Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 400 (2022) 115569
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Towards out of distribution generalization for problems in mechanics

Lingxiao Yuan, Harold S. Park*, Emma Lejeune*

Department of Mechanical Engineering, Boston University, United States of America

Received 30 June 2022; received in revised form 12 August 2022; accepted 14 August 2022
Available online xxxx

Abstract

There has been a massive increase in research interest towards applying data driven methods to problems in mechanics, with a particular emphasis on using data driven methods for predictive modeling and design of materials with novel functionality. While traditional machine learning (ML) methods have enabled many breakthroughs, they rely on the assumption that the training (observed) data and testing (unseen) data are independent and identically distributed (i.i.d). However, when these standard ML approaches are applied to real world mechanics problems with unknown test environments, they can be very sensitive to data distribution shifts, and can break down when evaluated on test datasets that violate the i.i.d. assumption. In contrast, out-of-distribution (OOD) generalization approaches assume that the data contained in test environments are allowed to shift (i.e., violate the i.i.d. assumption). To date, multiple methods have been proposed to improve the OOD generalization of ML methods. However, most of these OOD generalization methods have been focused on classification problems, driven in part by the lack of benchmark datasets available for OOD regression problems. Thus, the efficiency of these OOD generalization methods on regression problems, which are typically more relevant to mechanics research than classification problems, is unknown. To address this, we perform a fundamental study of OOD generalization methods for regression problems in mechanics. Specifically, we identify three OOD generalization problems: covariate shift, mechanism shift, and sampling bias. For each problem, we create two benchmark examples that extend the Mechanical MNIST dataset collection, and we investigate the performance of popular OOD generalization methods on these mechanics-specific regression problems. Our numerical experiments show that in most cases, while the OOD algorithms perform better compared to traditional ML methods on these OOD generalization problems, there is a compelling need to develop more robust OOD methods that can generalize the notion of invariance across multiple OOD scenarios. Overall, we expect that this study, as well as the associated open access benchmark datasets, will enable further development of OOD methods for mechanics specific regression problems.

Keywords: Machine Learning; Out of distribution; Regression; Covariate shift; Mechanism shift; Sampling bias

1. Introduction

Traditionally, the mechanical behavior of macroscale heterogeneous solid materials is obtained through laboratory experiments, or by simulation methods such as the finite element method (FEM) [1–4], while the mechanics of smaller length scale structures can be obtained using molecular dynamics simulations [5,6]. These methods are generally accurate and reliable, but can be extremely time-consuming and computationally expensive if there exists a large parameter space to search for an optimal design that maximizes specific functionalities and properties. Both

* Corresponding authors.
E-mail addresses: lxyuan@bu.edu (L. Yuan), parkhs@bu.edu (H.S. Park), elejeune@bu.edu (E. Lejeune).

https://doi.org/10.1016/j.cma.2022.115569
0045-7825/

human intuition and expert knowledge in mechanics can accelerate this process, but expert knowledge is expensive per se. Thus, to make the material design and analysis process more efficient and broadly accessible, there has been significant recent interest in applying machine learning (ML) methods to mechanics [7–13]. There are several compelling reasons why ML methods have become a viable approach for addressing mechanics problems. First, by learning from an existing collection of data samples, ML methods have shown the ability to accurately predict material properties[14–26]. Second, because it takes only seconds for a trained ML model to predict the target properties of thousands of new samples, this significant savings in computational expense enables the rapid and efficient exploration of large design spaces in search of new materials, material designs, and structural designs [27–36]. Third, as a data-driven method that makes predictions by statistically learning the correlation between input feature vectors and the target outputs, ML methods do not require any preliminary expert knowledge for either the learning or prediction processes. Fourth, the accuracy and reliability of a ML model can be continuously enhanced by adding more data through modern techniques such as active learning [27,34,37,38] and reinforcement learning [39–42]. Finally, these ML methods are often transferable such that a ML model trained on a large dataset often only requires minimal additional data when being adapted to a new yet related mechanics problem [43–45].

Despite these significant benefits of ML methods, there are many fundamental underlying challenges to be resolved before their potential can fully be realized in the context of mechanics problems. One of the most important problems is that the excellent performance of ML models on test data is established on the assumption that the training and testing data are independent and identically distributed (i.i.d) [46–48]. However, the data distribution of test environments in practical mechanics problems is usually unknown, and in general the data distribution of test environments and the training environments is not guaranteed to be i.i.d for practical applications. Generating the ML model to test environments for which the data distribution is not identical to the data distribution of the training data that the ML is trained on is known as an out-of-distribution (OOD) generalization problem. While the performance of ML methods on i.i.d. generalization problems is typically excellent given a sufficiently large training dataset, ML models have been shown to be very brittle when applied to OOD problems even for easy-to-learn computer vision tasks [49–51]. For example, after learning from a training dataset with the majority of the datapoints being blue-ish and a minority of the datapoints being green-ish, the accuracy of a ML model on a cat vs. dog classification task drops by more than 20% on a test dataset in which all datapoints are green-ish [51]. Similarly, when most of the cow pictures in the training dataset which the ML model is trained on are collected from common contexts like pastures and grassy fields, the ML model fails to recognize the cows in uncommon contexts like waves and beaches [52].

These results, and the results of many other studies [53–59], suggest that when unseen test data and training data are not i.i.d, the predictions of a ML model built on this assumption should be called into question. Because material design through data driven methods requires exploring large unseen domains through learning on limited available data, the i.i.d. assumption will often be violated in this context [31]. With poor knowledge of the data distribution of the unseen data, the poor generalization of ML models on OOD problems can lead to erroneous predictions on test data from unseen distributions and result in unreliable predictions, and in the context of material design, proposed designs that are far from optimal. For example, when designing composite materials with high strength and stiffness, Kim et al. [60] showed that when the ML model is trained on training data with lower strength and stiffness and used to predict data with higher strength and stiffness, the prediction accuracy drops dramatically, resulting in being trapped in local minima for forward material design. As one line of research, detection of OOD data is of interest because it is related to the reliability of ML model predictions [61–64]. For example, methods following a Bayesian approach that measure the uncertainty of ML predictions have been applied to OOD detection [65–68]. In addition to detecting OOD data, designing ML models that generalize better to OOD samples is an important new direction for applying ML methods to problems in mechanics.

Recently, to improve the robustness and generalizability of ML models, researchers have designed algorithms that embed the prior known physics into ML models when the underlying physics of the training and testing environments are similar and partially known. For example, the well-known approach of physics-informed neural networks (PINNs) embeds partial differential equations (PDEs) into neural networks during the learning process [69–72]. In addition, equivariant neural networks embed known structural symmetries into neural networks, such that the neural networks are robust towards the changes caused by permutation, translation or rotation [73–76]. Another approach, sparse identification of nonlinear dynamics (SINDy), assumes that a dynamical system can be described by a parsimonious model with few key items and parameters, which can be identified by sparse regression. Proponents

of this approach argue that parsimonious models are able to maintain accuracy while preventing over-fitting. Essentially, reducing the complexity of the model is a strategy to improve its generalizability [77–79]. Overall, the methods listed here approach OOD problems by leveraging prior physical knowledge of the system and extracting governing equations from the data. While these approaches are quite powerful, formulating predictive models in this manner is not necessarily suitable for all problems. For example, these approaches often do not extend naturally for making predictions on unseen heterogeneous domains. Alternatively, other recent works in mechanics have focused on methods that improve the robustness of ML models to new domains by active or transfer learning [43,60,80], which updates the current learned model by further collecting a few representative data points from the new test domain. This field is referred as domain adaptation [81–83], which assumes that the data from the test domain is available such that we can collect a few data from the test domain based on some available prior knowledge of the test data distribution [46].

In contrast, OOD generalization, which is also known as domain generalization [46,84], considers a broader situation in which the ML model will be generalized to domains that are unseen and unknown. This unknown distribution of future target datasets prevents us from augmenting the training dataset in a manner that is informed by the characteristics of the new target domain. To improve OOD generalization without collecting new data, ML models are encouraged to learn the causal correlation between inputs and outputs that will not change from training to test environments [85–88], i.e., stable correlations. This is because the failure of the traditional ML models, i.e. Empirical Risk Minimization (ERM) [50,51], on OOD generalization problems can be due to data being mixed and shuffled before training, which makes it difficult for ML models to learn whether a feature of the data is stable (referred to as "invariant") or unstable (referred to as "spurious") when the data distribution shifts from one environment to another [49]. Inspired by this, recently many methods have been proposed to improve OOD generalization by not shuffling data collected from different environments but instead learning the stable features across these environments (or domains) [46,84]. Specifically, these OOD generalization methods consider data collected from multiple environments, while setting two goals for a ML model trained on these environments. First, the ML model is trained to perform well on all training environments, which can be achieved by ERM by minimizing the loss between the predicted value and the ground truth. Meanwhile, the ML model is also trained to perform consistently across all environments, which is achieved by adding a penalty term to the loss function [49,89–91]. Furthermore, there are many other works that prevent a ML model from learning spurious features by different approaches like distributionally robust optimization [47,92], game-theoretic ML [93], and selective rationalization techniques [94]. Compared to ERM, researchers have shown that these methods perform better on OOD classification tasks like recognizing digits in the colored MNIST dataset [49], in which the color of a digit is the spurious and unstable feature that varies across training and testing datasets.

While many methods have been proposed to improve OOD generalization, applying these advances to problems in mechanics is not straightforward for multiple reasons. First, the vast majority of the OOD generalization algorithms have been applied to classification, and not regression problems [89,90,95,96]. As a result, the performance of these approaches on regression problems remains poorly understood. Second, there are only a few distribution shift benchmark datasets for validating the efficiency of these methods. To address this, Koh et al. [97] presented WILDS as a benchmark dataset which is a collection of ten distribution shift datasets from different fields like tumor identification and genetic perturbation classification. However, all ten datasets in WILDS are classification problems, which is again an issue for advancing methods relevant to mechanics as regression problems dominate the mechanics field. At present, due to the unavailability of benchmark regression datasets with data distribution shifts, the efficiency of OOD algorithms on regression problems is usually evaluated on synthetic data of artificially constructed toy problems which do not fully represent the challenges associated with practical regression problems [98–100]. Therefore, to address this gap in the literature, we perform a fundamental study of OOD generalization methods for regression problems in mechanics. Specifically, we identify three OOD generalization problems: covariate shift, mechanism shift, and sampling bias, and develop two benchmark problems for each problem based on extensions to the Mechanical MNIST dataset collection, with access information given in Section 6. Our experiments show that for most cases, while the OOD algorithms outperform traditional ML methods on OOD generalization problems in mechanics, there is a compelling need to develop more robust OOD methods that can generalize notions of invariance across multiple OOD scenarios. Overall, we expect that the combination of this study, as well as the benchmark datasets we developed, will enable further development of OOD methods for regression problems.

The work is organized as follows. In Section 2, we introduce the three kinds of OOD regression problems and present the details of creating the corresponding benchmark examples for each type of OOD problem. In Section 3,

we review the ERM method and introduce three popular algorithms (invariant risk minimization, or IRM [49], risk extrapolation, or REx [90], and inter-gradient-alignment, or IGA [89]) for solving OOD problems. In Section 4, we demonstrate and discuss the OOD generalization performance of ERM and the OOD algorithms on the OOD problems in mechanics introduced in Section 2. Finally, in Section 5, we summarize the main conclusions of this paper.

2. Dataset and computational experiment definition

2.1. Mechanical MNIST collection

The original MNIST dataset [101], the inspiration for Mechanical MNIST, is a benchmark image dataset of labeled handwritten digits $(0-9)$ with consistent preprocessing and formatting. Each image in the dataset is a digit represented by a 28×28 pixel bitmap. For the original MNIST dataset, the standard goal is to train a ML model to classify the correct digit based on the input pixel bitmap [102–104]. Inspired by the original MNIST dataset, we created the Mechanical MNIST dataset as a benchmark dataset specifically for problems in mechanics [105]. In Mechanical MNIST, the images are no longer bitmaps without physical meaning, but instead represent the elastic modulus distribution of a heterogeneous block of material. In Mechanical MNIST, to ensure the elastic modulus values for the image bitmaps have non-zero values and lie within a reasonable range, the 28×28 MNIST image bitmaps are transformed into a map of elastic moduli following the equation:

$$E = \frac{b}{255.0} * (s-1) + 1 \tag{1}$$

where b is the corresponding value of the grayscale bitmap in range $0-255$ and s is set to 100 for the original iteration of the dataset. After the transformation defined by Eq. (1), the elastic modulus distribution of the heterogeneous material lies in range from 1 to s.

In the original contribution to the Mechanical MNIST dataset collection, we considered four different mechanical processes (Confined Compression, Shear, Equibiaxial Extension and Uniaxial Extension). In all cases, these mechanical processes are simulated via the Finite Element Method (FEM) using the open source software FEniCS [106,107]. In this paper, we will focus on the Equibiaxial Extension load case as detailed in our previous work [105]. We note that this dataset is publicly accessible under an open source license, with access information given in Section 6. For the ML problems discussed in this paper, the feature vectors X will contain the modulus distribution of each sample, and the target interest y will contain the change in strain energy after the sample is subject to equibiaxial extension, and thus y is a scalar value for each sample. Specifically, for the finite element simulation, we performed a dimensionless study and used a compressible Neo-Hookean constitutive law, where the Poisson's ratio was set to 0.3 throughout the sample, and the applied extension displacement set to $d = 7.0$ on each boundary, which corresponds to 50% of the square domain side length. Given this basic problem definition, we have both sampled and extended the Mechanical MNIST Dataset Collection to be suitable for OOD experiments. The details of this work are given in the following Sections.

