npj | computational materials

Article

Published in partnership with the Shanghai Institute of Ceramics of the Chinese Academy of Sciences

https://doi.org/10.1038/s41524-026-02167-x

Leveraging neural network interatomic potentials for a foundation model of chemistry

Check for updates

So Yeon Kim[1,3], Yang Jeong Park[1,4] ✉ & Ju Li[1,2] ✉

Large-scale foundation models, including universal neural network interatomic potentials (NIPs) in computational materials science, have demonstrated significant progress. However, despite their success in accelerating atomistic simulations, NIPs still face challenges in modeling certain property classes. Machine learning (ML) offers alternatives for structure-to-property mapping but different ML approaches present distinct trade-offs: feature-based methods often lack generalizability, while deep neural networks require significant data and computational power. To address these trade-offs, we introduce HackNIP, a two-stage pipeline that leverages pretrained universal NIPs. This method first extracts embeddings from universal NIP foundation models and then uses these embeddings as fixed-length feature vectors to train downstream learners for structure-to-property predictions. This study investigates whether such a hybridization approach, by "hacking" the NIP, can outperform end-to-end deep neural networks, determines the dataset-size regime in which this approach surpasses direct fine-tuning of the NIP, and identifies which NIP embedding depths yield the most informative features. Together, this work demonstrates a hybridization strategy for overcoming trade-offs in structure-to-property predction in materials science, thereby lowering barriers to practical materials-design workflows.

Large-scale foundation models developed through self-supervised learning have demonstrated great potential across numerous applications, ranging from text generation and language comprehension to image generation and interpretation[1–6]. These models follow "a scaling law," where performance improves continuously with increased training compute. In computational materials science, a prominent example is machine learning interatomic potentials (MLIPs), which learn to approximate high-fidelity potential energy surfaces from quantum-mechanical data, serving as fast, accurate surrogates for expensive first-principles calculations. Particularly, MLIP foundation models based on deep neural networks (NNs)—known as neural network interatomic potentials (NIPs)—like GNoME[7] and MatterSim[8] have confirmed the scaling law, and being trained on large-scale density functional theory (DFT) datasets such as MPtrj[9] and OMat24[10], across a wide swath of the periodic table, these universal NIPs[11–13] can replicate DFT results at high accuracy at just a fraction of computational costs previously required, assuming we treat the costs of constructing DFT datasets as sunk.

The performance gains in large-scale universal NIP foundation models[7–9,14–17] have expanded the spatiotemporal limits of atomistic simulations achievable at DFT-level accuracy. These gains build on a long evolution of MLIP architectures, from landmark contributions such as Behler–Parrinello neural networks (BPNN) and Gaussian approximation potentials (GAP)[18,19] to modern equivariant GNN architectures that explicitly embed physical symmetries into the model[14,15,20], enabling more data-efficient learning that generalizes well to unseen structures. Nevertheless, many practical challenges remain in downstream structure-to-property predictions. First, because NIPs bypass explicit computation of electron density and directly map to energies, they often struggle to predict electronic properties. Second, predicting macroscopic properties still tend to require coupling to higher-scale models (e.g., coarse-grained potentials) or generating enormous MD ensembles. Third, bridging the gap between DFT predictions and experimental measurements for properties that are not directly accessible via atomistic simulations remains a key development goal.

[1]Department of Nuclear Science and Engineering, Massachusetts Institute of Technology, Cambridge, MA, USA. [2]Department of Materials Science and Engineering, Massachusetts Institute of Technology, Cambridge, MA, USA. [3]Present address: Department of Mechanical Engineering, University of Wisconsin-Madison, Madison, WI, USA. [4]Present address: Department of Materials Science and Engineering, Ulsan National Institute of Science and Technology, Ulsan, South Korea. ✉e-mail: parkyj@unist.ac.kr; liju@mit.edu

a

Aspect	Feature-based ML models	Deep NNs
Input	Hand-crafted descriptors	**Raw structural data**
Architecture	**Shallow**	Deep
Data efficiency	**Good for small datasets**	Requires large datasets
Training Cost	**Low**	High / Moderate
Structural Awareness	Limited by chosen features	**High (learned end-to-end)**
Generalizability	Limited by feature bias	**High if well trained**
Interpretability	**Moderate-high**	Low

b

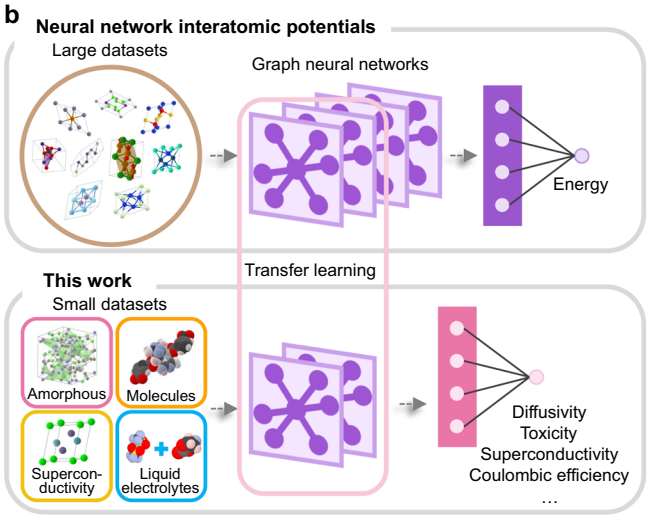


Fig. 1 | Overview of machine learning strategies for structure-to-property predictions in materials science. a Comparison of conventional feature-based machine learning (ML) models and deep neural networks (NNs). The comparative advantages of each model relative to the other and the architectural characteristics that provide these advantages are highlighted in bold. **b** Transfer learning using pretrained neural network interatomic potentials (NIPs) in this work. Traditional NIPs (top) are trained on large datasets comprising diverse atomic structures to learn interatomic interactions via deep NN. This work (bottom) applies transfer learning to small datasets spanning various material classes—including amorphous solids, molecular systems, superconductors, and liquid electrolytes—by leveraging pretrained representations that enable accurate modeling in data-scarce regimes.

Machine learning (ML) presents a viable alternative for high-throughput mapping of structure to properties in cases where atomistic simulations with NIPs encounter limitations. Figure 1a summarizes the architectural characteristics and trade-offs between conventional feature-based ML and deep NNs. Conventional feature-based ML models use fixed, hand-crafted descriptors and shallow architectures, making them efficient on small datasets and interpretable (though still far less so than atomistic simulations), but limited in generalizability due to the expense of structural detail and inherent feature biases. In contrast, deep NNs—like graph neural networks (GNNs)—can learn directly from atomic structures, capturing complex patterns with broader generalizability, but they demand more data and computational resources and are often difficult to interpret.

Parallel efforts for molecular and crystalline materials have also sought general-purpose descriptors via contrastive and self-supervised pretraining on molecular graphs and crystal structures[21–26], though these approaches do not readily generalize across the full diversity of materials. In this regard, transfer learning offers a promising strategy: extracting features from pretrained universal NIPs and feeding them into lightweight downstream learners[27]. This hybridization approach can potentially overcome those trade-offs, given the fact that they can use raw structural data as input and employ fixed-embedding learners for property mapping (Fig. 1a). Such an approach raises three key questions: First, can a downstream learner with NIP-derived embeddings outperform end-to-end deep NNs for property prediction, by harnessing broader knowledge of the base NIP through repurposing internal NIP embeddings without end-to-end fine-tuning—namely, "hacking" the universal NIP? Second, at what dataset size range does this transfer-learning pipeline surpass direct fine-tuning of the same NIP? Third, which embedding depths of an NIP yield the most informative features for downstream tasks?

To explore these questions, we introduce HackNIP, a two-stage pipeline (Fig. 1b): (1) feature extraction from NIP foundation models to convert atomic structures into fixed-length vectors; and (2) structure-to-property prediction using downstream learners trained on the extracted feature vectors paired with target properties. The term "HackNIP" reflects the idea of repurposing ("hacking") pretrained NIPs by extracting their internal embeddings and reusing them for downstream tasks beyond their original energy-prediction objective. Rather than modifying or fine-tuning the backbone, we leverage its learned representations as fixed descriptors for new property-prediction problems. First, we benchmark its performance on the standard Matbench[28] dataset. Second, we assess data efficiency by evaluating model performance across varying dataset sizes, and compare it to that of a direct fine-tuning approach. Third, we apply the method to diverse downstream tasks—encompassing dynamic properties from ab initio calculations, experimental datasets involving one or more materials, and molecular properties—to demonstrate generalizability. Finally, we discuss how the choice of embedding depth for feature extraction, together with NIP characteristics, influences performance, providing guidance for future development. Overall, this work demonstrates the potential of hybridization approaches to overcome trade-offs in ML for materials science and to democratize high-performing ML capabilities.

Results

Benchmarking model performance on Matbench tasks

We first benchmarked HackNIP on the Matbench test suite, an automated leaderboard for ML algorithms predicting a diverse range of solid-state properties. Matbench comprises 13 tasks over 10 datasets, varying in target property, sample size, and data-generation method. From these, we selected all regression tasks with available atomic-structure inputs. The target properties of these tasks are: exfoliation energy ($E_{\text{exfoliation}}$); frequency of the last phonon peak in the phonon density of states (ν); refractive index (n); logarithmic shear and bulk moduli (G and K, respectively); perovskite formation energy ($E_{\text{f,perovskite cell}}$); band gap ($E_g$); and general formation energy (E_f). Sample counts ranged from 636 (exfoliation energy) to 132,752 (formation energy). All structures and target values in the datasets were compiled by DFT calculations. We used the five predefined train/test splits provided by the Matbench Python package. The optimized hyperparameters for each task are summarized in Suppl. Table S1. Various combinations of NIP foundation models (ORB[16], Equiformer[15], and MACE[14]) and downstream learners (MODNet[29], XGBoost[30], and MLP[31]) were evaluated using a fixed set of hyperparameters (Fig. S1), and the best-performing pair under this setting (ORB + MODNet) was selected for demonstration (see Methods for more details).

