected (all nodes reachable by graph traversal). Invalid mutations are rejected and resampled up to 20 times. Valid candidates are meshed via the pipeline in Appendix B and evaluated with LEIA on two loading directions (stretch and shear) for 20 steps each to obtain force-displacement curves. The design metric is $s = W_{\text{stretch}}/(W_{\text{shear}} \cdot v_f + \varepsilon)$, where work W is the trapezoidal integral of |force| over |displacement|, v_f is the normalized volume fraction, and $\varepsilon = 10^{-8}$. The search runs for D=10 iterations, evaluating 553 total candidates from seed topology `mg_046`.

Table 5: **Mutation operators for surrogate-guided architected-lattice search.** Each mutation randomly composes 1–3 operators (drawn with the listed probabilities), validated for graph connectivity before evaluation.

Operator	Description	Parameters	Prob.
Perturb nodes	Displace 1–3 random nodes	$\delta \sim \mathcal{N}(0, 0.08)$ per axis	25%
Scale radii	Multiply radii of 1–4 random beams	Factor $\sim e^{\mathcal{N}(0,0.15)}$	20%
Add beam	Connect two unconnected nodes	Radius $\sim$ Uniform$(0.005, 0.04)$	15%
Remove beam	Delete a random beam	Requires connectivity check	15%
Split beam	Insert node at beam midpoint	Creates two beams, interpolates radii	15%
Randomize radii	Perturb all beam radii independently	Per-beam factor $\sim e^{\mathcal{N}(0,0.15)}$	10%

E Out-of-distribution Detection and Confidence Head

Candidate generation. We generate 7,981 mutated MicroPlate candidates from the 55 training topologies in two passes. Pass 1 (3,080 candidates): unguided random walks of depth $D = 55$ from each training plate (55 seeds $\times$ 56 entries per walk = 3,080), using the mutation operators of Table 5 with the same composition rules as the design search of Appendix D. Pass 2 (4,901 candidates): a 55-seed novelty-search variant with a shared global anchor list. Each training plate seeds an independent beam (width $B = 2$); at each of $D = 8$ iterations every beam member is expanded by $K = 6$ aggressive mutations (perturb $\sigma = 0.3$, scale $\sigma = 0.5$, plus a k-node subgraph-replace operator with $k \in \{2, 3, 4\}$). Per-seed top-B selection uses k-NN distance ($k = 15$) to a global anchor list seeded with the 3,080 Pass 1 latent projections (the 55 training plates plus their 3,025 random-walk mutants) and updated with every Pass 2 evaluated candidate; this couples the per-seed walks so that seeds in dense regions migrate into unexplored directions. Of the 7,981 candidates, 6,919 obtained valid FEM characterization under the protocol of Appendix D (1,062 diverged at the maximum strain level); the latter retain features but have no ρ label and contribute only to inference-time visualization.

Reconstruction error. For each candidate we encode the zero-displacement field (rest configuration), decode it back, and compute the mean squared displacement $\langle \|\hat{\mathbf{u}}_0\|^2 \rangle$, where $\hat{\mathbf{u}}_0 = \mathrm{decode}(\mathrm{encode}(\mathbf{0}))$. This scalar measures how well the tokenizer round-trips the rest configuration on a given mesh.

Encoder-cycle inconsistency. For each candidate we encode the rest configuration to $\mathbf{z}_0$, then run $T = 10$ autoregressive steps with action $[1, 0, 0, 0]$ (uniaxial stretch). At each step t the dynamics-predicted latent $\mathbf{z}_t$ is decoded to a displacement field $\mathbf{u}_t$, subsampled to the same $S = 10{,}240$ FPS-coarsened query points used at encoding time, and re-encoded as a 3-frame window $(\mathbf{u}_{t-2}, \mathbf{u}_{t-1}, \mathbf{u}_t)$ to produce $\hat{\mathbf{z}}_t$ (the leading two frames are zero-padded for $t < 3$). The per-frame metric is $c_t = \|\mathbf{z}_t - \hat{\mathbf{z}}_t\| / \|\mathbf{z}_t\|$ (L^2 norm of the flattened latent); the per-candidate scalar in Fig. 5(C) is the mean over $T = 10$ frames.