2.2. OOD experiment 1: Covariate shift

2.2.1. Problem definition

Covariate shift is one of the most common OOD shifts. It happens when only the marginal distribution $P(X)$ changes from the training environment to the testing environment $(P_{tr}(X) \neq P_{te}(X))$, while the conditional distribution $P(y|X)$ remains unchanged$(P_{tr}(y|X) = P_{te}(y|X))$ [46,84]. In simpler terms, covariate shift occurs when the input distribution changes while the output distribution does not. Real world covariate shift datasets are readily available for classification problems. For example, in the colored MNIST dataset [49], the same digit appears in different colors between the testing and training environments. In the PACS [108] dataset, dog images are collected from Photo, Art, and Cartoon environments for the training dataset but from a Sketch environment for the testing dataset. In the Waterbird dataset [47], waterbird photographs are mostly taken under a water background in the training dataset, but mostly taken with a land background in the testing dataset.

In contrast to classification problems, covariate shift in regression problems has been much less studied due to the limited availability of relevant regression focused benchmark datasets. This lack of benchmark data motivated us to introduce an open source benchmark dataset to investigate mechanics relevant problems with covariate shift. We introduce this dataset in the next Section.

2.2.2. Dataset description

As shown in Eq. (1), the Mechanical MNIST dataset input patterns are transformed from the original MNIST dataset through an environmental control factor s. The elastic modulus E of blocks that make up the digit are close to s while the elastic modulus E of the blocks in the background is set to 1. When s is much larger than 1 (e.g., $s = 100$), the blocks in the digit area are much stiffer than the soft blocks in the background. Thus the digit will barely deform and almost all deformation during equibiaxial extension loading occurs in the blocks in the background. As a result of this large stiffness mismatch, the effect of changing the environmental factor s within the range $s = 50 - 100$ on the total change in strain energy of the domain is small. Therefore, altering s within this range is an appropriate strategy for inducing a covariate shift as defined in Section 2.2.1. Following this strategy, we introduced two training environments and two test environments summarized below:

- Training Environment 1: $s = 100$, data size = 2500 (2000 for training, 500 for validation)
- Training Environment 2: $s = 90$, data size = 2500 (2000 for training, 500 for validation)
- Testing Environment 1: $s = 75$, data size = 2000
- Testing Environment 2: $s = 50$, data size = 2000

The input and output distributions of our training and testing environments are shown in Fig. 1. In Fig. 1a, we selected one representative example from each of the four environments. For each example, we show the elastic modulus distribution and the corresponding deformation after equibiaxial extension. In Fig. 1b, we visualize the input parameter space of each environment with Principal Component Analysis (PCA). Specifically, we use PCA to reduce the high-dimensional input feature space (784 parameters) to the first principal component and plot the distribution of this one dimensional variable in Fig. 1b with both a histogram and a redundant boxplot to aid in visualization. Note that because the unseen environments are considered unknown, the PCA model is fit exclusively on the data from the training environment. In Fig. 1c, we visualize the output strain energy via the same approach. In these plots, the training and testing environments are distinguished by red and grey hues respectively. As Fig. 1b–c shows, in this covariate shift problem, the input feature distribution shifts substantially between the different environments while the distribution of the strain energy output barely changes. In addition, we note that the difference between the two training environments $s = 100$ and $s = 90$ is substantially smaller than what we will investigate in the testing environments $s = 75$ and $s = 50$.

2.3. OOD experiment 2: Mechanism shift

2.3.1. Problem definition

The notion of concept drift has emerged in ML to describe the change in relationship between input and output data over time. Standard examples of concept drift include not accounting for the season when predicting temperature, or changes in how people use mobile phones over time [109,110]. The elements underpinning the concept drift, or change in the underlying data distribution $P(X, y)$, are referred to as the hidden context.

While primarily used in other areas, this notion is also applicable to mechanics and physics, in which the governing equations do not change over time, but can give inaccurate or physically unreasonable outputs when the control variable increases or decreases significantly. For example, continuum mechanics can break down if the representative length scale is small enough to be comparable to the atomic lattice spacing, or classical mechanics can break down when velocities approach the speed of light. To distinguish these examples from the term concept drift in which the data shift is typically caused by a time-dependent shift of the hidden context, we refer to this phenomena as a mechanism shift. In brief, a mechanism shift is when the data shift is caused by a change in the underlying mechanisms that control the mapping between the input and the output. We introduce our mechanism shift dataset in the next Section.

2.3.2. Dataset description

According to the discussion in Section 2.2.2, changing the value of s when s is large ($s = 50-100$) has little effect on the distribution of the strain energy. Both the training data and the testing data in Section 2.2.2 follow this same underlying deformation mechanism where the embedded digit behaves approximately as a rigid body embedded in a soft surrounding matrix. In this Section, we consider test environments in which s is smaller ($10 - 25$), i.e., the elastic mismatch between the digit and the background matrix is smaller. With a lower stiffness mismatch,

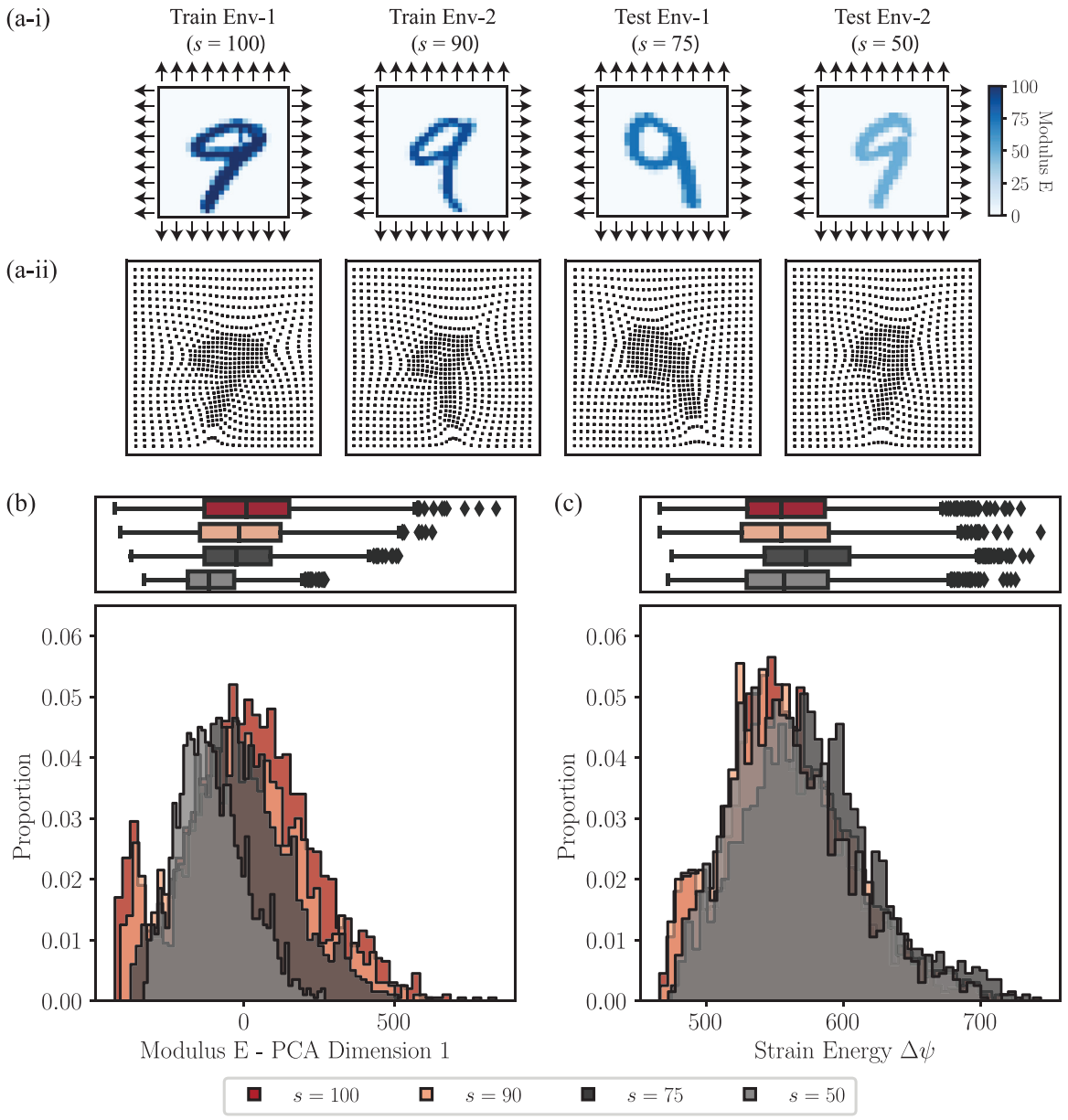

Fig. 1. Illustration of the covariate shift dataset. (a-i) Equibiaxial extension boundary conditions and elastic modulus distribution for a representative example from each environment. (a-ii) Deformation of each example in (a-i) after the completion of the equibiaxial extension simulations. (b) Input distribution of all environments described by the coefficient of the first principle component obtained through PCA performed on the input elastic modulus distribution of training data. (c) Output distribution of all environments defined as the total change in strain energy of the domain. Note that in (b–c) the histograms and boxplots are two ways of showing the same data.

equibiaxial extension loading will cause the digit to deform, and thus changing s will influence the distribution of the final change in strain energy. Following this strategy, we keep the training environments the same as in Section 2.2.2 while introducing two new test environments and summarizing these new test environments along with the previous training environments below:

- Training Environment 1: $s = 100$, data size $= 2500$ (2000 for training, 500 for validation)
- Training Environment 2: $s = 90$, data size $= 2500$ (2000 for training, 500 for validation)

- Testing Environment 1: $s = 25$, data size $= 2000$
- Testing Environment 2: $s = 10$, data size $= 2000$

The input and output distributions of our training and testing environments are shown in Fig. 2. As shown in Fig. 2(a-i) and (a-ii), compared to the representative examples from the training environments ($s = 100$, $s = 90$), the deformation of the blocks near the digit is much smoother for the representative examples from the test environments ($s = 25$, $s = 10$). Due to this deformation mechanism change, the distribution of the strain energy $\Delta\psi$ also shifts, as shown in 2(c). Therefore, in contrast to the scenario posed in Section 2.2.2, the scenario posed here contains both a shift in the input distribution and a shift in the output distribution. The bi-leveled distribution shift caused by the Mechanism shift makes it a larger challenge for the OOD generalization.

2.4. OOD experiment 3: Sampling bias

2.4.1. Problem definition

Sampling bias happens when the data that is selected for training a ML model is not representative of the entire data pool that will be explored once the model is deployed. The presence of sampling bias makes it more difficult to find causal effects in the dataset by potentially introducing spurious correlations between input features and output properties, thus causing failures in OOD generalization. Sampling bias can exist in any empirical research if the underlying causal relationship is unknown [111,112]. For example, although ImageNet [113] has a very large sampling size with more than 14 million images, CNNs trained on ImageNet show bias by recognizing an animal by its texture rather than by its shape [54].

Studying the OOD problems caused by sampling bias is critical for material design by data-driven methods. This is because collected samples usually contain some form of bias, and thus predictions made by a ML model may be a function of a spurious relationship that will not exist in the testing environment. To study the OOD shift caused by sampling bias, Kuang et al. [99] introduced a selection mechanism in which data points are selected by a selection probability determined by spurious relationships between the dataset inputs and outputs. Liu et al. [114] later applied state-of-the-art OOD algorithms on synthetic data generated by this selection mechanism and showed that they outperform traditional ML methods that do not account for the presence of sampling bias. Although it is convenient to perform experiments on synthetic data where spurious features directly augment the input feature vector, real-world problems often contain sampling biases that are far more subtle. Thus, additional benchmark datasets to examine the performance of ML models in the face of sampling bias remain an important need that the broader field is currently lacking. To provide a benchmark data resource for OOD studies on sample selection bias, we introduce a sampling bias dataset within the Mechanical MNSIT dataset collection described in the next Section.