Figure 2a shows the relative mean absolute errors (MAEs) of HackNIP and selected baselines compared to a dummy model, which always predicts the training-set mean. The Matbench leaderboard (as of May 2026) comprises 28 community-contributed models, each evaluated on a subset of tasks; here, we retained only those reporting results across all eight regression tasks. These baselines include graph neural networks (e.g., CGCNN and ALIGNN) as well as feature-based ML models (e.g., AMMExpress and MODNet). We denote the MAE of model i on fold j as MAE_i^j, and its average across folds as $\text{MAE}_i^{\text{avg}} = \frac{1}{5}\sum_{j=1}^{5}\text{MAE}_i^j$. The relative MAE is then defined as $\text{MAE}_i^{\text{avg}}$ divided by $\text{MAE}_{\text{dummy}}^{\text{avg}}$ where $\text{MAE}_{\text{dummy}}^{\text{avg}}$ is the average MAE of the dummy model on five folds. HackNIP achieves the state-of-the-art predictive performance with the lowest relative MAE on four tasks and remains

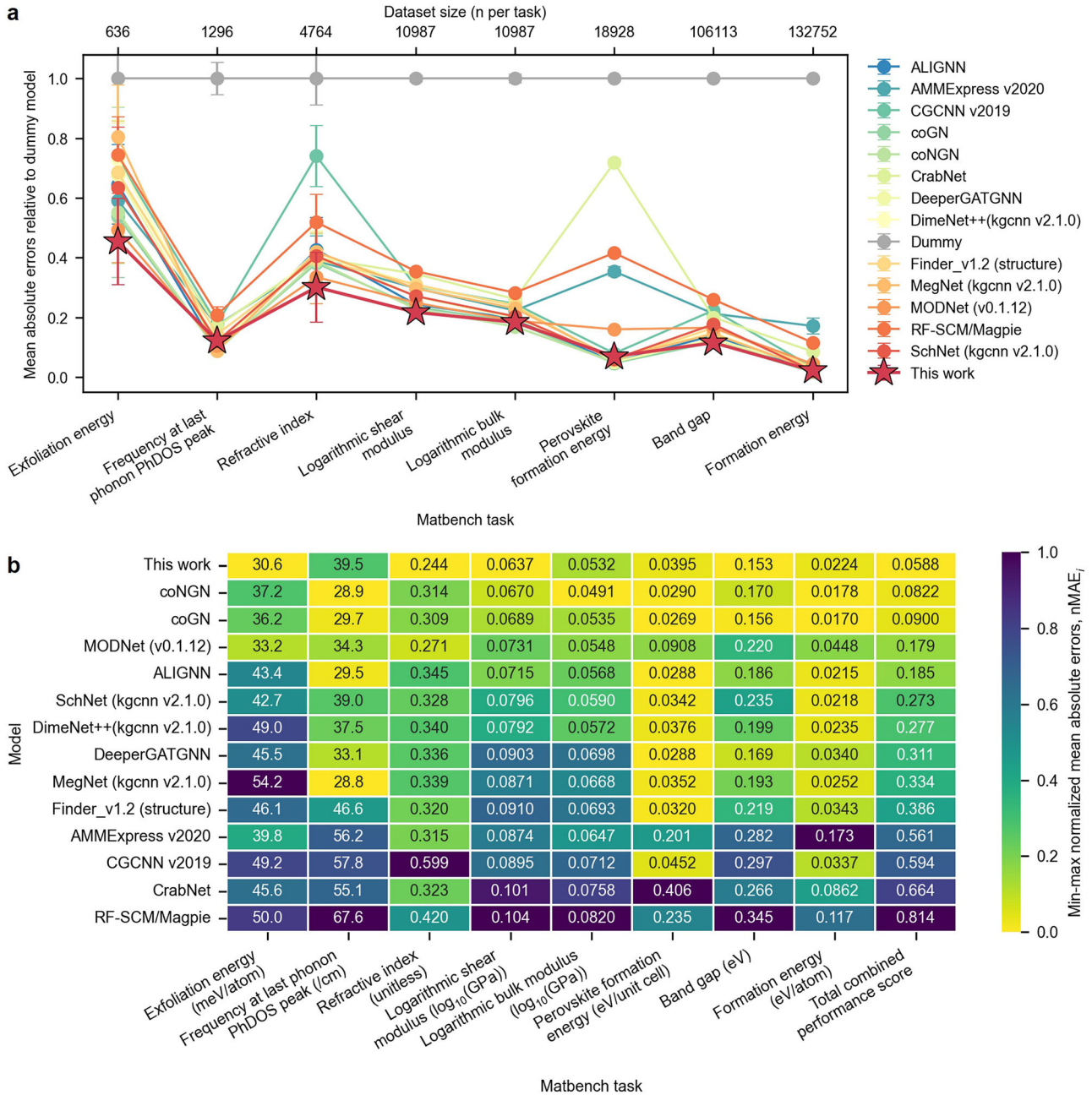


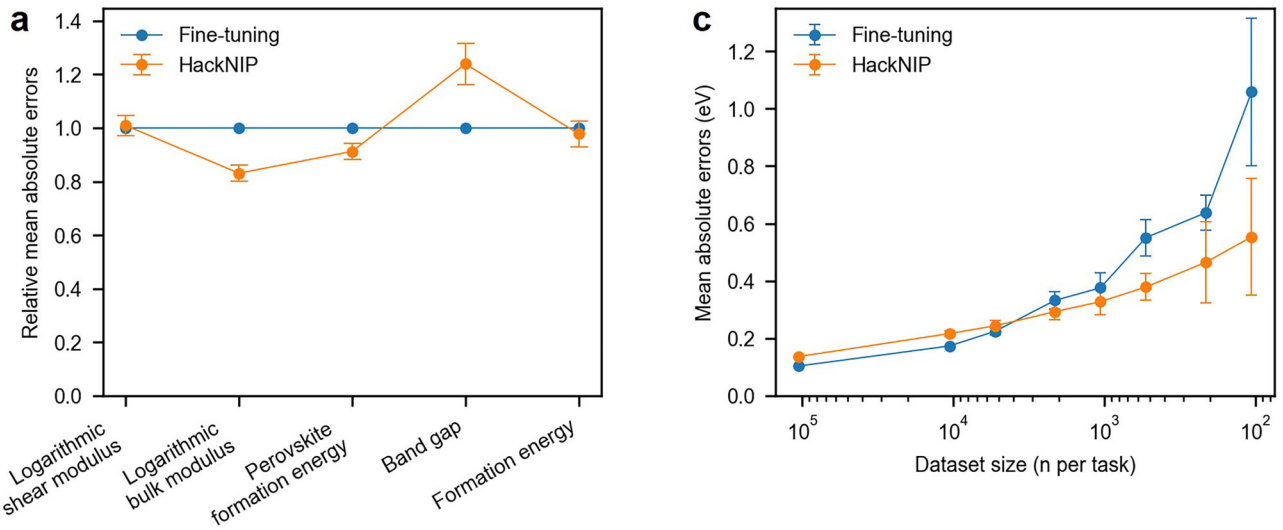
Fig. 2 | Benchmarking model performance across Matbench tasks. a Mean absolute error relative to a dummy model that predicts the mean of the dataset. Dataset size for each task is indicated above the plot. The performance of HackNIP (red stars) is compared to those of baseline and state-of-the-art models. Error bars represent standard deviations over five-fold experiments. Lower mean absolute error indicates better performance. **b** Raw mean absolute error of each model. Cells are color-coded by the scaled MAE ($nMAE_i$), ranging from 0 (best) to 1 (worst), calculated using min max normalization within each task.

competitive elsewhere, delineating an approximate Pareto-optimal front across the eight Matbench tasks with varying dataset sizes.

Figure. 2b shows the absolute MAEs, with each cell color-coded according to the mix-man normalized MAE of each model i ($nMAE_i$) defined as

$$\mathrm{nMAE}_i = \frac{\mathrm{MAE}_i^{\mathrm{avg}} - \min_k \mathrm{MAE}_k^{\mathrm{avg}}}{\max_k \mathrm{MAE}_k^{\mathrm{avg}} - \min_k \mathrm{MAE}_k^{\mathrm{avg}}}, \quad (1)$$

where the extrema run over all non-dummy models, k. Cells are colored by $nMAE_i$ between 0 (best) and 1 (worst). HackNIP attains $nMAE_i = 0$ for four tasks and remains below 0.11 for three, indicating it remains competitive with end-to-end deep NNs. Although HackNIP underperforms some recent GNNs on the last-phonon-peak frequency and perovskite formation energy tasks, it still outperforms all feature-based ML models and earlier GNN architectures. Notably, it achieves a high coefficient of determination $R^2 = 0.969 \pm 0.010$ for the phonon-peak frequency (see parity plot in Fig. S2) and an MAE of ~40 meV/unit cell for perovskite formation energy—below the so-called chemical accuracy (i.e., 1 kcal/mol or 43 meV/atom)—indicating it predicts DFT reference values more accurately than DFT itself predicts experimental formation energies. Conversely, for refractive index, HackNIP attains the lowest MAE but a poor R^2 (0.337 ± 0.268), likely due to data sparsity at high index values; logarithmic scaling might mitigate this, but no additional preprocessing was applied to preserve comparability. Averaging $nMAE_i$ over the eight tasks yields a combined performance score, for which HackNIP ranks first among all leaderboard models. The performance of MODNet alone without the features extracted from NIP foundation models



b

Matbench task	Fine-tuning MAE	HackNIP MAE
Logarithmic shear modulus ($\log_{10}$ GPa)	0.0631	0.0637 ± 0.0024
Logarithmic bulk modulus ($\log_{10}$ GPa)	0.0640	0.0532 ± 0.0019
Perovskite formation energy (eV/unit cell)	0.0548	0.0500 ± 0.0017
Band gap (eV)	0.1740	0.2156 ± 0.0133
Formation energy (eV/atom)	0.0408	0.0410 ± 0.0019

Fig. 3 | Comparison of fine-tuning approach and HackNIP across varying dataset sizes. a Mean absolute errors of HackNIP relative to fine-tuned NIP across five Matbench tasks when using ~10^4-sample subsets: shear and bulk moduli (10,987 samples), perovskite and general formation energies (10,000 samples), and band gap (10,611 samples). **b** Summary of raw mean absolute errors. **c** Raw mean absolute errors of fine-tuned NIP and HackNIP as a function of dataset fraction (0.1%, 1%, 10%, and 100%) for the band gap prediction task. Error bars represent standard deviations over five independent subsets.

is also shown in Fig. 2; the HackNIP pipeline consisting of ORB and MODNet outperformed it on all tasks except phonon-peak frequency prediction, achieving a three-fold improvement in the combined performance score.