Confidence head architecture. The head input is a 1,050-dimensional concatenation of: tokenizer mean-pooled latent ($\mathrm{mean_pool}(\mathbf{z}_0)$, 1,024 dimensions); tokenizer reconstruction error (1 dimension); per-frame consistency vector $(c_1, \ldots, c_{10})$ (10 dimensions); 15-dimensional graph statistics (node and beam counts, spatial spread, beam lengths, radii, connectivity, matching the features used in Fig. 8). The head is a 3-layer MLP: $\mathrm{LayerNorm} \to \mathrm{Linear}(1050, 256) \to \mathrm{GELU} \to \mathrm{Dropout}(0.2) \to \mathrm{Linear}(256, 128) \to \mathrm{GELU} \to \mathrm{Dropout}(0.2) \to \mathrm{Linear}(128, 20)$. The 20 outputs are softmax-normalised over a uniform partition of $[-0.05, 1.0]$ in ρ. The predicted scalar $\hat{\rho}$ in Fig. 5(D) is the expectation of bin centers under the softmax.

Training procedure. Optimization uses Adam (learning rate 10^{-3}, weight decay 10^{-4}), batch size 256, up to 200 epochs with early stopping (patience 30) on validation cross-entropy, on a single H100. We train three input configurations for ablation: free signals only (reconstruction error + 10-frame consistency, 11 input dimensions), latent only (mean-pooled $\mathbf{z}_0$, 1,024 input dimensions), and the full hybrid (1,050 input dimensions). The full configuration is used for Fig. 5(D).

F Additional Results

Per-frame rollout curves. Figure 9 shows the per-frame normalized work error during 30-step autoregressive rollout across 55 architected lattices, and Figure 10 shows example rollout work vs. time curves compared to ground truth.

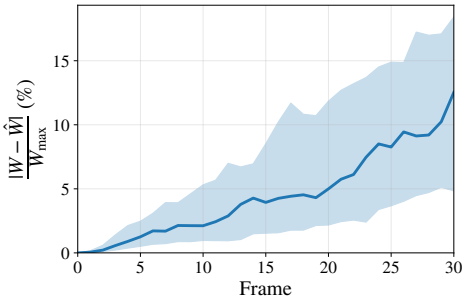

Figure 9: **Per-frame normalized work error on MicroPlate.** Per-frame normalized work error $|W - \hat{W}|/W_{\max}$ across 55 architected lattices during 30-step autoregressive rollout. Solid line: median. Shaded band: interquartile range.

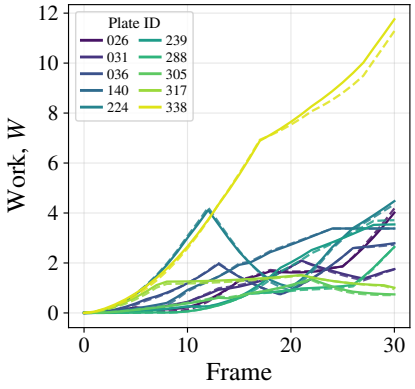

Figure 10: **Work vs. time on ten MicroPlate shapes.** Solid: FEM ground truth. Dashed: prediction.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction claim (i) accurate stress prediction on architected plates, (ii) the MicroPlate benchmark, covering microstructure modeled both explicitly through 3D architected lattices and implicitly through internal degrees of freedom on a visco-hyperelastic plate, (iii) surrogate-guided design search with FEM validation, and (iv) a learned confidence head for surrogate reliability; each is demonstrated in Sections 4.1–4.4.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Section 5.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not include theoretical results.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Datasets, FEM solver details, model architectures, training hyperparameters, and evaluation protocols are documented in the Methods section and Appendices A–E.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code is released at `https://github.com/HaiqianYang-MechE/leia`.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Data splits are described in Appendix B; full hyperparameters and optimizer settings for LEIA and all baselines are in Appendix C.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All main-table metrics are medians across test trajectories (100 trajectories for the visco-hyperelastic plate; one held-out trajectory on each of the 55 training plates and the 8 held-out plates for the lattice regime), computed per trajectory. Figure 4(d) reports the standard deviation of per-candidate evaluation time, and the per-frame rollout curves in Appendix F show the median and interquartile range across the 55 lattices. We do not report variance across training seeds, because each lattice-regime model requires several days on 8 H100 GPUs to train.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: GPU type and count for each model are reported in Appendix C; FEM data generation compute is described in Appendix A.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The research conforms with the Code of Ethics.

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section 1 discusses the positive impact of accelerating the design of architected materials for applications such as orthopedic implants, energy-absorbing structures, and lightweight components. The work concerns surrogate simulation of mechanical response for engineering design, and we do not foresee direct negative societal impacts beyond those of faster engineering simulation in general.

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All relevant sources are cited.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: MicroPlate is documented in Appendix B (graph representation, meshing, FEM protocol, splits, and statistics), and the released repository includes the mesh-generation and FEM data-generation code, dataset manifests, and documentation for the trained models.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not part of the core methodology. They were used only for code assistance and writing/editing support, with all code and text reviewed by the authors.