2.4.2. Dataset description

The input for the Mechanical MNIST dataset is the elastic modulus map for each material domain, which contains no inherent spurious features as every component of the input vector is the Young's modulus E of a block domain and thus contributes to the final change in strain energy of the domain. However, the dataset does potentially contain implicit spurious features embedded in the input vectors that can incorrectly be regarded as causal of the output by a ML model during the training process. In other words, certain characteristics of the input distribution may be spuriously correlated with the simulation output. And, more critically, we have the ability to define spurious features and intentionally select for them when we are establishing our test and training environments. To induce spurious correlations, we define the sum of the modulus of each block domain $V_i = \sum_{j=1}^{784} E_j$ as an implicit spurious feature. Then, we sample data points from the original Mechanical MNIST distribution ($s = 100$) through a biased selection mechanism. Each point in the original distribution has a selection probability P_i that is designed to induce sampling bias in a controlled fashion through the distribution control parameters r. To establish P_i, the selection probability of data point i, we consider the sum of modulus V_i and the final change in strain energy y_i. The biased selection mechanism, inspired by the mechanism defined in Kuang et al. [99], then takes the form:

$$P_i = |r|^{-5|\,\tilde{y}_i - \mathrm{sign}(r)\tilde{V}_i\,|} \tag{2}$$

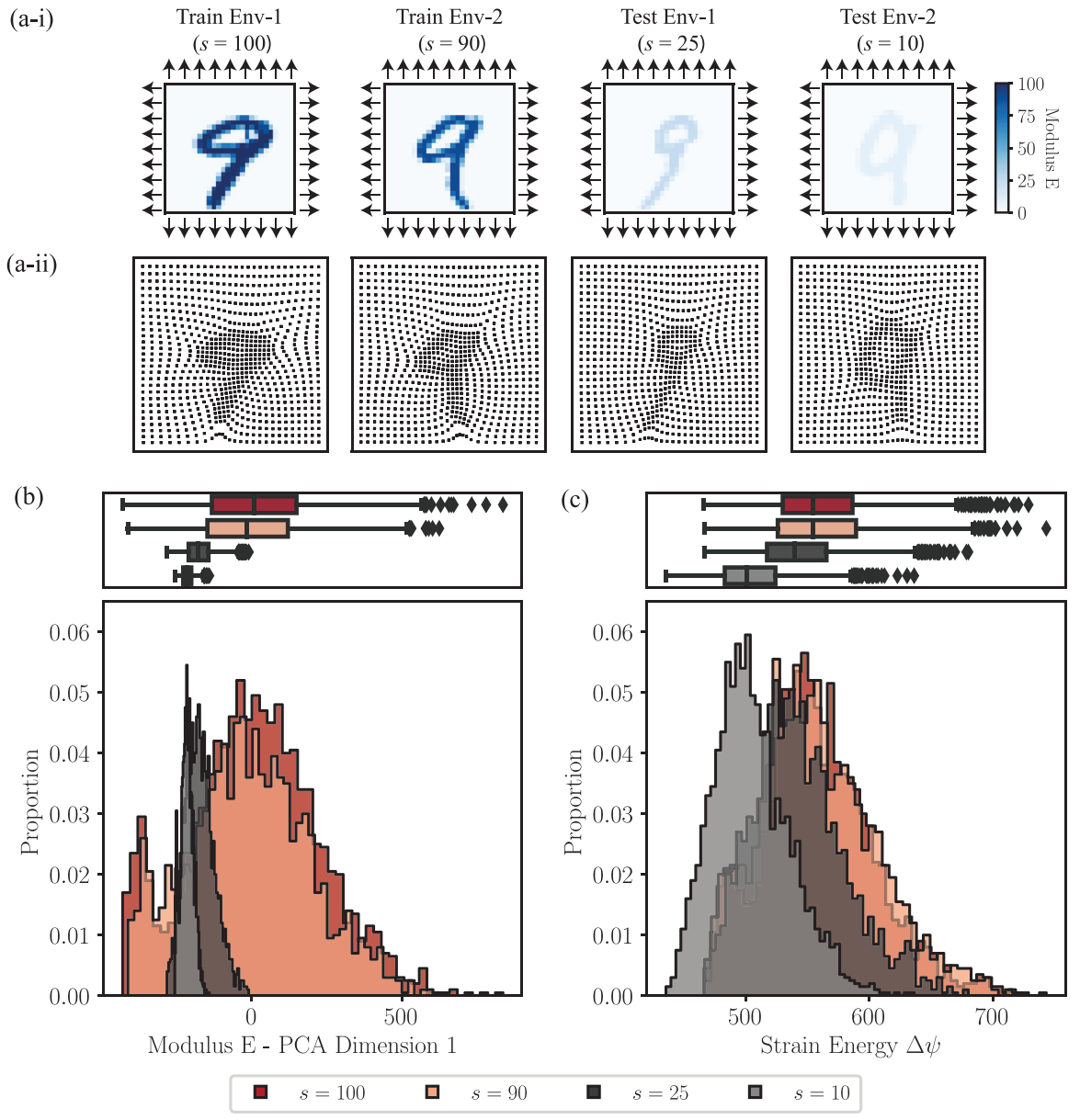

Fig. 2. Illustration of the mechanism shift dataset. (a-i) Equibiaxial extension boundary conditions and elastic modulus distribution for a representative example from each environment. (a-ii) Deformation of each example in (a-i) after the completion of the equibiaxial extension simulations. (b) Input distribution of all environments described by the coefficient of the first principle component obtained through PCA performed on the input elastic modulus distribution of training data. (c) Output distribution of all environments defined as the total change in strain energy of the domain. Note that in (b–c) the histograms and boxplots are two ways of showing the same data.

where $|r| \geq 1$ and sign(r) is a sign function with sign$(r) = 1$ if $r > 0$ and sign$(r) = -1$ if $r \leq 0$. In addition, our implicit spurious feature V_i and the output distribution of y are normalized through the functions:

$$\tilde{V}_i = \frac{V_i - V_{\text{mean}}}{V_{\text{std}}} \qquad \text{and} \qquad \tilde{y}_i = \frac{y_i - y_{\text{mean}}}{y_{\text{std}}} \tag{3}$$

where $V_{\text{mean}} = 10925.34$ and $V_{\text{std}} = 3360.66$ for the sum of modulus implicit feature V of the Mechanical MNIST training dataset, and $y_{\text{mean}} = 567.52$ and $y_{\text{std}} = 47.35$ corresponds to the output change in strain energy y of the

Mechanical MNIST training dataset. Parameter r in this selection mechanism controls the direction and the strength of the spurious correlation: positive spurious correlation occurs when $r > 1$, and negative spurious correlation occurs when $r < -1$. Specifically, when $r > 1$, the data points with the implicit spurious feature $\tilde{V}_i$ more highly correlated to their final change in strain energy $\tilde{y}_i$ will have a larger chance of being selected; when $r < -1$, the data points with the negative value of its implicit spurious feature $\tilde{V}_i$ closer to their final change in strain energy $\tilde{y}_i$ will have a larger chance of being selected. In addition, larger $|r|$ means larger selection bias. We note that for ease of implementation of our sampling algorithm, the final selection probability of each data point is normalized as $\tilde{P}_i = P_i / \sum_{j=1}^{n} P_j$ where n is the total number of data points. This ensures that the sum of the probability of all data points is equal to 1, which is statistically meaningful. Based on this selection mechanism, we created two training environments and three testing environments dictated by the selection parameters below:

- Training Environment 1: $r = 15$, data size = 9800 (7840 for training, 1960 for validation)
- Training Environment 2: $r = -2$, data size = 200 (160 for training, 40 for validation)
- Testing Environment 1: $r = -5$, data size = 2000
- Testing Environment 2: $r = -10$, data size = 2000
- Testing Environment 3: $r = 1$, data size = 2000

In Fig. 3a, we illustrate the relationship between the implicit spurious feature $\tilde{V}_i$ and the output $\tilde{y}_i$ for each environment. The color of each plot background represents the logarithmic selection probability of each area calculated by Eq. (3). Note that the selection probability is normalized through being divided by 10000, which is the total number of pixel points that construct the background of the image. In training environment 1 where $r > 1$, data with positive spurious correlations will be overrepresented in the environment, thus $\tilde{y} \approx \tilde{V}$ throughout the dataset. In training environment 2, testing environment 1, and testing environment 2 where $r < -1$, data points with a negative spurious correlation will be overrepresented in the environment, thus data points around $\tilde{y} = -\tilde{V}$ have a larger probability to be selected. In testing environment 3 where $r = 1$, the selection probability is identical for every data point (i.e., $P_i = 1/n$), thus testing environment 3 is a representative sampling of the entire Mechanical MNIST dataset. In Fig. 3b–c, we plot the data input and output distributions following the format in Figs. 1–2b and c. As shown in Fig. 3a, the implicit spurious features and the output strain energy are strongly correlated in the training dataset but not in the testing datasets. Furthermore, we note that in Fig. 3b, though the data in training environment 1 is from a biased selection, in terms of the first principle component of PCA, the biased selection data distribution of training environment 1 (mean = −1.38, standard deviation = 216.65) and testing environment 3 (representative Mechanical MNIST dataset, mean = 65.06, standard deviation = 211.18) are within one standard deviation of each other, which demonstrates that it can be non-trivial to detect the presence of implicit spurious features.

2.5. Mechanical MNIST – EMNIST letters collection

The original Extended MNIST (i.e., EMNIST) dataset is a benchmark image dataset following the same conversion paradigm used to create the MNIST dataset, except that each image in the original EMNIST Letters dataset is a letter (a–z, A–Z) rather than a digit $(0-9)$ [115]. Identical to the MNIST dataset, each input domain is represented by a 28×28 pixel bitmap. To create the Mechanical MNIST – EMNIST Letters dataset, we follow the same process as described in Section 2.1 for Mechanical MNIST. Specifically, we created the Mechanical MNIST – EMNIST Letters dataset by transforming the 28×28 image bitmaps to a map of the elastic modulus for each input pattern in EMNIST Letters through Eq. (1). We also created a covariate shift dataset through the same process described in Section 2.2, a mechanism shift dataset through the same process described in Section 2.3, and a sampling bias dataset through the same process described in Section 2.4. Note that when creating the sampling bias dataset for Mechanical MNIST – EMNIST Letters through the selection probability defined in Eq. (3), $V_{\text{mean}} = 14151.59$ and $V_{\text{std}} = 4045.36$ are the mean and standard deviation of the spurious feature V of the Mechanical MNIST – EMNIST Letters training dataset, and $y_{\text{mean}} = 708.50$ and $y_{\text{std}} = 88.24$ corresponds to the mean and the standard deviation of the output change in strain energy y of the Mechanical MNIST – EMNIST Letters training dataset. Illustrations of the three OOD datasets for Mechanical MNIST – EMNIST Letters are shown in Appendix A. In this work, the results presented in Section 4 are obtained for the three OOD datasets with both Mechanical MNIST and Mechanical MNIST – EMNIST Letters. The methods used to obtain these results are described next in Section 3.

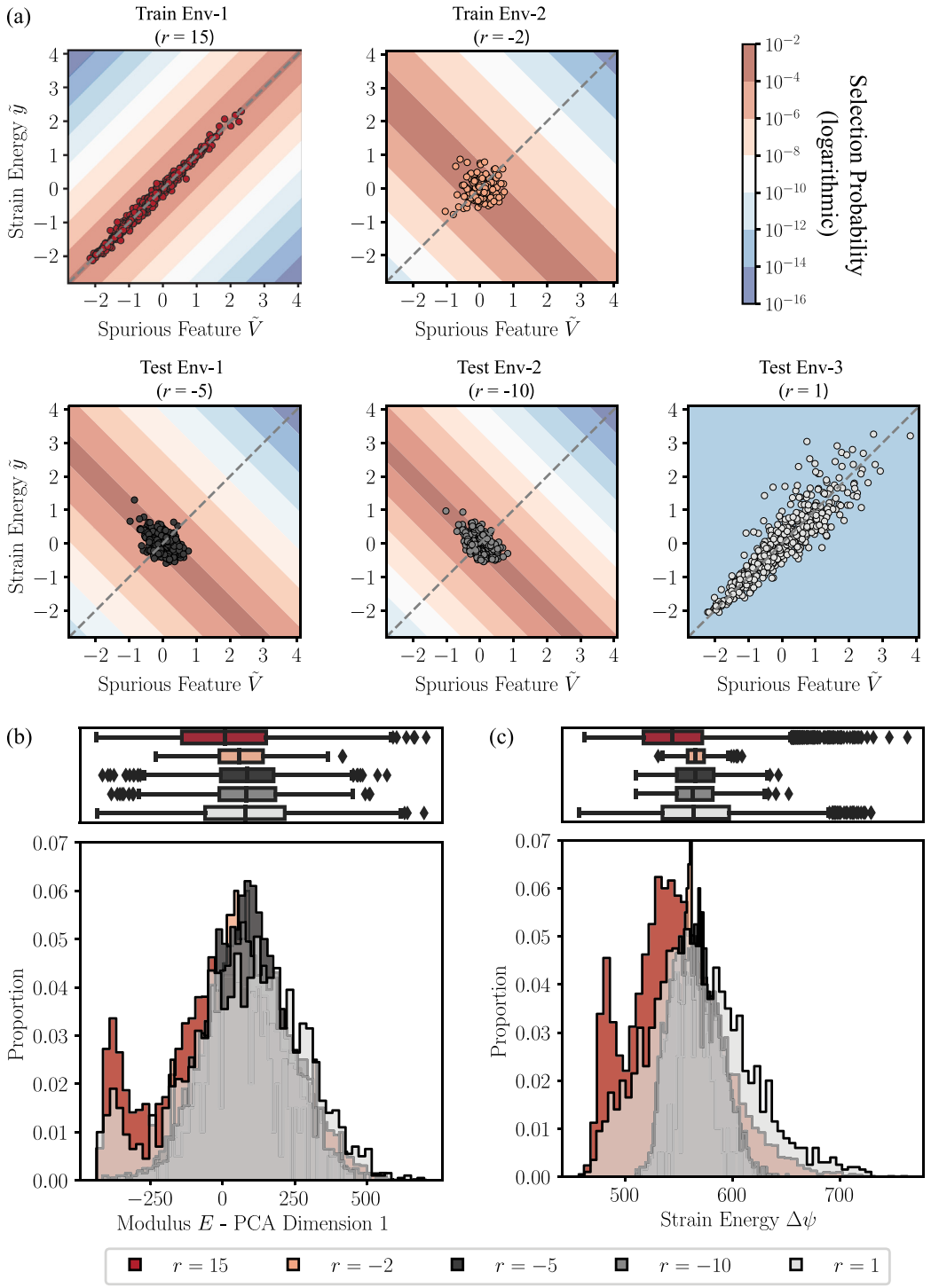

Fig. 3. Illustration of the sampling bias dataset. (a) Relationship between the spurious feature and the output target (the total change in strain energy) for each environment. Note that we only show the 500 randomly selected data points for each environment (with the exception of the 200 available data points for training environment 2) to aid in visualization. The color of each plot background represents the logarithmic selection probability of each area (b) Input distribution of all environments described by the coefficient of the first principle component obtained through PCA performed on the input elastic modulus distribution of training data. (c) Output distribution of all environments defined as the total change in strain energy of the domain. Note that in (b–c) the histograms and boxplots are two ways of showing the same data. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

3. Methods

3.1. Algorithms and notation

For the supervised learning problems that we will pursue in this paper, we define the input space as $X \subseteq \mathbb{R}^{784}$ and the output space as $y \subseteq \mathbb{R}^1$. The goal of machine learning models is to find a predictor $f_\theta(X)$ that minimizes the loss function $l(f_\theta(X), y)$. Traditionally, the training (observed) data and test (unseen) data are assumed to be independent and identically distributed (i.i.d). Thus, the optimal predictor can be obtained by minimizing the loss of the model on the training data. This method is referred to as Empirical Risk Minimization (ERM). However, out-of-distribution (OOD) generalization problems deviate from the i.i.d. assumption underlying ERM. In other words, for OOD problems we cannot assume that the training data and testing data are i.i.d. This means that the distribution of unseen data is allowed to shift, which is practically relevant to real world problems. Specifically, OOD generalization approaches acknowledge that the observed training data are collected from different environments $\mathcal{E}^{\text{train}} = \{e_1, \ldots, e_m\}$, which are only a subset of all environments $\mathcal{E}^{\text{all}}$. For an OOD generalization method to perform well, this implies that the predictor $f_\theta(X)$ obtained through learning from the training environments should perform well across all unseen environments that are under consideration. This goal can be expressed by minimizing the worst-case risk defined as:

$$R^{\text{OOD}}(\theta) = \max_{e \in \mathcal{E}^{\text{all}}} R^e(\theta) \tag{4}$$

where θ represents the parameters of the predictor (e.g., weights for a neural network). To minimize the OOD risk defined in Eq. (4), Arjovsky et al. [49] argued that ML models should learn the causation that truly defines the outcome. In defining causation, we distinguish it from correlation, where correlation can be either spurious or causal. A ML model that learns a spurious correlation between input and output can make accurate predictions in training environments, but not in test environments. This is the main reason why a ML model driven by ERM fails in OOD generalization. In contrast, a causal correlation is one that is invariant, and thus does not change across different environments. As a result, a ML model that learns the causation would perform consistently well across all environments, which is the ultimate goal of OOD generalization.