It should be noted that ORB is not strictly E(3)-equivariant, and orientation-dependent effects can in principle appear. We thus also examined the sensitivity of HackNIP to structural orientation. Specifically, for the Matbench band gap task, the test structures in each fold were randomly rotated about randomly chosen axes prior to featurization. The resulting MAE with respect to the true band gaps was 0.152 ± 0.014 eV (Fig. S3a), which is similar to the MAE obtained without rotation (0.153 ± 0.016 eV; Fig. S3b). Furthermore, predictions from rotated and non-rotated structures are strongly correlated (R^2 = 0.98; Fig. S3c), with an MAE of 0.102 ± 0.014 eV, indicating that orientation-dependent effects have only a minor impact on the predictive performance of HackNIP in this setting.

Data efficiency

To examine data efficiency, we conducted HackNIP training and ORB fine-tuning on the five Matbench tasks with more than 10^4 samples—shear and bulk moduli, perovskite formation energy, band gap, and general formation energy. For property-specific fine-tuning, we attached a single-output MLP[31] head (see Methods for more details) and unfroze most of the ORB backbone while keeping the initial encoder and final decoder frozen. This configuration allows the internal representations to adapt to each target property while preserving pretrained low-level geometric representations and avoiding unnecessary updates to the force-field decoder.

Figure 3a compares the relative MAE of HackNIP against fine-tuned ORB when each is trained on approximately ~10^4 samples—10,987 samples for moduli (full dataset); 10,611 for band gap (10% subset); 10,000 for formation energies (53% of the perovskite formation energy dataset and 7.5% of the general formation energy dataset). The raw (non-normalized) MAE, are summarized in Fig. 3b. For each subset, the desired fraction of the full dataset was drawn by random sampling. The ORB embedding depths used for feature extraction are identical to those used in Fig. 2, although MODNet hyperparameters were re-optimized for the smaller subsets. The performance of HackNIP matches or exceeds that of fine-tuned ORB in four of the five tasks—band gap being the sole exception—suggesting that it may serve as a competitive universal predictor up to dataset sizes of ~ 10^4 without costly end-to-end fine-tuning for each property.

Interestingly, while HackNIP and fine-tuned ORB performed similarly when predicting shear modulus and formation energies, they differed substantially in predicting bulk modulus and band gap. Specifically, HackNIP outperformed direct fine-tuning by 17% on bulk modulus, even though this was the task where HackNIP exhibited less competitive performance relative to the leaderboard models (Fig. 2b), showing the lowest relative MAE across the five tasks with > 10^4 datapoints. In contrast, HackNIP underperformed compared to fine-tuned ORB for band gap despite HackNIP topping the leaderboard (Fig. 2b). These differences may be attributed to specific characteristics of the ORB architecture.

We then further examined the band gap task to identify the dataset size at which HackNIP begins to outperform fine-tuned ORB. The hyperparameters were fixed for both approaches, and the training configurations were set such that the computational budget for the HackNIP approach remains approximately sixfold smaller than that of the fine-tuning approach. The training configurations are summarized in Suppl. Table S2. The dataset was

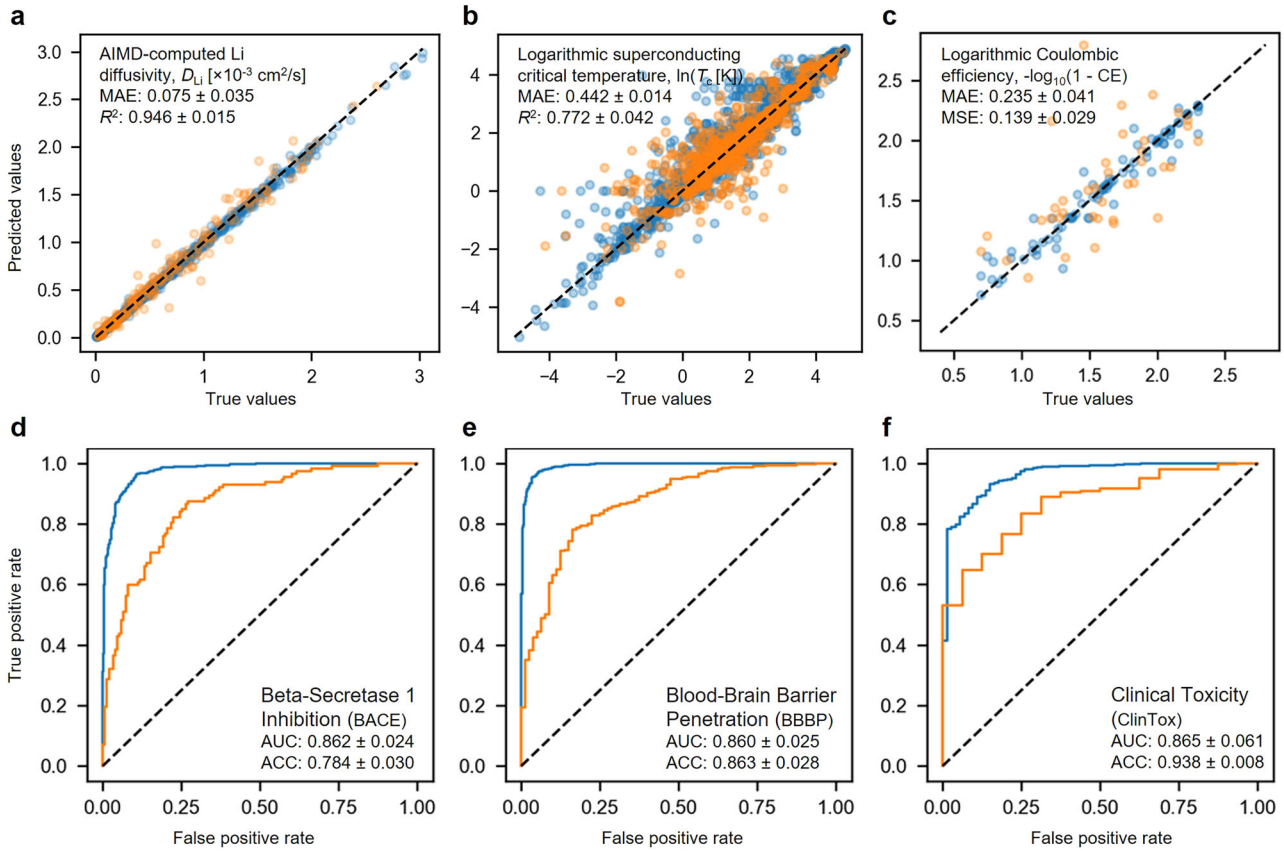


Fig. 4 | Model performance across diverse downstream tasks. Parity plots for regression tasks on: (**a**) Li diffusivity calculated by ab initio molecular dynamics simulation; (**b**) superconducting critical temperatures; and (**c**) Coulombic efficiency. Mean absolute errors (MAEs) and either the coefficients of determination (R^2) or mean squared errors (MSEs)–matching the metrics used in previous studies–are noted. Receiver operating characteristic curves for classification tasks on: (**d**) beta-secretase 1 inhibition (BACE); (**e**) blood-brain-barrier penetration (BBBP); and (**f**) clinical toxicity (ClinTox). Area under the curve (AUC) and accuracy (ACC) values are noted. Blue: train; Orange: test; Dashed black line: identity line.

randomly split into training, validation, and test sets in an 8:1:1 ratio. Fig. 3c plots MAE versus the total number of samples used for training and testing, with the x-axis shown on a logarithmic scale. For dataset sizes below 10^4, we performed fine-tuning on five different randomly sampled subsets to ensure statistical robustness. With 100% of the data, fine-tuned ORB outperforms, achieving an MAE of ~0.10 eV compared with 0.137 eV for HackNIP. As the dataset size decreases, the MAE of fine-tuned ORB grows roughly as a power law, whereas HackNIP's MAE increases approximately log-linearly. Consequently, the two curves cross between 10^3 and 10^4 samples, with HackNIP increasingly outperforming fine-tuned ORB as the dataset size decreases.

Generalizability

All target properties in the aforementioned Matbench benchmarking tasks are zero-temperature DFT properties. In contrast, real-world materials research often requires ML models for finite-temperature, dynamic properties—whose datasets are far more time-consuming to assemble even with high-throughput workflows—as well as those for properties that cannot be computed directly from first principles or even those for system-level performance metrics arising from interactions among multiple materials. To demonstrate the broad applicability of HackNIP, we evaluated it on a diverse collection of literature datasets: (1) Li diffusivity in amorphous materials (regression), computed via ab initio molecular dynamics (AIMD)[32]; (2) superconducting critical temperature (regression), measured experimentally[33]; (3) Coulombic efficiency of Li-ion batteries (regression), involving combinations of electrolyte precursors[34]; and (4–6) molecular-property classification tasks such as beta-secretase 1 inhibition (BACE)[35], blood–brain barrier penetration (BBBP)[36], and clinical toxicity (ClinTox)[37] from the MoleculeNet database[38].

Figure 4 presents parity plots for the regression tasks and receiver operating characteristic (ROC) curves for the classification tasks. For each regression dataset, we report MAE alongside either R^2 or mean squared errors (MSEs), matching the metrics used in previous studies in the literature. All three regression targets were trained on logarithmically scaled outputs, while plots and metric calculations use whichever scale (linear, natural log, or base-10 log) the relevant literature employed, to facilitate direct comparison. For the Li diffusivity and superconducting critical temperature tasks, we performed five-fold experiments using an 80/20 train/test split created with a Matbench seed of 18012019. For the Coulombic efficiency task, we conducted three independent evaluations using randomly generated 70/30 train/test splits with non-overlapping test sets, due to the limited dataset size and to maintain consistency with prior studies. The optimized hyperparameters are summarized in Suppl. Table S3. We discuss the predictive performance for each task below.

Figure 4a displays a parity plot of the model performance on AIMD-computed Li diffusivity. The database includes, but is not limited to, Li-containing amorphous solids, and the diffusivities of constituent elements were computed at high temperatures ranging from 1000 K to 5000 K. Among these, we used every Li-containing data point in the database[32] and included temperature as an additional feature alongside ORB embeddings. For direct comparison with the literature—which likely excluded the 5000 K data, perhaps because many compounds are no longer in the condensed phase at that temperature (though this is not stated explicitly), we drew the

parity plot on a linear scale and estimated predictive performance in this representation, yielding MAE = 0.075 and $R^2 = 0.946$. In comparison, Ref. 32 used human-engineered descriptors and reported MAE = 0.081 and $R^2 = 0.956$ with a random forest regressor, and MAE = 0.083 and $R^2 = 0.960$ with an XGBoost regressor, both using 85/15 train/test splits (vs. 80/20 splits for HackNIP). It is notable that HackNIP can achieve performance comparable to these state-of-the-art models for finite-temperature, dynamic properties as well, without hand-engineered features—which might require substantial prior knowledge—and thus without human bias.

Figure 4b shows HackNIP's performance on predicting superconducting critical temperatures (T_c), which must be determined experimentally rather than computed from first principles. We used the 3DSC database[33], which augments the SuperCon composition entries with approximate three-dimensional crystal structures drawn from the Materials Project, wherever possible. This merged dataset comprises more than 5000 entries. After applying a logarithm to the critical-temperature values, we removed any entries with non-finite targets, yielding a final training set of approximately 4000 entries. The parity plot is drawn in natural log scale following Ref. 39. In the low-T_c regime, deviations from the parity line were relatively large, resulting in MAE = 0.442 and $R^2 = 0.772$, while at higher levels the predictions align closely with the identity line, which is the region of greatest practical interest. We note that Ref. 39 reported a higher $R^2 = 0.88$ with composition-based features from Matminer[40] and a random-forest regressor, while direct comparison to such ML studies is challenging, since most employ larger SuperCon subsets (12,000+) via composition-only featurization, adopt different target ranges ($\ln(T_c)$: 2.3–5 vs. –4–4 here), and employ different train/test splits (85/15 in Ref. 39 vs. our 80/20).

Figure 4c illustrates the results for Coulombic efficiency of Li-ion batteries with organic liquid electrolytes. We used the experimental dataset[34], which contains 124 entries. Since each electrolyte formulation can involve multiple precursor species—solvents, solutes, additives, etc.—mixed in different proportions, we first convert them to molar concentrations. Then, the embedding of each compound was computed, and a weighted sum of these embeddings was calculated using the molar concentrations as coefficients to obtain the final embedding. HackNIP achieved superior predictive performance, with an MAE of 0.235 and an MSE of 0.139—about a 60% reduction compared to the minimum MSE of 0.333 attained by linear, random-forest, boosting, or bagging models using human-engineered features.

For classification tasks on molecular data, we plot true-positive rate versus false-positive rate (ROC curves)—a threshold-agnostic metric that is robust to class imbalance—and report area under the curve (AUC) values in addition to accuracy (ACC) values. Figure. 4d–f show the classification performance for BACE, BBBP, and ClinTox, respectively. Those MoleculeNet benchmark datasets are highly relevant to real-world drug discovery. For example, BACE1 is a key enzyme involved in producing β-amyloid peptides, which aggregate in the brains of Alzheimer's patients. Inhibiting BACE1 has long been considered a promising strategy for slowing Alzheimer's progression. Whether a small molecule can penetrate the blood–brain barrier (BBB) is critical for developing central nervous system drugs (e.g., for Parkinson's, Alzheimer's, depression). The ClinTox dataset collects clinical trial outcomes (approved vs. withdrawn/toxic) for thousands of drugs and drug-like molecules.

Since the molecular data are provided as Simplified Molecular Input Line Entry System (SMILES) strings, we first generated three-dimensional structures and performed geometry optimization before feature extraction. As with regression, we extracted features from ORB, performed layer-search with fixed MODNet hyperparameters, then optimized those hyperparameters and evaluated via five-fold experiments. In all three cases, AUC values are in the mid-80% range, indicating strong classification performance. This result is comparable to pretrained models such as MolCLR[22] and GEM[23], which have been specifically trained to extract useful representations for molecules, whereas ORB was trained to predict energy using a supervised-learning approach.

NIP embedding depth for feature extraction and model performance

Next, we examine the effect of NIP embedding depths used for feature extraction on HackNIP performance. The ORB (orb-v2 in this work) backbone comprises 15 layers, each of which can be used to extract embeddings. Obtaining embeddings from the last layer requires propagating raw coordinates and species labels through the shared encoder and all 15 backbone layers, which substantially increases featurization cost—especially when processing on the order of $> 10^4$ samples. As summarized in Suppl. Table S1, most Matbench regression tasks attain their best MAE using early-layer embeddings ($1 \le L \le 6$), with band gap and exfoliation energy as exceptions (optimal at $L = 8$ and $L = 11$, respectively). Note that the best-performing embedding depth for the exfoliation energy task in the initial search without hyperparameter optimization was $L = 6$. After optimization, the best MAE achieved with $L = 6$ was 29.3 meV/atom, which still outperforms the other models, as shown in Fig. 2b. Although the difference is modest compared to the gain over other models, this result demonstrates that the optimal embedding depth can change depending on the hyperparameter optimization. This should be taken into account, particularly when further analyzing the data, as illustrated in the Discussion section. Likewise, Li diffusivity, superconducting critical temperature, and Coulombic efficiency reach minimal MAE at $L = 2$, 3, 1, respectively (Suppl. Table S3). By contrast, molecular-property classification tasks (BACE, BBBP, ClinTox) achieve peak AUC at deeper layers ($L = 8$, 13, 11), as shown in Suppl. Table S3.

First, we investigated how HackNIP's MAE varies with the ORB embedding depth (L) used for feature extraction on Matbench tasks (Fig. 5a), employing MODNet with a fixed hyperparameter without further optimization. To compare across tasks, we scaled each MAE between its task-specific minimum and maximum, defining the mix-max normalized MAE for task t using embeddings from layer L ($\mathrm{nMAE}_{t,L}$) as

$$\mathrm{nMAE}_{t,L} = \frac{\mathrm{MAE}_{t,L} - \min\limits_{L'} \mathrm{MAE}_{t,L'}}{\max\limits_{L'} \mathrm{MAE}_{t,L'} - \min\limits_{L'} \mathrm{MAE}_{t,L'}}, \tag{2}$$

where $\mathrm{MAE}_{t,L}$ is the average MAE for task t at layer L over five-fold experiments. The minimum and the maximum are taken over all layers L' for the same task. Plots showing the raw (non-normalized) MAE ($\mathrm{MAE}_{t,L}$) are also provided in Fig. S4. In most cases, embeddings from the final layer yield the highest (i.e., worst) MAE, whereas optimal performance is achieved by using embeddings from the first half of the ORB network for feature extraction. The optimal layer shifts depending on the downstream ML algorithm as shown in Fig. S5 for two additional HackNIP combinations (ORB + XGBoost and ORB + MLP); however, both combinations still exhibit increasing MAE for later layers.

To visualize how the layer-wise embeddings evolve through the NIP network, we projected them into two dimensions (2D) using Uniform Manifold Approximation and Projection (UMAP)[41]. Figure 5b–i show the 2D UMAPs of embeddings from the best performing layer L^* for each Matbench task and the last layer ($L = 15$). In most cases, the final-layer embeddings collapse into a highly condensed manifold, indicating that the shared encoder and deep layers have compressed the representation into a low-dimensional subspace. In contrast, embeddings extracted from the layers that yielded the best predictive performance exhibit a much more diffuse UMAP layout; clusters corresponding to different regions of the property space are more widely separated, and outliers stand out more clearly. When starting from these less-compressed embeddings, a downstream model (i.e., MODNet) might better harness material characteristics that were de-emphasized (and thus suppressed) during construction of NIP foundation model, thereby leading to improved performance when using shallower layers of a given NIP architecture for feature extraction.

Lastly, to investigate possible reasons for the wider variation of best-performing layers observed for diverse downstream tasks, we examined the Jensen-Shannon Divergence (JSD) from a layer-contrast perspective. For