Because causation does not change across environments, Arjovsky et al. [49] promoted "invariance" as the main feature of causation and determined that ML models that perform well in OOD generalization should find a predictor that learns the invariant correlation across all environments. We call a predictor that reaches this goal an invariant predictor. In addition to the general requirement for a predictor that it should perform well on the whole training data set collected from all environments, the invariant predictor is also required to exhibit a second quality, referred to as "invariance", which is defined as having consistent performance across every environment.

While the ERM method achieves the general requirement for a predictor, how to get an invariant predictor that exhibits invariance across different environments is the main challenge for developing OOD generalization methods. In the remainder of this Section, we first introduce ERM as a baseline method, and then introduce three additional widely applied OOD algorithms that aim to find an invariant predictor.

3.1.1. Empirical Risk Minimization (ERM)

The Empirical Risk Minimization (ERM) approach assumes that the training data and the test data are i.i.d. Thus, the optimal predictor f_θ can be found by minimizing the average risk of all training environments. The risk form of ERM is defined as:

$$R^{\text{ERM}} = \sum_{e=1}^{m} R_e(\theta) \tag{5}$$

where e denotes each training environment. In brief, ERM makes no special acknowledgments that the problem at hand is an OOD problem. Thus, ERM is typically used to define a baseline prediction.

3.1.2. Invariant Risk Minimization (IRM)

Motivated by the idea that causation often does not happen explicitly through input variables of a dataset (e.g., in computer vision the collection of correlated pixels that define a recognizable object are often not explicitly definable), Arjovsky et al. [49] proposed the Invariant Risk Minimization (IRM) algorithm to search for an invariant

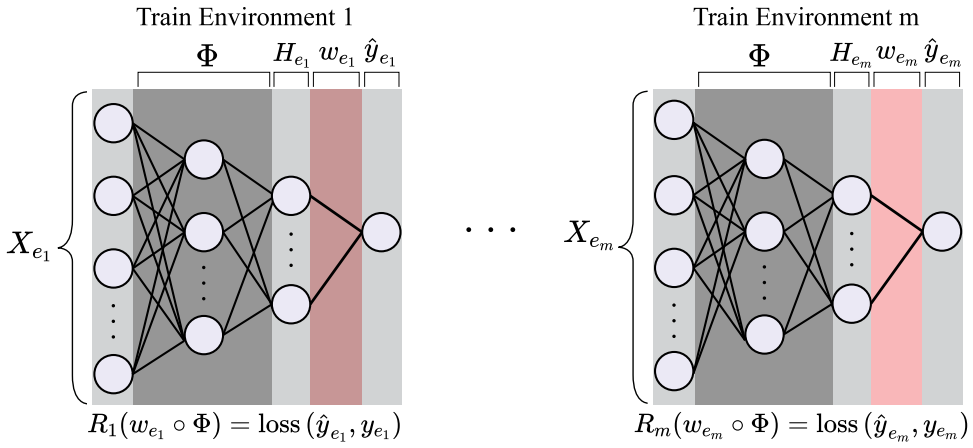

Fig. 4. Example illustration for IRM.

predictor through finding a data representation for which the optimal predictor is the same for all environments. Formally, they define a data representation $\Phi : \mathcal{X} \to \mathcal{H}$ that elicits an invariant predictor $w \circ \Phi$ across environments $\mathcal{E}^{\text{train}}$ if there is an optimal classifier w for $\bar{w} : \mathcal{H} \to \mathcal{Y}$, that is simultaneously optimal for all environments, i.e., $w \in \arg\min_{\bar{w}:\mathcal{H}\to\mathcal{Y}} R_e(\bar{w} \circ \Phi)$, for all $e \in \mathcal{E}^{\text{train}}$. We note that the authors of IRM method used the term "classifier" to denote the last layer $\bar{w}$ for both classification and regression problems. Though here we only consider regression problems in mechanics, we still use the term "classifier" in order to keep it consistent with the original IRM paper [49]. The mathematical form of IRM is given as:

$$\min_{\substack{\Phi:\mathcal{X}\to\mathcal{H} \\ w:\mathcal{H}\to\mathcal{Y}}} \sum_{e=1}^{m} R_e(w \circ \Phi) \quad \text{subject to} \quad w \in \arg\min_{\bar{w}:\mathcal{H}\to\mathcal{Y}} R_e(\bar{w} \circ \Phi), \quad \text{for all } e \in \mathcal{E}^{\text{train}}. \tag{6}$$

To help interpret the terms in Eq. (6) in the context of neural networks, we illustrate the corresponding components schematically in Fig. 4. In this example, we show training data collected from m environments. The input and output data of each training environment is defined as $\{X_{e_i}, y_{e_i}\}$. The data representation Φ that transforms X_{e_i} to H_{e_i} is the same across all environments. Determining the associated weights of Φ, w_{e_i}, is the goal of the IRM algorithm. The optimal w_{e_i} for each training environment is found by minimizing the loss $l(\hat{y_{e_i}}, y_{e_i})$ between $\hat{y_{e_i}} = w_{e_i} \circ \Phi(X_{e_i})$ and y_{e_i}, i.e., $R_i(w_{e_i} \circ \Phi)$. Thus, in this example the goal of Eq. (6) is to search for a data representation Φ where w_{e_i} is identical across all training environments, i.e., $w_{e_1} = w_{e_2} = \ldots = w_{e_m}$.

Because solving the bi-level optimization problem defined in Eq. (6) is very challenging, Arjovsky et al. [49] simplified the optimization problem by assuming that the optimal classifier w is a linear and fixed vector $w = w^*$. Thus, the risk given by the practical version of IRM is:

$$R^{\text{IRM}} = \sum_{e=1}^{m} R_e(w^* \circ \Phi) + \lambda \cdot \left\| \nabla_{w|w=w^*} R_e(w \circ \Phi) \right\|^2. \tag{7}$$

For this version of IRM, the goal becomes to find a data representation Φ such that the optimal w_{e_i} of each training environment is w^*. The first item of Eq. (7) measures the predictive power of the predictor $w^* \circ \Phi$ on the training data environments. The second term is the gradient penalty that measures the optimality of the choice $w = w^*$ for all training environments. Because the practical form assumes that the optimal w for all training environments is $w = w^*$, the gradient of $R_e(w \circ \Phi)$ should reach its minimum at $w = w^*$, i.e., the gradient of $R_e(w \circ \Phi)$ with respect to w should be zero at $w = w^*$. More discussion about the relationship between the original objective Eq. (6) and the practical objective Eq. (7) can be found in [49]. Then, Arjovsky et al. [49] set w^* to be a fixed scalar $w^* = 1.0$ such that the practical form of IRM is given by:

$$R^{\text{IRM}} = \sum_{e=1}^{m} R_e(\Phi) + \lambda \cdot \left\| \nabla_{w|w=1.0} R_e(w \cdot \Phi) \right\|^2. \tag{8}$$

This practical version of IRM is composed of two terms with a penalty weight λ that controls the balance between the terms. The first term is identical to the ERM term defined in Eq. (5). The second term is the gradient penalty that measures the optimality of the pragmatic choice $w = 1$ for all training environments. We will evaluate the IRM method on out of distribution tasks through its practical form Eq. (8) in Section 4.

3.1.3. Risk Extrapolation (REx)

Inspired by IRM, Krueger et al. [90] aim to find an invariant predictor through finding an "equipredictive" representation Φ which they defined as a data representation with the property that the distribution $P_e(y|\Phi)$ is equal for $e \in \mathcal{E}^{\text{all}}$. In order to help build a geometric intuition of REx, they emphasized that their method is an extension to another OOD generalization method, Distributionally Robust Optimization (DRO) [47], the objective of which is a risk interpolation defined as:

$$R^{\text{RI}} = \max_{\substack{\sum_e \lambda_e = 1 \\ \lambda_e \geq 0}} \sum_{e=1}^{m} \lambda_e R_e(\theta) \tag{9}$$

The REx method instead considers that the coefficient of the risk from each environment can be negative such that it allows us to extrapolate to more extreme variations. Since the final goal of OOD generalization is to minimize the maximal risk (or the worst case risk) across all environments, they call their method as minimax Risk Extrapolation (MM-REx), the objective of which is defined as:

$$R^{\text{MM-REx}} = \max_{\substack{\sum_e \lambda_e = 1 \\ \lambda_e \geq \lambda_{\min}}} \sum_{e=1}^{m} \lambda_e R_e(\theta) \tag{10}$$

where $\lambda_{\min}$ is a hyperparameter that controls how much to extrapolate. To minimize Eq. (10), the optimal solution is obtained at $R_1 = R_2 = \cdots = R_m$. In other words, REx encourages the equality of risks from different environments, which can also be achieved by minimizing the variance of risks across training environments. In practice, they found the optimization landscape is smoother by using the variance of risks to be the penalty term of the risk objective. Therefore their practical form of the risk extrapolation is given by:

$$R^{\text{REx}} = \sum_{e=1}^{m} \mathcal{R}_e(\theta) + \lambda \text{Var}(\{\mathcal{R}_1(\theta), \ldots, \mathcal{R}_m(\theta)\}) . \tag{11}$$

Similar to Eq. (8), the form of REx is composed of an ERM term and a penalty term where a penalty weight λ controls the balance between the two terms. In Eq. (11), the penalty term measures the variance of risks across training environments. By directly focusing on the invariance of risks, the creators of REx argue that it provides robustness to covariate shift where IRM can easily fail. We will evaluate the veracity of this claim later through its practical form Eq. (11) in Section 4.

3.1.4. Inter Gradient Alignment (IGA)

Inspired by information theory, Koyama and Yamaguchi [89] attempt to find a data representation Φ that maximizes the mutual information between Φ and the output interest y. In statistics, the mutual information measures the mutual dependence between two variables. It becomes zero if Φ and Y are independent, and becomes largest if Φ is a deterministic function of y. They called the invariant predictor found by this method as the maximal invariant predictor (MIP). To seek the MIP, they proposed Inter Gradient Alignment (IGA) that forces the risk gradient of each training environment to align with each other, i.e., $\nabla_\theta R_1 = \nabla_\theta R_2 = \cdots = \nabla_\theta R_m$. This was implemented by minimizing the variance of the gradient of risks across all environments. Thus the risk form of the IGA algorithm is defined as:

$$R^{\text{IGA}} = \sum_{e=1}^{m} \mathcal{R}_e(\theta) + \lambda \, \text{trace} \left(\text{Var}(\{\nabla_\theta R_1(\theta), \ldots, \nabla_\theta R_m(\theta)\}) \right) . \tag{12}$$

Like with Eqs. (8) and (11), the first term of the IGA equation is an ERM term. In Eq. (12), the second term is the trace of the variance of risk gradient, and λ is a penalty weight control parameter that controls the balance between the two terms. The mathematical relationship between MIP objective and IGA algorithms can be found in [89]

3.1.5. Limitations

The risk functions of the three algorithms designed for OOD generalization problems all follow the same form. The first term is an ERM term that corresponds to the general quality of a predictor, i.e., low error on training data. The second term penalizes the variance of the predictor across all training environments $\mathcal{E}_{\text{train}}$, which aims to reach the second quality of an invariant predictor defined in Section 3.1, i.e., invariance across all environments $\mathcal{E}^{\text{all}}$. To make sure the invariant predictor for $\mathcal{E}^{\text{train}}$ is also the invariant predictor for $\mathcal{E}^{\text{all}}$, the invariance across $\mathcal{E}^{\text{train}}$ should imply the invariance across $\mathcal{E}^{\text{all}}$ [49]. However, this is difficult to realize for real world data. On the other hand, based on the different assumptions underlying each of these algorithms, they may optimally target different types of OOD problems, though unfortunately there exists no guide for the optimal ODD problem for a given OOD algorithm. In Section 4 we will test these algorithms on the different types of OOD problems defined in Section 2 to have a better understanding on how they perform on different OOD problems specific to mechanics data.

3.2. Models

In this Section, we briefly introduce the two base models used to test the four algorithms introduced in Section 3.1 on the three OOD datasets introduced in Section 2. The first model is a Multilayer Perceptron (MLP) Model, while the second model is a LeNet [101] model which was originally designed for digit classification on the MNIST dataset. In comparison to the original LeNet model that was designed for classification problems, we modified the activation (Softmax $\rightarrow$ ReLu) and pooling (Avepooling $\rightarrow$ Maxpooling) layers of the LeNet model to better serve our regression problems from the Mechanical MNIST dataset. The structure of both models is schematically illustrated in Fig. 12, Appendix B.1.

For the OOD datasets targeting covariate shift and mechanism shift, we use the true unscaled modulus values as model inputs because the shifts in the output feature strain energy are controlled by the scale of the modulus. Thus, rescaling the input could impede the OOD algorithms from learning invariant features between the different environments. For the OOD dataset targeting sampling bias where $s = 100$ is applied for all data, the inputs for the models are modulus values that were scaled to be within $0 \sim 1$ by dividing by 100 to help achieve faster model convergence.

3.3. Hyperparameter tuning and performance evaluation metrics

3.3.1. Model selection/hyperparameter tuning

Although OOD generalization problems are very important for the real world applicability of many ML techniques, and many promising algorithms have recently been proposed [46], one major issue is that there does not exist a standardized approach to select models for such problems [96]. In ML frameworks, it is imperative that performance is ultimately evaluated on unseen test data [96]. For OOD approaches, the delineation between test and training datasets remains essential, yet the violation of the i.i.d. assumption between the test and training datasets adds another layer of complication. Because each test environment can be qualitatively and quantitatively different, it is likely that performance will vary widely across different test environments for an identical set of model hyperparameters. Although existing OOD algorithms have shown robustness on shifted test datasets in comparison to the ERM method, these performance enhancements are obtained by tuning the hyperparameters of the OOD algorithm (i.e., tuning the penalty weight term in the risk function) on test environments [90]. In this work, we take special care to define test environments that are truly "unseen" i.e., are not used in any way to select or tune model hyperparameters. To accomplish this, we split our training datasets into training (80%) and validation datasets (20%) while tuning the hyperparameters on the validation datasets alone. For the OOD algorithms introduced in Section 3.1, we followed the implementation process used in [49]. Specifically, there are two main hyperparameters to tune: the penalty weight λ, and the anneal step t, after which we add the penalty (invariant term) to the loss function. The anneal step is determined by the step after which the validation error stops decreasing for ERM. During the training process, the penalty weight is set to $10^{-4}\lambda$ before the anneal step and set to be λ afterwards. The final optimal penalty weight is then selected as the weight that reaches the lowest mean validation error for three models trained with different random initializations. Note that if the penalty weight is too small, the penalty term has little influence on the training process such that the training history of the OOD algorithms will be similar to ERM, and so we neglect such small penalty weights during the hyperparameter tuning process. One example of this is shown in

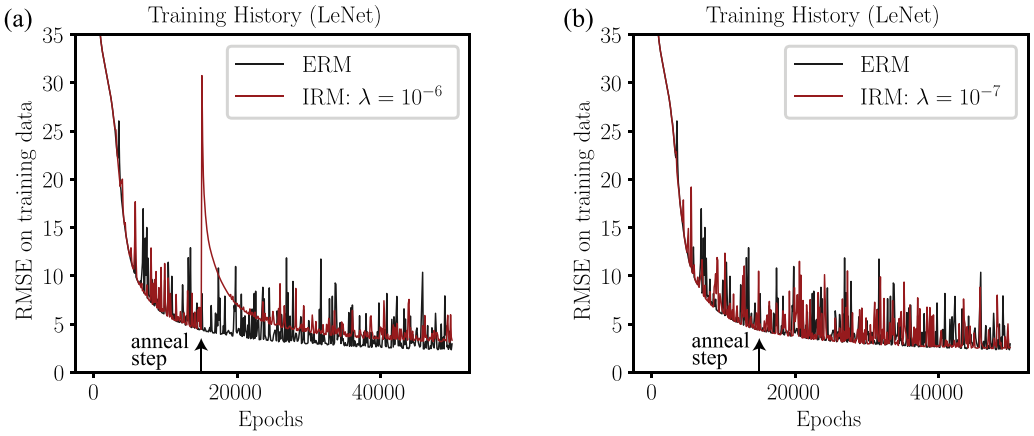

Fig. 5. Example of how penalty weight influences the training error history of a modified LeNet. The penalty weight is formally introduced after the anneal step (15 000). (a) training error history of ERM and IRM with penalty weight is equal to 10^{-6} (b) training error history of ERM and IRM with penalty weight is equal to 10^{-7}.

Fig. 5, which compares the training history of the modified LeNet using ERM and IRM algorithm on the covariate shift dataset defined in Section 2.2. In Fig. 5(a), the penalty weight of IRM before the anneal step (15 000) is 10^{-10}, which is small enough such that the training history is almost the same as ERM. The penalty weight changes to $\lambda = 10^{-6}$ from the anneal step where the training error suddenly increases at step $t = 15000$. However, in Fig. 5, the penalty weight $\lambda = 10^{-7}$ is too small, as the training error after the anneal step still follows the track of ERM, which means that the penalty term does not work. Thus, we ignore the value $\lambda = 10^{-7}$ regardless of the magnitude of the validation error for the model trained with this penalty weight.

The final selection of hyperparameters for all approaches is given in Appendix B.2.

3.3.2. Evaluation metrics

We trained the two models described in Section 3.2 15 times each for each of the four algorithms with initializations defined by seeds from $k = [1, 2, 3 \dots 15]$. We report the results of many initializations to ensure that our conclusions are not based on outlier results obtained due to the randomness of the training process. We note briefly that for each approach, if the lowest training error obtained from an initialization is too large, where large is defined as three times higher than the lowest training error obtained from at least ten other initializations, we re-trained the model with a different initialization not used in the initial group of 15. After that, we tested each model on all test environments and calculated the root mean squared error (RMSE) between the predicted change in strain energy $\hat{y}$ and the ground truth y. We evaluate the effectiveness of each algorithm on every test environment in two ways. First, we report the root mean squared error of the aggregate mean prediction across all 15 models with different initializations defined as:

$$\overline{RMSE} = \sqrt{\left(y - \frac{1}{15}\sum_{k=1}^{15}\hat{y}_k\right)^2}. \tag{13}$$

Second, we report the average of the root mean squared error of all 15 models defined as:

$$\overline{RMSE}' = \frac{1}{15}\sum_{k=1}^{15}\sqrt{\left(y - \hat{y}_k\right)^2}. \tag{14}$$

The RMSE of aggregated mean prediction defined in Eq. (13) is used to evaluate the four methods in Section 3.1 and the results are discussed in the following Section 4. The average of RMSE defined in Eq. (14) is represented in appendix as supplemental materials.

4. Results and discussion

In this Section, we examine the performance of all four algorithms introduced in Section 3.1 on the OOD datasets defined in Section 2. In Section 4.1, we discuss the results for the OOD problem caused by covariate shift. In Section 4.2, we discuss the results for the OOD problem caused by mechanism shift. In Section 4.3, we discuss the results for the OOD problem caused by sampling bias. Finally, in Section 4.4, we discuss common findings across all investigations.

4.1. ML model performance on the covariate shift dataset

In this Section, we evaluate the four different ML methods introduced in Section 3.1 on the two covariate shift datasets from Mechanical MNIST and Mechanical MNIST-EMNIST Letters. The problem definition and details of creating the covariate shift datasets were introduced in Section 2.2. For all four algorithms (ERM, IRM, REx, and IGA), we train both the MLP model and the modified LeNet model separately (see model architectures in Appendix B.1). The performance of the MLP and LeNet model on the training, validation, and testing environments of each covariate dataset are shown in Fig. 6 for each algorithm. Specifically, Fig. 6a shows the performance results for the covariate shift dataset from Mechanical MNIST and Fig. 6b shows the performance results for the covariate shift dataset from Mechanical MNIST-EMNIST Letters. In each Figure, we plot the root mean square error (RMSE) of the aggregate mean prediction described in Eq. (13). The performance of the baseline ERM method is illustrated with black markers, and the performance of the other three OOD algorithms is represented in red hues. For all algorithms, we plot the RMSE of aggregate mean prediction for the training environment, the validation environment, and both test environments.

In Fig. 6a-i, we show the results of training the MLP model with all four algorithms on the covariate shift dataset from Mechanical MNIST. Here, the ERM approach achieves a very low error on the training data (RMSE 2.01) with a slightly higher error on the validation data (RMSE 20.85). For context, in Fig. 6a, the training dataset has mean 559.12, and standard deviation 45.99, and the validation dataset has mean 559.35, and standard deviation 45.74. For a typical ML modeling approach driven by ERM (i.e., traditional ML method), the model performance on the validation dataset is assumed to be very close to the performance of the model on unseen test datasets. However, we note that compared to the validation data, the performance of ERM on both test environments with covariate shift is substantially worse. Specifically, the test error of test environment 1 is RMSE 55.53, and the test error of test environment 2 is RMSE 92.52. For the MLP trained by the three OOD algorithms, test error on both test environments 1 and 2 is lower than ERM. In addition to plotting these values in Fig. 6, they are listed in Table 3 in Appendix C.2. Overall, the lowest test error is obtained by IRM with a validation error of RMSE 16.48, and test error of RMSE 39.85 on test environment 1 and RMSE 44.23 on test environment 2. Through the same evaluation process as the MLP model, the results on the covariate shift dataset from Mechanical MNIST obtained by a modified LeNet model are shown in Fig. 6a-ii. Overall, the performance of all methods improves because of the deeper and more powerful Convolutional Neural Network model architecture. However, consistent with the MLP results, the methods designed for the OOD problems outperform the ERM algorithm. In this case, the best performance is obtained by REx with the training and test error consistently below RMSE 10, with the error on test environment 1 (RMSE 8.77) being almost a third of ERM (RMSE 22.46), while the error on test environment 2 (RMSE 7.94) is more than four times smaller than ERM (RMSE 35.75).

The results of repeating this process on the covariate shift dataset from Mechanical MNIST - EMNIST Letters are shown in Fig. 6b. For context, in Fig. 6b, the training dataset has mean 703.01, and standard deviation 86.26, and the validation dataset has mean 706.98, and standard deviation 88.45. For the evaluation results on MLP in Fig. 6b-i, the ERM method obtained a low error on the validation dataset (RMSE 36.21) but behaved poorly on test error with RMSE 87.24 on test environment 1 and RMSE 167.57 on test environment 2. For the evaluation results on modified LeNet in Fig. 6b-ii, the ERM method again obtained low error on validation dataset (RMSE 28.47) but high test error with RMSE 80.77 on test environment 1 and RMSE 162.48 on test environment 2. We then noticed that for ERM, although the validation error decreased by 21.38% by using a deeper convolutional neural network (LeNet) compared to the simple MLP model, the decrease of test error by using LeNet is very small, as the error drops by only 7.42% for test environment 1 and 3.04% for test environment 2. In contrast to ERM, the OOD algorithms still performed better on test environments for both the MLP model as shown in Fig. 6b-i and the LeNet model as shown in Fig. 6b-ii. The best performance is obtained by IGA with RMSE 52.24 on test environment

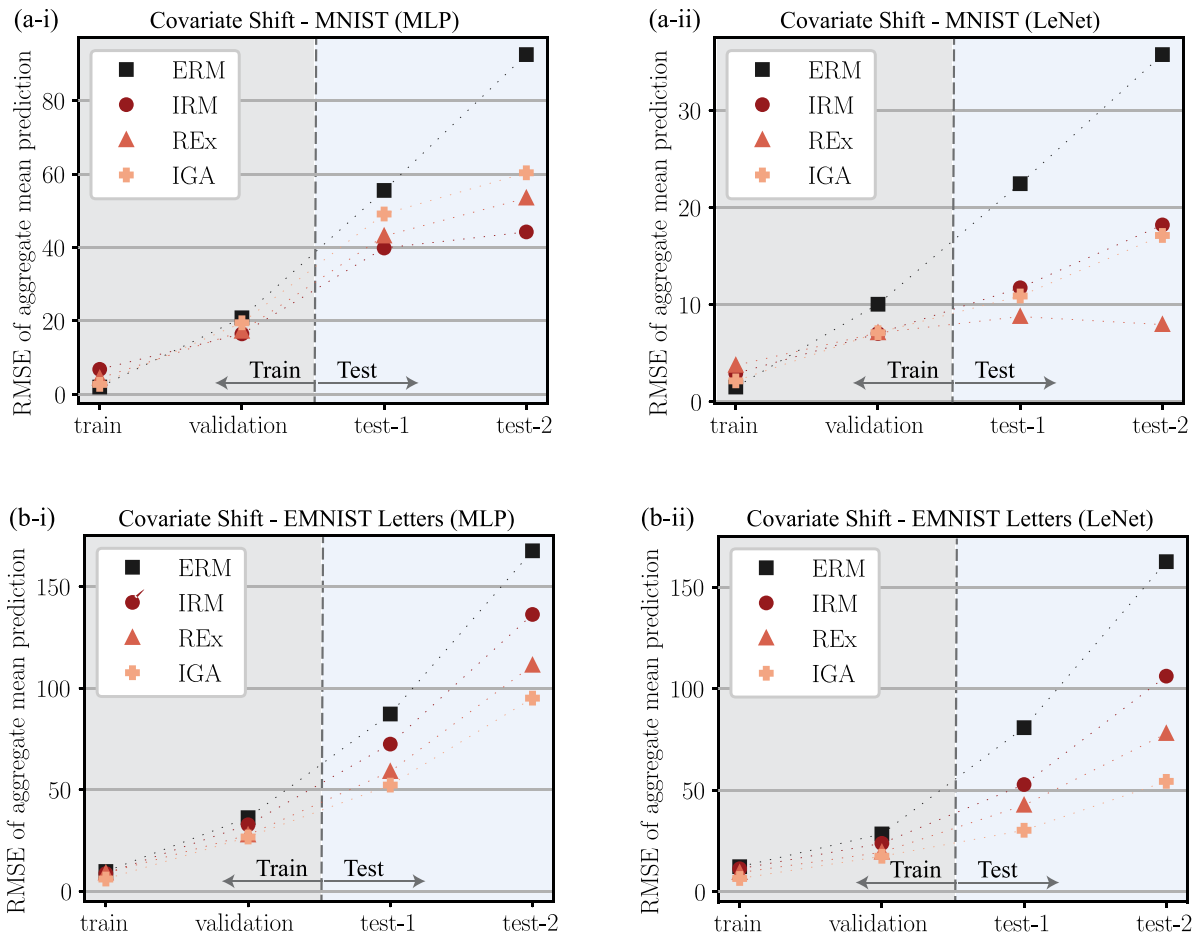

Fig. 6. The performance of four algorithms (ERM, IRM, REx, IGA) on the covariate shift datasets defined in Section 2.2. The RMSE is calculated using Eq. (13). (a) The performance of a MLP model (a-i) and a modified LeNet model(a-ii) trained by four algorithms on training, validation, and testing data from Mechanical MNIST Collection. (b) The performance of a MLP model (b-i) and a modified LeNet model (b-ii) trained by four algorithms on training, validation, and testing data from Mechanical MNIST - EMNIST Letters Collection. Additional visualization of these results can be found in Appendix C.3, which shows a comparison of prediction vs. ground truth, and Appendix C.4 which shows representative samples of different error levels.