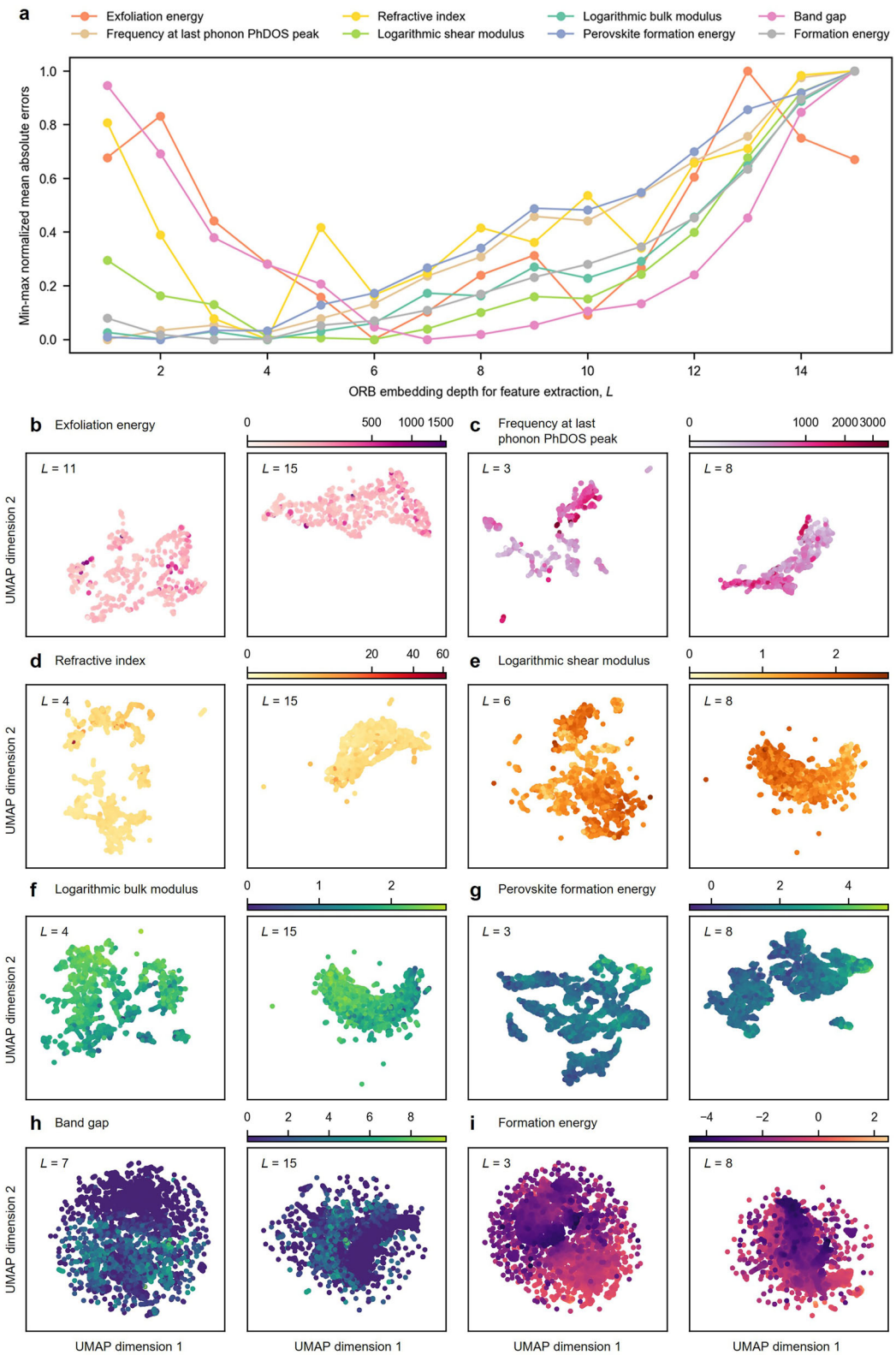


Fig. 5 | Embedding depth used for feature extraction and model performance. **a** HackNIP performance as a function of ORB embedding depth used for feature extraction. Mean absolute errors are min-max normalized across embedding depths for each Matbench task. Five-fold experiments were performed for each point and only the mean values are presented for clarity. **b**–**i** Uniform manifold approximation and projection (UMAP) of ORB embeddings for each Matbench task. ORB embeddings from the best-performing layer (L^*) and the final layer ($L = 15$) are reduced from 256 dimensions to two dimensions for each task, with color indicating the target value. Depending on the target-value distribution, the color scheme is either power-normalized or linearized.

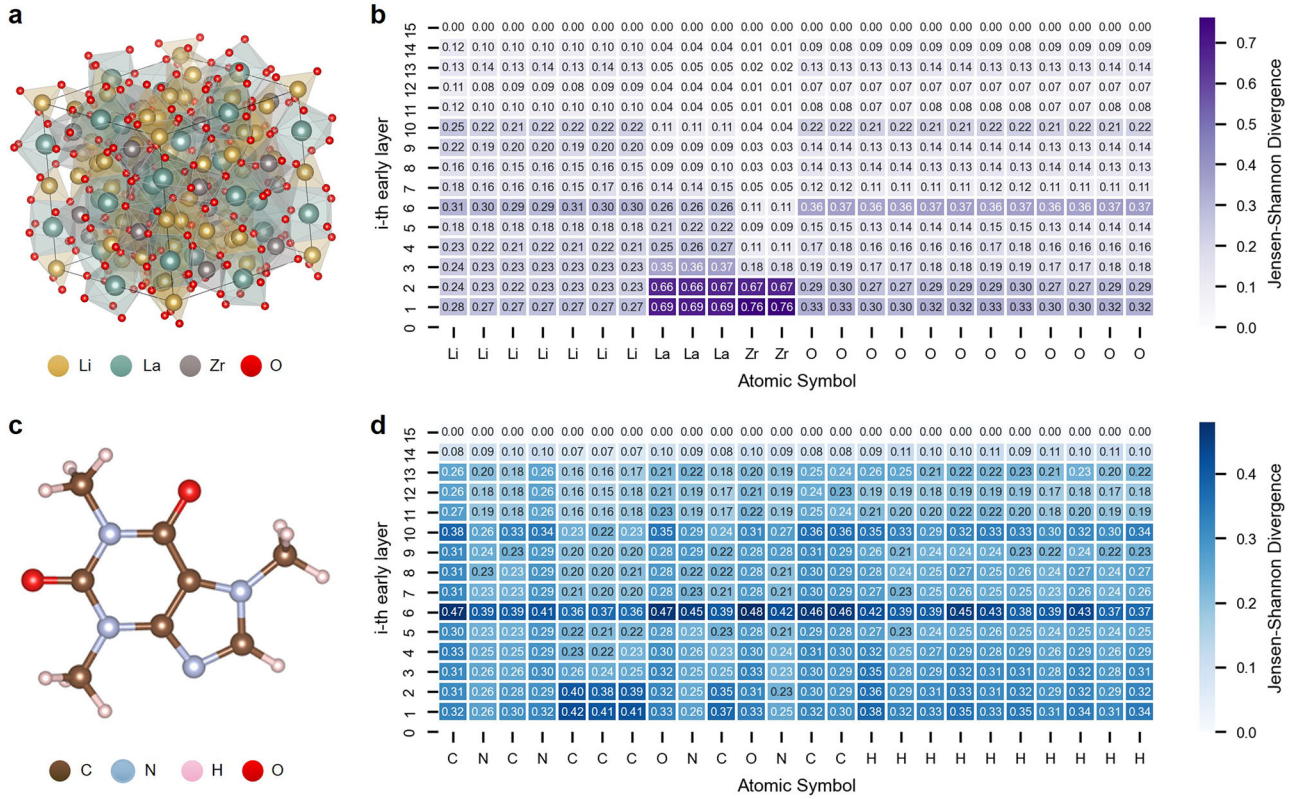


Fig. 6 | Jensen–Shannon divergence (JSD) distributions across embedding depths. a Atomic structure of the solid electrolyte, $Li_7La_3Zr_2O_{12}$ (LLZO). **b** Per-atom JSD values vs. embedding depth for LLZO, computed against the final-layer reference ($L_{ref} = 15$); therefore, values at $L = 15$ are zero by definition. Early transition metals exhibit very high divergence in layers 1–3 before rapidly diminishing. **c** Molecular structure of caffeine, $C_8H_{10}N_4O_2$. **d** Per-atom JSD values vs. embedding depth for caffeine, showing that divergence remains relatively high through deep layers. This contrast correlates with the finding that the best-performing layer for Li diffusivity prediction is shallow, whereas optimal layers for molecular-property prediction in caffeine lie much deeper.

each atom a, let $\mathbf{h}_a^{(L)} \in \mathbb{R}^d$ denote the ORB embedding at layer L. We convert $\mathbf{h}_a^{(L)}$ into a feature-channel probability vector $q_a^{(L)}$ by applying a channel-wise softmax, and define the per-atom JSD at depth L against the final-layer reference as

$$\mathrm{JSD}_a(L) = \mathrm{JSD}\left(q_a^{(L)} \parallel q_a^{(L_{\mathrm{ref}})}\right),\ L_{\mathrm{ref}} = 15. \tag{3}$$

The Jensen-Shannon divergence for two distributions P and Q is

$$\mathrm{JSD}(P||Q) = \frac{1}{2}D_{\mathrm{KL}}(P||M) + \frac{1}{2}D_{\mathrm{KL}}(Q||M), \tag{4}$$

where D_{KL} denotes the KL divergence. The JSD is always non-negative, symmetric, and equals 0 if and only if P and Q are identical. Therefore, $\mathrm{JSD}_a(15) = 0$ by definition.

Figure 6a, b display the atomic structure of the representative solid electrolyte, $Li_7La_3Zr_2O_{12}$ (LLZO). Figure 6c, d show the molecular structure of caffeine, $C_8H_{10}N_4O_2$, and the corresponding JSD distribution across embedding depths for each atom constituting the molecule. Under this definition, each per-atom/per-layer value quantifies how far that atom's layer-L representation is from its final-layer representation. Thus, high JSD indicates that representations are still evolving at that depth, whereas low JSD indicates that they are already close to the final representation. For both examples, the average JSD values across all atoms are high at embedding depths of 6 and 10. Meanwhile, in the LLZO example, JSD values for early transition metals—which shape the Li-conduction pathways—are very high in the first two to three layers, quickly diminishing afterward. This suggests that the conclusion for the final prediction is already made very early in the neural network. This behavior may be correlated with the finding that the best-performing layer for Li diffusivity prediction is $L = 2$, which is relatively shallow. On the other hand, in the caffeine example, the JSD values remain relatively high for most atoms into deeper layers (e.g., $L = 13$), compared to those of LLZO. This suggests that more message passing are needed to make a final conclusion. This behavior may be related to the observation that the best-performing layers for molecular properties are relatively deep. An in-depth analysis of the JSD distribution, combined with feature-importance analysis available for downstream learners, may offer deeper insights into structure–property relationships—for instance, by identifying the most critical atoms and their JSD distribution across embedding depths.

Discussion

Materials design is often a multi-objective optimization problem, where one must improve certain properties without degrading others[42]. Historically, designers have been driven to establish interdisciplinary knowledge[43] and handle a larger number of elements to strike the right balance among multiple properties[44], as exemplified by "high-entropy" materials (alloys, ceramics, catalysts, etc.)[45]. That strategy creates an effectively infinite design space, yet available data remain scarce because both computational and experimental investigations of these materials are often resource-intensive. Accordingly, ML models that are both universal and data-efficient are critical for transforming materials design practices, and lightweight yet robust approaches are essential to democratize state-of-the-art methods for academic/industrial groups with relatively limited computational resources. As demonstrated by its benchmarking performance and scaling capability, HackNIP's hybridization approach holds immense promise.

NIP architectures are continually evolving, incorporating more advanced, higher-order relational features. As NIPs become more

sophisticated, better capturing more diverse physics, HackNIP's performance may also improve. Meanwhile, such sophistication often incurs significant computational cost. One pragmatic alternative may be to extract features from multiple NIPs, each optimized for different orders of relational complexity, and then apply downstream, feature-based ML models. Analogous to emerging multi-agent strategies in the large-language model community, multi-NIP featurization strategies could offer a cost-effective alternative to building a single, computationally intensive NIP. Moreover, as in the hybridization approach adopted by HackNIP—where separate ML models divide roles (one model converts raw structures into physics-rich representations while the other maps those representations to properties)—combining several specialized deep NNs with downstream learners may further help balance predictive performance and computational efficiency.