1 and RMSE 95.04 on test environment 2 for the MLP model, while its RMSE was 30.27 on test environment 1 and 54.33 on test environment 2 for the modified LeNet model. And compared to ERM, the performance of all OOD algorithms was improved by using a deeper Neural Network. For example, compared to the MLP model, by implementing IGA on the modified LeNet model, the test error decreased by 42.06% on test environment 1 and 42.83% on test environment 2. More statistics about the data and the evaluation results are given in Table 4 in Appendix C.2. Further discussion of these results is also given in Section 4.4.

4.2. ML model performance on the mechanism shift dataset

By using the same evaluation method described in Section 4.1, we evaluate the two ML models on the two mechanism shift datasets from both Mechanical MNIST and Mechanical MNIST-EMNIST Letters. The problem definition and processing details of the mechanism shift datasets was introduced in Section 2.3. Since the training environments in this Section are the same as in Section 4.1, we use the two ML models (MLP and modified LeNet) already trained in Section 4.1 to test the two new test environments with mechanism shift. For all algorithms introduced in Section 3.1, we plot the aggregate mean prediction RMSE for the training environment, the validation

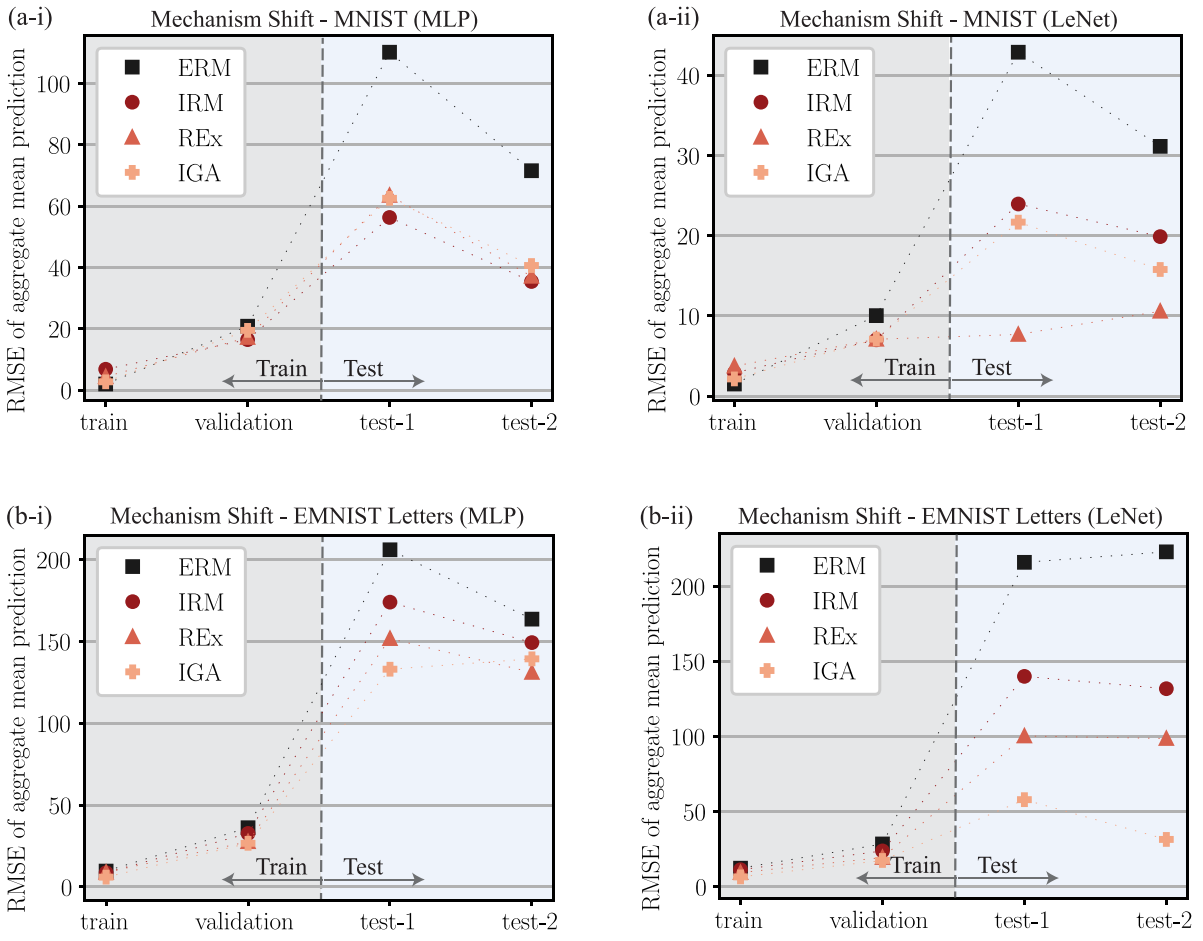

Fig. 7. The performance of four algorithms (ERM, IRM, REx, IGA) on mechanism shift data defined in Section 2.3. The RMSE is calculated using Eq. (13). (a) The performance of a MLP model(a-i) and a modified LeNet model(a-ii) trained by four algorithms on training, validation, and testing data from Mechanical MNIST Collection. (b) The performance of a MLP model(b-i) and a modified LeNet model(b-ii) trained by four algorithms on training, validation, and testing data from Mechanical MNIST - EMNIST Letters Collection. Additional visualization of these results can be found in Appendix C.3, which shows a comparison of prediction vs. ground truth, and Appendix C.4 which shows representative samples of different error levels.

environment, and both test environments in Fig. 7. Specifically, Fig. 7a shows the performance results for the mechanism shift dataset from Mechanical MNIST and Fig. 7b shows the performance results for the mechanism shift dataset from Mechanical MNIST - EMNIST Letters. Similar to the results for the covariate shift datasets in Section 4.1, for both Mechanical MNIST and Mechanical MNIST - EMNIST Letters, the three OOD algorithms perform better than the baseline ERM method on both test environments for both the MLP and LeNet models.

For the mechanism shift data from Mechanical MNIST, Fig. 7a-i shows the performance of the four algorithms using the MLP model. The performance of the three OOD algorithms for a MLP model is similar with the IRM method achieving the lowest test error for both environments 1 (RMSE 56.31) and 2 (RMSE 35.43). In contrast, the test error achieved by ERM is RMSE 110.19 for test environment 1 and RMSE 71.55 for test environment 2. In Fig. 7a-ii, which shows the performance of the four algorithms on the modified LeNet model, the test error of ERM drops significantly to RMSE 42.88 for test environment 1 and RMSE 31.15 for test environment 2. However, these numbers are still much larger than the test error obtained through the OOD algorithms. Specifically, REx achieves the lowest test error for both environments 1 (RMSE 7.70) and 2 (RMSE 10.59). Since the mechanism shift is a more challenging phenomena to capture than covariate shift due to a shift in both the input and output data, we

note that for the ERM, the test error of test environment 1 in this Section is larger than the test error of the two test environments shown in Section 4.1, while the RMSE of REx is similar to that seen in the covariate shift example.

For mechanism shift data from the Mechanical MNIST - EMNIST Letters Collection, Fig. 7b-i shows the performance of the four algorithms using the MLP model. Compared to ERM which gets RMSE 206.02 on test environment 1 and 163.66 on test environment 2, the decrease in test error obtained by the OOD algorithms is still lower where IGA obtained the lowest error (RMSE 133.05) on test environment 1 while REx obtained the lowest error (RMSE 131.10) on test environment 2. Fig. 7b-ii shows the performance of the four algorithms on LeNet model. The performance of all OOD algorithms improved significantly with IGA obtaining the lowest error on both environments (RMSE 57.96 for test environment 1, and RMSE 31.49 for test environment 2) while ERM still obtaining very high test error (RMSE 215.98 on test environment 1 and RMSE 222.95 on test environment 2). Further discussion of these results is given in Section 4.4.

Furthermore, the test error of test environment 2 in this Section drops slightly compared to test environment 1 error for both mechanism shift datasets. We note that for mechanism shift datasets from both Mechanical MNIST and Mechanical MNIST - EMNIST Letters, the output standard deviation of test environment 2 is slightly smaller than the output standard deviation of the test environment 1 (see Tables 5 and 6), which can cause test environment 2 to get a smaller RMSE. Taking the mechanism shift dataset from Mechanical MNIST Collection as an example, the output standard deviation of test environment 2 is 31.37 while the output standard deviation of the test environment 1 is 37.94, as given in Table 5.

4.3. ML model performance on the sampling bias dataset

In this Section, we evaluate the four algorithms described in Section 3.1 on OOD problems caused by sampling bias as described in Section 2.4. Similar to Sections 4.1 and 4.2, we plot the RMSE of aggregate mean prediction for the training environment, the validation environment, and all test environments on Fig. 8. The performance of the four algorithms on the sampling bias data from Mechanical MNIST is shown in Fig. 8a. The performance of ERM and IRM on all environments is very close for both the MLP and LeNet models. The REx method obtained lower test error than ERM for the MLP model, but unlike the covariate shift and mechanism shift examples, achieves no improvement for a LeNet model. The IGA method is the only algorithm that consistently performs better than ERM on test environments, though the improvements are relatively small. Specifically, for the MLP model, ERM achieved a RMSE of 29.54 on test environment 1, a RMSE of 26.63 on test environment 2 and a RMSE of 29.21 on test environment 3, while IGA achieved a RMSE of 16.25 on test environment 1, a RMSE of 15.91 on test environment 2 and a RMSE of 18.24 on test environment 3. For the LeNet model, ERM achieved a RMSE of 10.97 on test environment 1, a RMSE of 10.89 on test environment 2, and a RMSE of 13.22 on test environment 3. In contrast, IGA achieved a RMSE of 10.28 on test environment 1, a RMSE of 10.08 on test environment 2, and a RMSE of 11.74 on test environment 3. Although IGA performs better than ERM on test environments for a LeNet model, we note that the training and validation error (RMSE 6.21) of IGA was slightly larger than the validation error (RMSE 5.41) for ERM. Critically, this demonstrates that a lower training or validation error does not necessarily mean a lower test error if there is a strong spurious correlation present in the training environments.

The performance of the four algorithms on the sampling bias data from Mechanical MNIST - EMNIST Letters is shown in Fig. 8b. For the performance on the MLP model in Fig. 8b-i, all three OOD algorithms obtained lower test error than the ERM, with IGA obtaining the lowest test error where the RMSE is 26.91 for test environment 1, 27.08 for test environment 2, and 33.28 for test environment 3. In contrast, ERM obtained a RMSE 36.48 for test environment 1, 36.69 for test environment 2, and 42.42 for test environment 3. However for the performance on LeNet model, as shown in Fig. 8b-i, the performance of the three OOD algorithms is similar to ERM, with no improvement observed for the three test environments. We will further discuss these results in the next Section.