Overall, we demonstrate that assigning distinct roles to ≥ 2 ML models can outperform a single, specialized end-to-end model on materials-property prediction tasks. Specifically, the two-stage hybrid framework, HackNIP (ORB + MODNet), matches or exceeds state-of-the-art performance on the Matbench leaderboard, often achieving accuracy within the inherent uncertainty of DFT data relative to experiment. It is lightweight enough to run on free-tier cloud computing services for datasets up to 10^5 samples, and data-efficient enough to outperform direct fine-tuning on datasets up to 10^4 samples—the size range within which most materials fall—while remaining broadly applicable across downstream tasks. These include molecular versus extended structures, computational versus experimental datasets, multi-phase phenomena, and both classification and regression problems. Our detailed analysis indicates that closer investigation into performance's dependence on embedding depths used for feature extraction has the potential to provide deeper insight into structure-property relationships, offering guidelines for future model development. The balanced robustness of this hybridization strategy could offer a fast yet reliable solution for many materials-science ML applications and is expected to significantly lower the resource-barrier to leveraging advanced ML capabilities in materials design.

Methods

Datasets

We leveraged a diverse suite of open-access materials and molecular benchmarks. The datasets included: Matbench[28] (comprising exfoliation energy; the frequency of the last phonon peak in the phonon density of states; refractive index; shear and bulk moduli; perovskite formation energy; band gap; and formation energy), MoleculeNet[38] (beta-secretase 1 inhibition[35], blood-brain barrier penetration[36], and clinical toxicity[37]), a curated compilation of superconducting critical temperatures for compounds with available structural data ($3DSC_{MP}$)[33], Li diffusivity in amorphous solids calculated via ab initio molecular dynamics simulation[32], and Coulombic efficiency data for electrolyte precursor combinations[34].

Neural network interatomic potentials and conventional machine learning models

We used NIPs to extract hierarchical, layer-wise embeddings from each atomic structure and then applied conventional ML models to map these embeddings to their corresponding properties. We evaluated five hybrid two-stage frameworks constructed from different combinations of NIPs and conventional ML models. The pretrained NIP foundation models tested were ORB ("orb-v2")[16], EquiformerV2 ("eqV2 M")[15], and MACE ("MACE-MP-0a medium")[14], and the downstream learners evaluated were MODNet[29], extreme gradient boosting (XGBoost[46]), and multilayer perceptron (MLP[31]). The tested combinations were ORB–MODNet, ORB–XGBoost, ORB–MLP, EquiformerV2–MODNet, and MACE–-MODNet. Under a chosen set of hyperparameters (detailed below), ORB–MODNet achieved the best overall performance on Matbench tasks and was selected as the model HackNIP architecture for this study; note that other combinations might outperform it under different settings.

Data preprocessing

Raw inputs were first sorted into two categories: isolated molecules and extended structures (i.e., atomic assemblies that go beyond discrete molecules, including both crystalline lattices and amorphous solids). Molecular structures were generated from SMILES strings and subsequently relaxed via geometry optimization using ORB prior to feature extraction. Extended structures were expanded into supercells with each lattice vector exceeding 10 Å for ORB[16] and EquiformerV2[15], and 12 Å for MACE[14] to ensure complete neighbor connectivity within the largest model cutoff, considering that edges are created using distance thresholds of 10 Å, 6 Å, and 12 Å for chosen versions of ORB, EquiformerV2 and MACE foundational models, respectively, during graph construction. No further relaxation was performed, as all extended-structure inputs were sourced from DFT-relaxed databases.

Feature extraction

Each atomic structure was passed through NIP foundation models truncated at a specified network depth to yield layer-wise embeddings. In all cases, the raw atomic coordinates and species were converted into the model's internal representation, propagated through the shared encoder, and then through the first L layers of the neural backbone (where L was varied between experiments). The resulting atom-level features were finally averaged over all atoms to produce a fixed-length descriptor vector. By extracting these mean-pooled embeddings at different depths ($1 \le L \le 15$ for ORB, $0 \le L \le 9$ for EquiformerV2, and $1 \le L \le 2$ for MACE), we obtained a hierarchical set of features that capture progressively more global structural information for downstream analysis. The feature dimensionality was 256 for every ORB layer, 128 for every EquiformerV2 layer, and 128- and 256-dimensional for the first and second MACE layers, respectively.

Feature-based machine learning

For downstream property prediction, we treated layer-wise embeddings from each NIP layer as a fixed feature vector and trained three types of ML models: MODNet, XGBoost, and MLP. MODNet models were constructed with four layers with 256 neurons each. XGBoost models were fit with 256 trees, a maximum depth of 4, a learning rate of 0.1, and 80% subsampling of both rows and columns. MLPs employed two hidden layers of 100 units each and were trained for up to 200 epochs. For each model and each embedding depth L, we performed five-fold experiments (using Matbench's standard seed of 18012019), except for the Coulombic-efficiency task with a limited dataset size, for which we used three independent 70/30 splits to match prior studies. We identified the best-performing embedding depth L^* and considered its corresponding predictive performance to represent the HackNIP architecture's performance.

Hyperparameter optimization

For the model HackNIP architecture (ORB + MODNet), hyperparameters were optimized using Optuna[47] within a nested cross-validation framework to avoid potential data leakage. For each outer fold, the dataset was first divided into outer-training and outer-test sets. Hyperparameter optimization was then performed exclusively on the outer-training set with an inner cross-validation loop.

The search space included the following MODNet hyperparameters: batch size, learning rate, number of input features, network depth, and layer width.We performed Optuna trials for each of five ORB depths around the best-performing embedding depths (from $L^* - 2$ to $L^* + 2$). For the exfoliation energy task, where the dataset size is small (< 1000), we performed hyperparameter optimization over the entire range of embedding depths. Within each outer fold, 50 trials were evaluated based on the MAE averaged across inner cross-validation splits with early stopping after 10 non-improving trials.

After selecting the best hyperparameters for each outer fold, the MODNet model was retrained on the full outer-training set and evaluated on the corresponding outer-test set. Final model performance was obtained by averaging the MAEs from the five outer-test folds, and the standard

deviation across folds was reported. Parity plots were generated for the fold whose MAE was closest to the average MAE across all outer folds.

Fine-tuning

For property-specific fine-tuning—used to compare against the HackNIP architecture with ORB feature extraction—we employed the same ORB foundation model (i.e., orb-v2) initialized with pretrained weights and appended a single-output MLP head. The MLP head consisted of two 128-unit layers (256 → 128 → 128) with SiLU activations, followed by a final 128 → 1 linear layer; all linear weights use Xavier uniform initialization, and biases were zero. The dataset was randomly split into training, validation, and test sets in an 8:1:1 ratio. In standard fine-tuning runs, the entire dataset (100%) was used; for data-efficiency experiments, we sampled reproducible random fractions of the data to assess performance as a function of training set size. Most of the ORB backbone was unfrozen, except for the initial encoder and final decoder blocks, which remained fixed; only the MLP head and the unfrozen backbone parameters were optimized. Training proceeded for 200 epochs with AdamW[48] using learning rates of 1×10^{-5} for the backbone and 1×10^{-3} for the head and a weight decay of 1×10^{-4}. The model achieving the lowest validation MAE was checkpointed and then evaluated on the held-out test split to report final test MAE. All model-training experiments were conducted on a workstation equipped with ten NVIDIA RTX 6000 Ada graphics cards and dual Intel Ice Lake–class processors.

Data availability

The data that support the findings of this study are available at Matbench[28], MoleculeNet[38] 3DSC[33], MPContribs[32], and Supplementary information of Ref. 34. The Python code for HackNIP is available on GitHub at https://github.com/parkyjmit/HackNIP.

Code availability

The Python codes used for this work are available on GitHub at https://github.com/parkyjmit/HackNIP.

Received: 16 June 2025; Accepted: 26 May 2026;
Published online: 17 June 2026

References

1. Radford, A., Narasimhan, K., Salimans, T. & Sutskever, I. Improving Language Understanding by Generative Pre-Training (2018).
2. Radford, A. et al. Language models are unsupervised multitask learners. *OpenAI blog* **1**, 9 (2019).
3. Radford, A. et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning* (eds MeilEntity, M. & Zhang, T.), 8748–8763. (PMLR, 2021).
4. Touvron, H. et al. Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971 (2023).
5. Touvron, H. et al. Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288 (2023).
6. Jablonka, K. M., Schwaller, P., Ortega-Guerrero, A. & Smit, B. Leveraging large language models for predictive chemistry. Nature Machine Intelligence 1–9 (Nature Publishing Group UK London, 2024).
7. Merchant, A. et al. Scaling deep learning for materials discovery. *Nature* **624**, 80–85. https://www.nature.com/articles/s41586-023-06735-9 (2023).
8. Yang, H. et al. MatterSim: A Deep Learning Atomistic Model Across Elements, Temperatures and Pressures. https://www.microsoft.com/en-us/research/publication/mattersim-a-deep-learning-atomistic-model-across-elements-temperatures-and-pressures/ (2024).
9. Deng, B. et al. CHGNet as a pretrained universal neural network potential for charge-informed atomistic modelling. *Nat. Mach. Intell.* **5**, 1031–1041 (2023).
10. Barroso-Luque, L. et al. Open Materials 2024 (OMat24) Inorganic Materials Dataset and Models. http://arxiv.org/abs/2410.12771 (2024).
11. Takamoto, S., Izumi, S. & Li, J. TeaNet: Universal neural network interatomic potential inspired by iterative electronic relaxations. *Comput. Mater. Sci.* **207**, 111280 (2022).
12. Takamoto, S. et al. Towards universal neural network potential for material discovery applicable to arbitrary combination of 45 elements. *Nat. Commun.* **13**, 2991 (2022).
13. Li, J. Universal interatomic potentials shine in finding crystal structures. *Nat. Mach. Intell.* **7**, 985 (2025).
14. Batatia, I., Kovacs, D. P., Simm, G., Ortner, C. & Csanyi, G. MACE: Higher Order Equivariant Message Passing Neural Networks for Fast and Accurate Force Fields. *Adv. Neural Inf. Process. Syst.* **35**, 11423–11436 (2022).
15. Liao, Y.-L., Wood, B. M., Das, A. & Smidt, T. EquiformerV2: Improved Equivariant Transformer for Scaling to Higher-Degree Representations. https://openreview.net/forum?id=mCOBKZmrzD (2023).
16. Neumann, M. et al. Orb: A Fast, Scalable Neural Network Potential. http://arxiv.org/abs/2410.22570 (2024).
17. Rhodes, B. et al. Orb-v3: atomistic simulation at scale. http://arxiv.org/abs/2504.06231 (2025).
18. Behler, J. & Parrinello, M. Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces. *Phys. Rev. Lett.* **98**, 146401 (2007).
19. Bartók, A. P., Payne, M. C., Kondor, R. & Csányi, G. Gaussian Approximation Potentials: The Accuracy of Quantum Mechanics, without the Electrons. *Phys. Rev. Lett.* **104**, 136403 (2010).
20. Batzner, S. et al. E (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nat. Commun.* **13**, 2453 (2022).
21. You, Y. et al. Graph contrastive learning with augmentations. *Adv. neural Inf. Process. Syst.* **33**, 5812–5823 (2020).
22. Wang, Y., Wang, J., Cao, Z. & Barati Farimani, A. Molecular contrastive learning of representations via graph neural networks. *Nat. Mach. Intell.* **4**, 279–287 (2022).
23. Fang, X. et al. Geometry-enhanced molecular representation learning for property prediction. *Nat. Mach. Intell.* **4**, 127–134 (2022).
24. Magar, R., Wang, Y. & Barati Farimani, A. Crystal twins: self-supervised learning for crystalline material property prediction. *npj Comput. Mater.* **8**, 231 (2022).
25. Koker, T., Quigley, K., Spaeth, W., Frey, N. C. & Li, L. Graph Contrastive Learning for Materials. arXiv preprint arXiv:2211.13408 (2022).
26. Park, Y. J., Kim, H., Jo, J. & Yoon, S. Deep contrastive learning of molecular conformation for efficient property prediction. *Nat. Comput. Sci.* **3**, 1015–1022 (2023).
27. Shiota, T., Ishihara, K. & Mizukami, W. Universal neural network potentials as descriptors: towards scalable chemical property prediction using quantum and classical computers. *Digital Discov.* **3**, 1714–1728 (2024).
28. Dunn, A., Wang, Q., Ganose, A., Dopp, D. & Jain, A. Benchmarking materials property prediction methods: the Matbench test set and Automatminer reference algorithm. *npj Comput Mater.* **6**, 1–10 (2020).
29. De Breuck, P.-P., Hautier, G. & Rignanese, G.-M. Materials property prediction for limited datasets enabled by feature selection and joint learning with MODNet. *npj Comput. Mater.* **7**, 1–8 (2021).
30. Chen, T. et al. xgboost: Extreme Gradient Boosting. R package version 3.3.0.0. https://github.com/dmlc/xgboost (2026).
31. Pedregosa, F. et al. Scikit-learn: Machine learning in Python. The *JMLR* **12**, 2825–2830 (2011).
32. Zheng, H. et al. The ab initio non-crystalline structure database: empowering machine learning to decode diffusivity. *npj Comput. Mater.* **10**, 1–11 (2024).

33. Sommer, T., Willa, R., Schmalian, J. & Friederich, P. 3DSC - a dataset of superconductors including crystal structures. *Sci. Data* **10**, 816 (2023).
34. Kim, S. C. et al. Data-driven electrolyte design for lithium metal anodes. *Proc. Natl. Acad. Sci.* **120**, e2214357120 (2023).
35. Subramanian, G., Ramsundar, B., Pande, V. & Denny, R. A. Computational Modeling of β-Secretase 1 (BACE-1) Inhibitors Using Ligand Based Approaches. *J. Chem. Inf. Modeling* **56**, 1936–1949 (2016).
36. Martins, I. F., Teixeira, A. L., Pinheiro, L. & Falcao, A. O. A Bayesian Approach to in Silico Blood-Brain Barrier Penetration Modeling. *J. Chem. Inf. Modeling* **52**, 1686–1697 (2012).
37. Gayvert, K. M., Madhukar, N. S. & Elemento, O. A Data-Driven Approach to Predicting Successes and Failures of Clinical Trials. *Cell Chem. Biol.* **23**, 1294–1301 (2016).
38. Wu, Z. et al. MoleculeNet: a benchmark for molecular machine learning. *Chem. Sci.* **9**, 513–530 (2018).
39. Stanev, V. et al. Machine learning modeling of superconducting critical temperature. *npj Comput. Mater.* **4**, 29 (2018).
40. Ward, L. et al. Matminer: An open source toolkit for materials data mining. *Comput. Mater. Sci.* **152**, 60–69 (2018).
41. McInnes, L. Healy, J. & Melville, J. Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426 (2018).
42. Olson, G. B. Computational design of hierarchically structured materials. *Science* **277**, 1237–1242 (1997).
43. Fine, M. E. & Marcus, H. L. Materials science and engineering, an educational discipline. *Annu. Rev. Mater. Sci.* **24**, 1–19 (1994).
44. Oh, H. S. et al. Engineering atomic-level complexity in high-entropy and complex concentrated alloys. *Nat. Commun.* **10**, 2090 (2019).
45. Han, L. et al. Multifunctional high-entropy materials. *Nat. Rev. Mater.* **9**, 846–865 (2024).
46. Chen, T. et al. Xgboost: extreme gradient boosting. R package version 0.4-2. **1**, 1–4 (2015).
47. Akiba, T., Sano, S., Yanase, T., Ohta, T. & Koyama, M. Optuna: A next-generation hyperparameter optimization framework. In *Proc. 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (eds Krishnapuram, B. et al.), 2623–2631 (2019).
48. Loshchilov, I. & Hutter, F. Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017).

Acknowledgements

We acknowledge support from Samsung SDI. S.Y.K. gratefully acknowledges support from the Kwanjeong Fellowship. The authors thank Dr. Choah Kwon and Mr. Hao Tang for their valuable discussions. Y.J.P was supported by a grant from the National Research Foundation of Korea (NRF) funded by the Korean government, Ministry of Science and ICT (MSIT) (No. RS-2026-25497753). The 05-Neuron supercomputer was provided by the Korea Institute of Science and Technology Information (KISTI) National Supercomputing Center.

Author contributions

S.Y.K., Y.J.P., and J.L. designed the study. S.Y.K. carried out key investigations and performed the analyses alongside Y.J.P. and J.L. S.Y.K. and Y.J.P. drafted the original manuscript. Y.J.P. curated the data. J.L. supervised the project. All authors reviewed and edited the manuscript, approved the final version, and agree to be accountable for all aspects of the work.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41524-026-02167-x.

Correspondence and requests for materials should be addressed to Yang Jeong Park or Ju Li.

Reprints and permissions information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

© The Author(s) 2026

Supplementary information.

1 Exploration of model performance on Matbench tasks

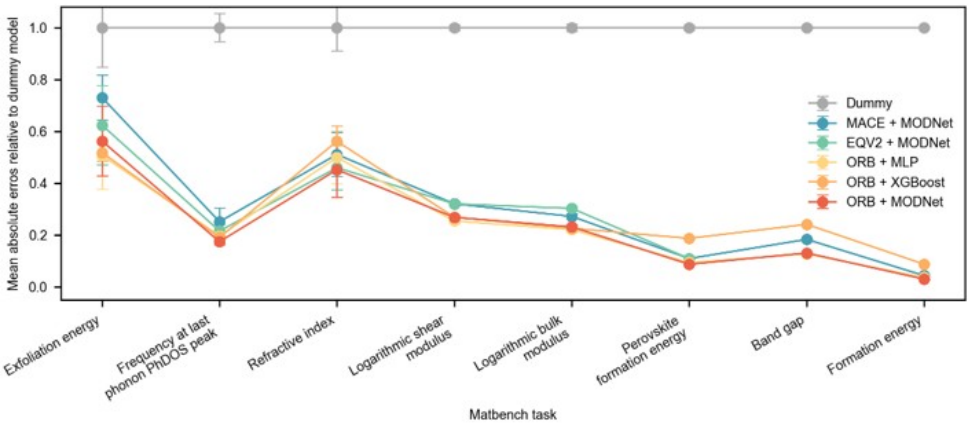


Fig. S1 HackNIP combinations and performance across eight Matbench tasks. HackNIPs built from various NIP foundation models (ORB, EquiformerV2, and MACE) and downstream learners (MODNet, XGBoost, MLP) are shown, with relative mean absolute errors against a dummy model that predicts the training-set mean. Mean absolute errors are averaged over five-fold experiments. When the downstream learner is fixed to MODNet, ORB outperforms EquiformerV2 and MACE. When the NIP model is fixed to ORB, MODNet and MLP outperform XGBoost, and MODNet training is more cost-efficient than MLP training, for the chosen hyperparameters. In other settings, different combinations may outperform.

Table S1 Summary of optimized model parameters for Matbench tasks. ORB embedding depths used for feature extraction and optimized MODNet hyperparameters are shown.

Matbench tasks	ORB embedding depth, L^*	MODNet model parameters			MODNet fitting parameters		
		n_feat	num_neurons	out_act	batch_size	lr	loss
Exfoliation energy, $E_{\text{exfoliation}}$ [meV/atom]	11	256	([32], [32], [32], [32])	relu	32	0.001583	mae
Frequency at last phonon PhDOS peak, ν [/cm]	2	200	([70], [70])	relu	16	0.009284	mae
Refractive index, n [dimensionless]	6	256	([251])	relu	128	0.000175	mae
Logarithmic shear modulus, $\log_{10}(G$[GPa])	6	208	([242], [242], [242])	relu	16	0.000383	mae
Logarithmic bulk modulus, $\log_{10}(K$[GPa])	4	234	([33], [33], [33], [33])	relu	32	0.003104	mae
Perovskite formation energy, $E_{f,\text{perovskite}}$ [eV/unit cell]	3	229	([198], [198], [198])	linear	32	0.000892	mae
Band gap, E_g [eV]	8	236	([88], [88], [88])	relu	32	0.000112	mae
Formation energy, E_f [eV/atom]	3	246	([133], [133], [133])	linear	32	0.000250	mae