4.4. Common findings across all datasets

Sections 4.1–4.3 show the results of all the OOD experiments that we have conducted. Across all experiments, the error on OOD test data using the traditional ERM method (not designed with OOD problems in mind) is much higher than the error that would be predicted based on the validation data. This indicates that models trained by ERM are vulnerable to suffering poor performance when exposed to OOD test data. Thus, even if a ML model

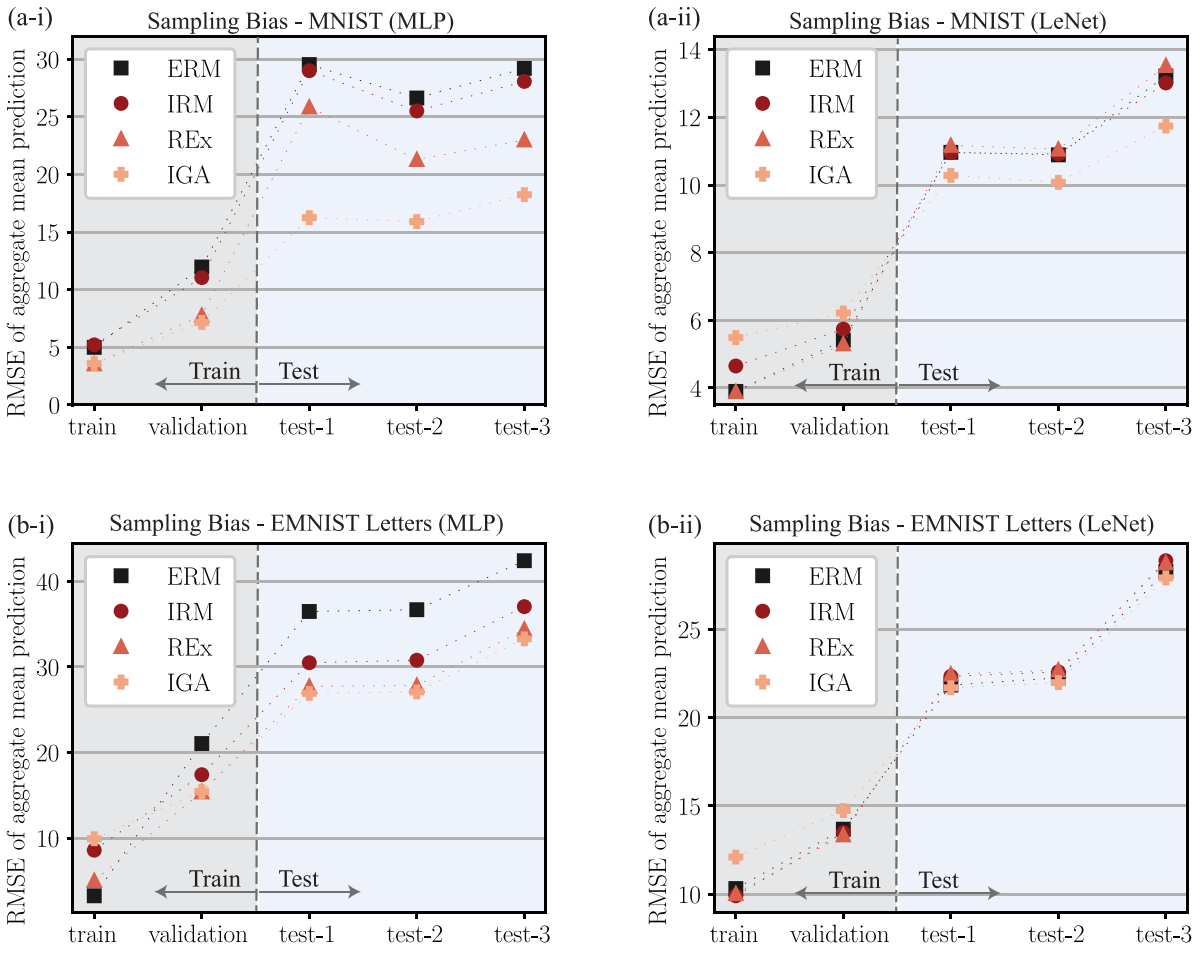

Fig. 8. The performance of four algorithms (ERM, IRM, REx, IGA) on the sampling bias data defined in Section 2.4. The RMSE is calculated using Eq. (13). (a) The performance of a MLP model (a-i) and a modified LeNet model (a-ii) trained by four algorithms on training, validation, and testing data from Mechanical MNIST Collection. (b) The performance of a MLP model (b-i) and a modified LeNet model (b-ii) trained by four algorithms on training, validation, and testing data from Mechanical MNIST - EMNIST Letters Collection. Additional visualization of these results can be found in Appendix C.3, which shows a comparison of prediction vs. ground truth, and Appendix C.4 which shows representative samples of different error levels.

obtains a low error for validation data that is i.i.d with respect to the training data, the test error can be very high when it is applied to practical test situations when the test data distribution may be shifted. In contrast, we found that the OOD generalization methods were effective in reducing the test error on OOD test data, in particular for the covariate shift and mechanism shift challenge problems. However, the effectiveness of the OOD algorithms in decreasing the test error on sampling bias data was small, in particular when using a deeper ML (LeNet) model. This makes the sampling bias problem that introduces spurious correlations the most challenging out of the three OOD problems in mechanics that we have investigated.

Secondly, we also found that although the OOD generalization methods try to find an invariant predictor enabling consistent performance across all environments, the test error is still much larger for all of the OOD experiments we have conducted in comparison to the training and validation error. One potential reason is that we tune the hyperparameters on the validation dataset, which is still collected from the training environments. Thus we introduced bias in the model selection in terms of hyperparameter tuning. This means that while our model selection method of tuning the network hyperparameters on the validation dataset can guarantee that the penalty weight we select is within a reasonable regime (i.e., sufficiently large such that it does impact the training process, but not too large such that it jeopardizes convergence), the selected penalty weight is not guaranteed to be the optimal penalty

weight for each OOD dataset or for each algorithm. As a result of this, there is variability in performance for each OOD algorithm on the different OOD problems. Specifically, we do not see one OOD algorithm consistently outperforming or underperforming the others.

Another possible reason why these OOD algorithms cannot obtain a test error as low as the training error is that the "invariance" term introduced by these algorithms is not sufficient for a ML model to learn all factors contributing to the invariance across environments. Thus, while some biased solutions can be avoided during the training process, the penalty introduced by these algorithms is not formulated such that a ML model will find the causal solution exclusively. This implies that there may need to be a re-evaluation of how invariance is defined in this context in order to make the OOD methods more generalizable for problems in mechanics. Specifically, assumptions on the "invariance" of each OOD generalization method should be more clear on how they are related to each type of OOD problem. This development would be critical to confidently solving OOD problems in mechanics, like predicting the mechanical behavior of materials.

Third, we found that the test error for the covariate shift and mechanism shift datasets from the Mechanical MNIST - EMNIST Letters dataset is much larger than from the standard Mechanical MNIST examples. This can be attributed to the larger mean and the standard deviation of the covariate dataset from Mechanical MNIST - EMNIST Letters than the covariate shift dataset from Mechanical MNIST. For example, for the covariate shift data from the Mechanical MNIST Collection, the standard deviation of the target property, i.e., the change in strain energy, is 48.17 in test environment 1 and 44.90 in test environment 2, while for the covariate shift data from Mechanical EMNIST - Letters, the standard deviation of the same target property is 86.30 in test environment 1 and 79.58 in test environment 2, both of which are about twice the standard deviation value in covariate shift data from Mechanical MNIST Collection. As a result, the RMSE of results on Mechanical MNIST - EMNIST Letters is also about twice larger than the RMSE of results on Mechanical MNIST Collection. More statistics of each dataset and the details of evaluation results can be found in Appendix C, Tables 3 and 4. In addition, the data distribution the original MNIST and EMNIST Letters are not the same, which does not assure that the model trained on these two datasets can get the same performance.

Fourth, our objective is to assess these OOD algorithms not only by their performance on different types of OOD problems, but also by their demand for computational resources and feasibility of implementation. To this end, we note that the penalty term of REx is the only one that does not require the computation of the gradient of risks during the training process. Thus its applicability is not restricted by computational capacity and can be broadly applied to deeper and more complex ML models. Since the performance of REx on OOD test data was consistently better using the deeper LeNet model, this may imply that REx is a better choice than the other two OOD algorithms for large and complex datasets that require deeper ML architectures for good performance. In contrast, the IGA method is more computationally intensive because it requires computing the risk towards all the weights of a Neural Network. However, the performance of IGA on the three OOD problems that we investigated is overall the best as it outperforms the other two OOD algorithms in most scenarios. Thus, IGA may be the most appropriate choice when the problem of interest involves a small dataset without requiring a complex ML model. In contrast to IGA, the IRM method needs to compute the gradient of risk for only the weight of the last layer of a Neural Network, which means it requires less computational capacity than IGA. However, because its performance was generally worse than REx and IGA for the three OOD examples that we investigated, IGA and REx appear to be better choices for the OOD problems studied here.

Finally, we note that compared to the performance of these OOD algorithms on classification problems in which they were shown to have similar accuracy on test data and training data [49,89,90], the test error obtained by these OOD algorithms for the regression problems in mechanics is still significantly higher than their validation error. This is reasonable, because as compared to a classification problem with binary outputs of either 1 (belong to this class) or 0 (does not belong to this class) in both training and test datasets, the output of regression problems is usually continuous and can shift to very different value ranges than the values in the training dataset. This characteristic of regression problems makes capturing shifts in the data more challenging than in classification. Therefore, as mechanics problems typically require regression, and because of the many different types of OOD problems that are possible, this may require further development of OOD methods that can robustly handle multiple types of OOD shifts in order to solve OOD regression problems in mechanics.

5. Conclusion

In this paper, we have systematically investigated OOD generalization problems in mechanics by identifying three new challenge problems with test data distribution shifts: covariate shift, mechanism shift, and sampling bias. To study these problems, we created two OOD benchmark datasets for each of the three challenge problems based on the Mechanical MNIST and Mechanical MNIST – EMNIST datasets. We then independently trained two types of ML models, a multilayer perceptron (MLP) and a convolutional neural network (modified LeNet), on these datasets with four different risk minimization algorithms. One algorithm, the classical Empirical Risk Minimization (ERM) served as a baseline algorithm for comparison with three other popular methods specially designed for solving OOD generalization problems, Invariant Risk Minimization (IRM), Risk Extrapolation (REx) and Inter Gradient Alignment (IGA). Through evaluating the performance of these algorithms, we found that OOD generalization methods typically outperform ERM by achieving a lower predicting error on OOD test data while still maintaining good performance on training data. And, all algorithms tended to work better when paired with the more complex LeNet model than when paired with the simpler MLP. However, no algorithm had consistent top performance across all six OOD generalization problem datasets. And, while these OOD generalization algorithms have been reported in the literature to achieve near consistent performance on training and test environments for OOD classification problems [49,89,90], for the OOD regression problems in mechanics covered in this paper, there are only a few cases where these OOD generalization algorithms achieved comparable performance on both training and testing environments. In most cases, the error in the test environments was much higher than the training and validation errors. These results suggest that there is a need for methods to make these ML models more robust so that they can generalize invariance to multiple OOD scenarios.

We note that beyond the three kinds of OOD problems considered in this paper (i.e., covariate shift, mechanism shift, and sampling bias), there are additional factors that can cause poor generalization of ML models for problems in mechanics. For example, OOD problems also exist if the scope of the training data does not cover the whole landscape of the data distribution, i.e. the scenario where new physics emerge due to the shifts of the landscape in the testing environments. Examples of this include different flow behavior for low and high Reynolds numbers [116], or linear vs. nonlinear mechanical response of soft tissues in the small and large strain regimes [117]. In these situations, it is difficult for traditional ML methods to predict the new physics and handle the OOD problem simply by learning from the training data. Because the OOD methods explored in this paper have only been tested on covariate shift, mechanism shift, and sampling bias, OOD problems driven by the emergence of fundamentally different physical regimes are beyond the scope of this paper, and represent a promising avenue for future work.

Overall, this paper provides a critical first evaluation of OOD ML methods in mechanics. We anticipate that the benchmark regression datasets that we have created for OOD problems in mechanics can accelerate the study of OOD generalization problems in the context of regression. And, our work of examining OOD generalization problems in mechanics is a critical step towards applying ML methods to practical problems in real world mechanics. Looking forward, we anticipate that defining OOD generalization methods that are specific to problems in mechanics will be an important direction of future research. Furthermore, it is very important to establish a standard way of selecting hyperparameters when developing new OOD generalization methods. Finally, we note that in this paper we assume that the environment label of each dataset is known. Future investigation should consider alternative methods that have been proposed to divide the training data into different environments as a preprocess step [114,118]. Broadly speaking, the methods and results presented in this paper are a starting point for future work in OOD generalization tasks in mechanics.

6. Additional information

The extensions of Mechanical MNIST data are available through the OpenBU Institutional Repository at https://open.bu.edu/handle/2144/39371. The access to all the OOD datasets is at https://open.bu.edu/handle/2144/44485. The code to reproduce equibiaxial simulation on FEniCS is at https://github.com/elejeune11/Mechanical-MNIST/blob/master/generate_dataset/Equibiaxial_Extension_FEA_test_FEniCS.py The code for implementing the four

algorithms introduced in Section 3.1 and for creating the datasets is available at https://github.com/lingxiaoyuan/ood_mechanics

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

All data is available on publicly-accessible websites.

Acknowledgments

All authors gratefully acknowledge the support of the College of Engineering and Department of Mechanical Engineering at Boston University, United States of America. EL acknowledges the support of the David R. Dalton Career Development Professorship, United States of America, the Hariri Institute Junior Faculty Fellowship, United States of America, and the Office of Naval Research Award, United States of America N00014-22-1-2066. We would like to thank the staff of the Boston University Research Computing Services and the OpenBU Institutional Repository (in particular Eleni Castro) for their invaluable assistance with generating and disseminating the "Mechanical MNIST – Distribution Shift" dataset.

Appendix A. Illustration of ood datasets based on mechanical MNIST - EMNIST letters

In Section 2, we visualize the different OOD training and test environments created based on the Mechanical MNIST dataset in Figs. 1–3. In order to better evaluate the efficacy of the different algorithms designed for OOD data, we also created similar OOD training and test environments based on the Mechanical MNSIT – EMNIST Letters dataset, introduced in Section 2.5. Throughout Sections 4.1–4.3, we report the results based on both MNIST and EMNIST Letters. Here, in Figs. 9–11, we visualize the EMNIST Letters data. Specifically, Fig. 9 illustrates the OOD data distribution for covariate shift, Fig. 10 illustrates the OOD data distribution for mechanism shift, and Fig. 11 illustrates the OOD data distribution for sampling bias.

Appendix B. Additional information for methods introduced in Section 3

B.1. Details of ML models

In Section 3.2, we describe the two ML model architectures used for evaluating the different OOD algorithms. Here, we present schematic illustrations of the MLP model and the modified LeNet model that are introduced in Section 3.2 and trained with the algorithms introduced in Section 3.1. As Fig. 12 shows, the MLP model is a feedforward Neural Network composed of forward fully connected layers and ReLU activation layers. The LeNet model is a Convolutional Neural Network composed of convolutional layers, maxpooling layers and ReLU activation layers [101].