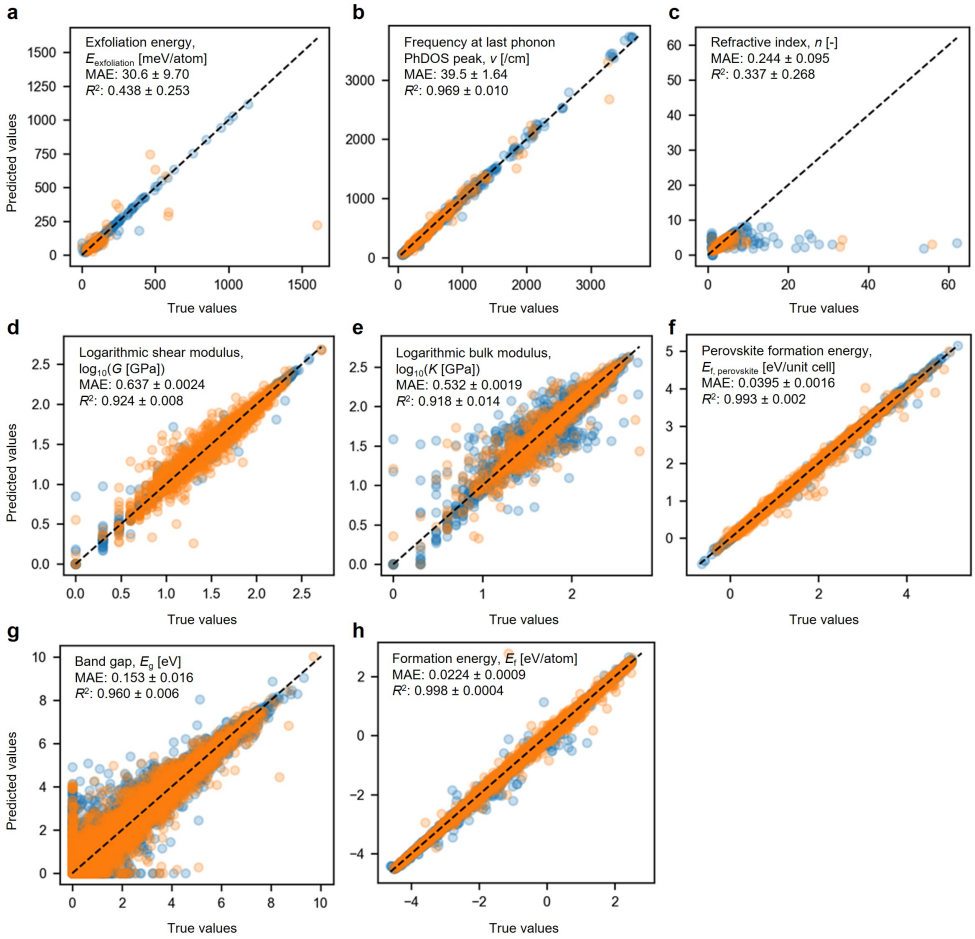


Fig. S2 Parity plots for eight Matbench tasks. a, Exfoliation energy. **b,** Frequency at the last phonon peak in the phonon density of states (PhDOS). **c,** Refractive index. **d,** Logarithmic shear modulus. **e,** Logarithmic bulk modulus. **f,** Perovskite formation energy. **g,** Band gap. **h,** Formation energy. Mean absolute errors (MAEs) and coefficients of determination (R^2) are provided. Blue: Train; Orange: Test.

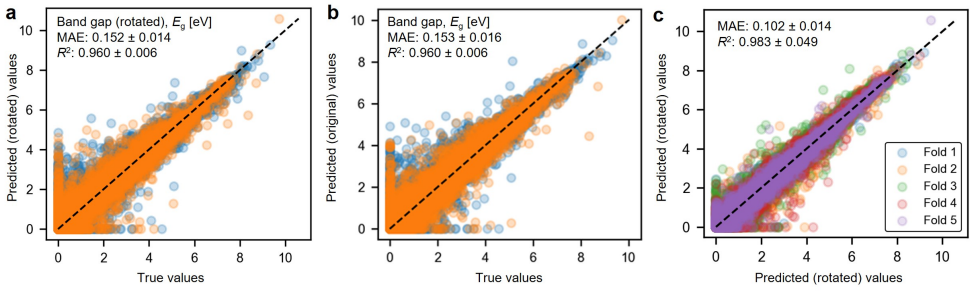


Fig. S3 Geometric orientations and model performance. a, Parity plots for band gap predictions without rotation. **b,** Parity plots for band gap predictions with rotation. **c,** Parity plots comparing predictions with and without rotation. For **b**, only the structures in the test datasets were rotated, and predictions on these datasets were made using models trained on the corresponding unrotated training datasets. Mean absolute errors (MAEs) and coefficients of determination (R^2) are provided. In **a** and **b**, blue: train; orange: test.

2 Data Efficiency

Table S2 Summary of training configurations used to examine data efficiency.

	Fine tuning	HackNIP
Trainable parameters	24M	263k [a]
Epochs	200	200
Batch size	64	64
Number of models	5	5 × 15 [b]

[a] Number of features = 256; width = 256; depth = 3; learning rate = 10^{-4}; activation = ReLU.
[b] Five subsets per layer across 15 ORB layers.

3 Exploration of model performance on diverse downstream tasks

Table S3 Summary of optimized model parameters for diverse downstream tasks. ORB embedding depths used for feature extraction and optimized MODNet hyperparameters are provided.

Tasks	ORB embedding depth, L^*	MODNet model parameters			MODNet fitting parameters		
		n_feat	num_neurons	out_act	batch_size	lr	loss
AIMD[a]-computed Li diffusivity, D_{Li} [cm^2/s]	2	235	([218], [218], [218])	linear	32	0.007568	mae
Logarithmic critical temperature, ln(T_c [K])	3	109	([174], [174], [174])	linear	16	0.000764	mae
Logarithmic Coulombic efficiency, -$\log_{10}$(1 - CE)	1	82	([36])	linear	32	0.000158	mae
Beta-Secretase 1 Inhibition (BACE)	8	240	([194], [194], [194])	-	32	0.000258	-
Blood-Brain Barrier Penetration (BBBP)	13	238	([114])	-	32	0.000183	-
Clinical Toxicity (ClinTox)	11	105	([89], [89])	-	128	0.000147	-

[a]AIMD: ab initio molecular dynamics simulation

4 Embedding depth for feature extraction and model performance

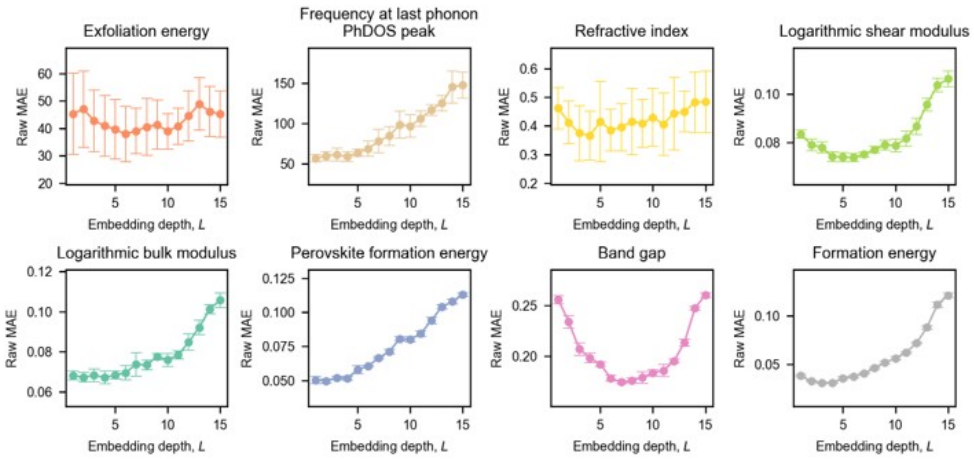


Fig. S4 Embedding depth used for feature extraction and model performance. Raw MAE values corresponding to the normalized MAE values shown in Figure 5a, obtained using fixed hyperparameters prior to optimization. These values therefore differ from those in Figure 2b, which were obtained following optimization.

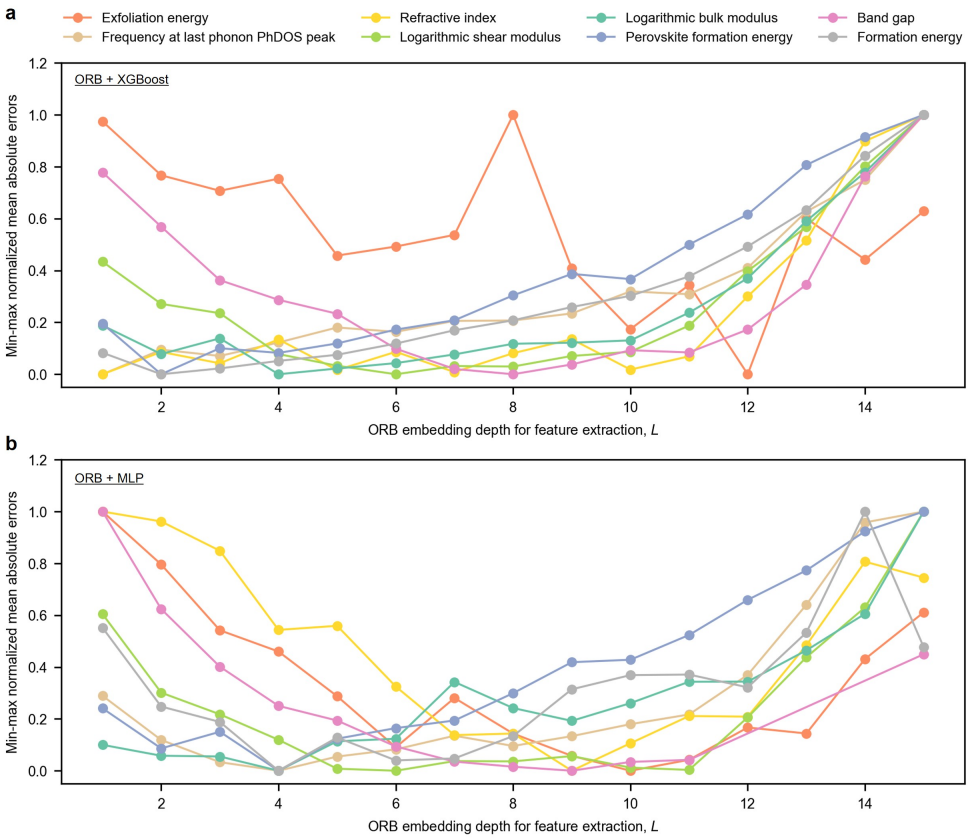


Fig. S5 HackNIP performance as a function of ORB embedding depth used for feature extraction. Two downstream learners are shown: **a,** extreme gradient boosting algorithm (XGBoost) and **b,** multilayer perceptron (MLP). Mean absolute errors are min–max normalized across embedding depths for each Matbench task. Five-fold experiments were performed for each point and only the mean values are presented for clarity.