B.2. Details of hyperparameters

For all ML model training, the learning rate is fixed at 0.001, the total number of training epochs is fixed at 50 001, and for each epoch, each model was trained with one single batch with all training data, this information is listed in Table 1. The final selection of penalty weight λ and anneal step t (count from zero) for the two ML models using the three OOD generalization algorithms(see Section 3.1) is based on the approach for hyperparameters tuning introduced in Section 3.3.1. These values are listed in Table 2.

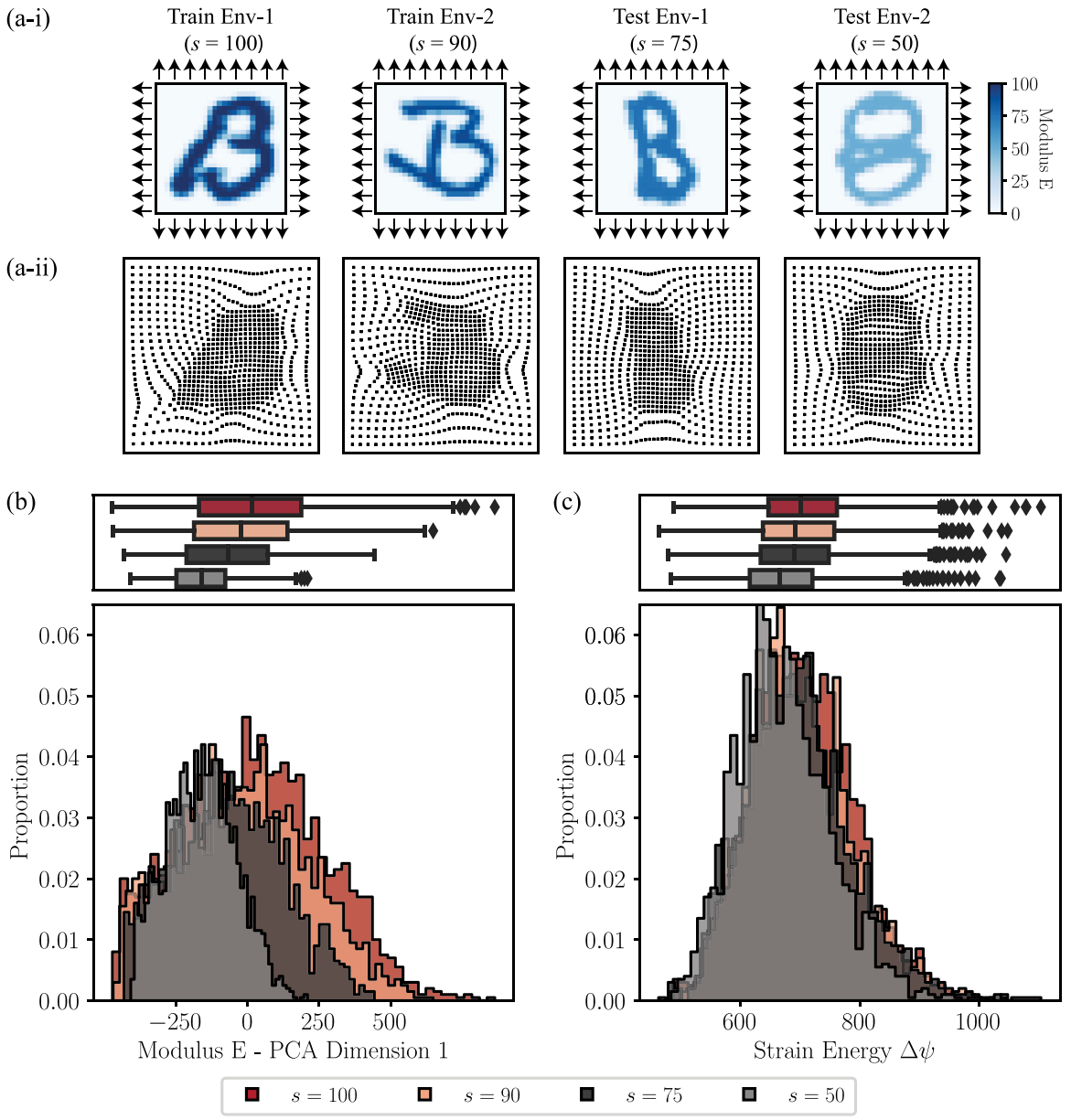

Fig. 9. Illustration of the EMNIST covariate shift dataset. (a-i) Equibiaxial extension boundary conditions and elastic modulus distribution for a representative example from each environment. (a-ii) Deformation of each example in (a-i) after the completion of the equibiaxial extension simulations. (b) Input distribution of all environments described by the coefficient of the first principle component obtained through PCA performed on the input elastic modulus distribution of training data. (c) Output distribution of all environments defined as the total change in strain energy of the domain. Note that in (b–c) the histograms and boxplots are two ways of showing the same data.

Table 1
Training hyperparameters.

	Learning rate	Epochs	Batch size
MLP	0.001	50 001	Training data size
LeNet	0.001	50 001	Training data size

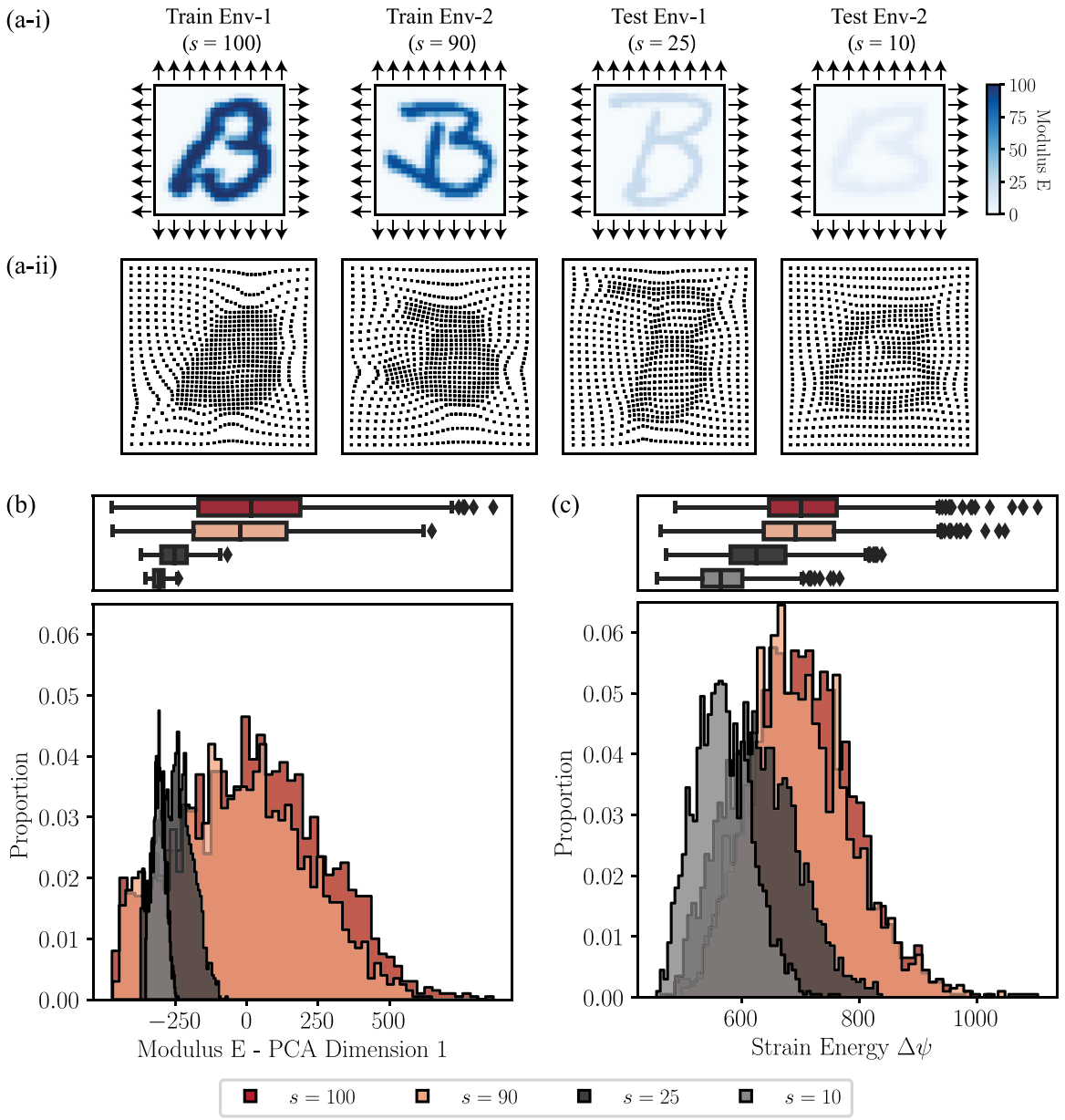

Fig. 10. Illustration of the EMNIST mechanism shift dataset. (a-i) Equibiaxial extension boundary conditions and elastic modulus distribution for a representative example from each environment. (a-ii) Deformation of each example in (a-i) after the completion of the equibiaxial extension simulations. (b) Input distribution of all environments described by the coefficient of the first principle component obtained through PCA performed on the input elastic modulus distribution of training data. (c) Output distribution of all environments defined as the total change in strain energy of the domain. Note that in (b–c) the histograms and boxplots are two ways of showing the same data.

Appendix C. Additional figures and tables to support the results presented in Section 4

C.1. Violin plots

Here we provide supporting information for the evaluation results shown in Section 4. In Section 4, only the aggregated mean prediction defined by Eq. (13) is shown in Figs. 6–8 to ensure a clear comparison between the

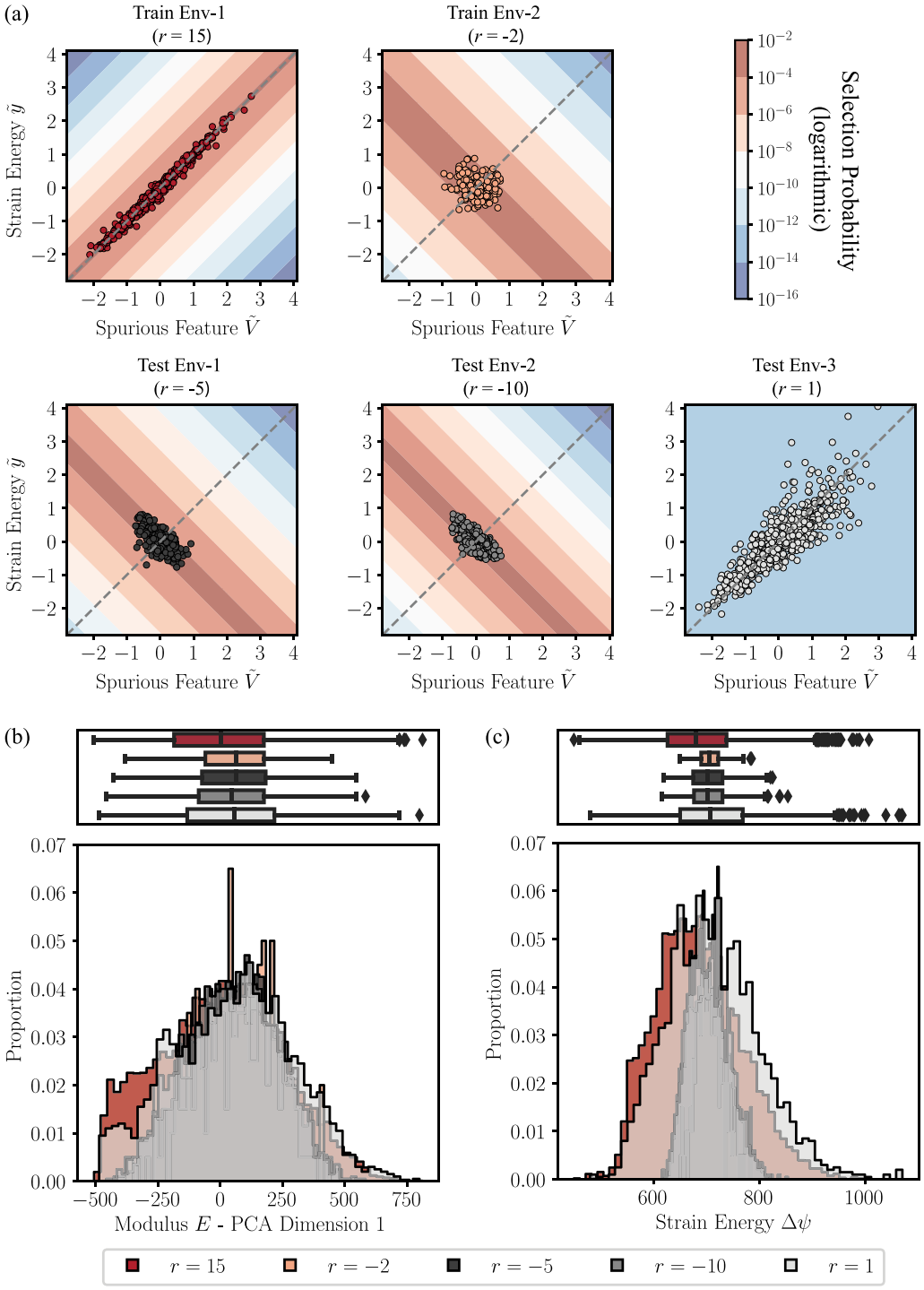

Fig. 11. Illustration of the EMNIST sampling bias datasets. (a) Relationship between the spurious feature and the output target (the total change in strain energy) for each environment. Note that we only show the 500 randomly selected data points for each environment (with the exception of the 200 available data points for training environment 2) to aid in visualization. The color of each plot background represents the selection probability of each area (b) Input distribution of all environments described by the coefficient of the first principle component obtained through PCA performed on the input elastic modulus distribution of training data. (c) Output distribution of all environments defined as the total change in strain energy of the domain. Note that in (b–c) the histograms and boxplots are two ways of showing the same data. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

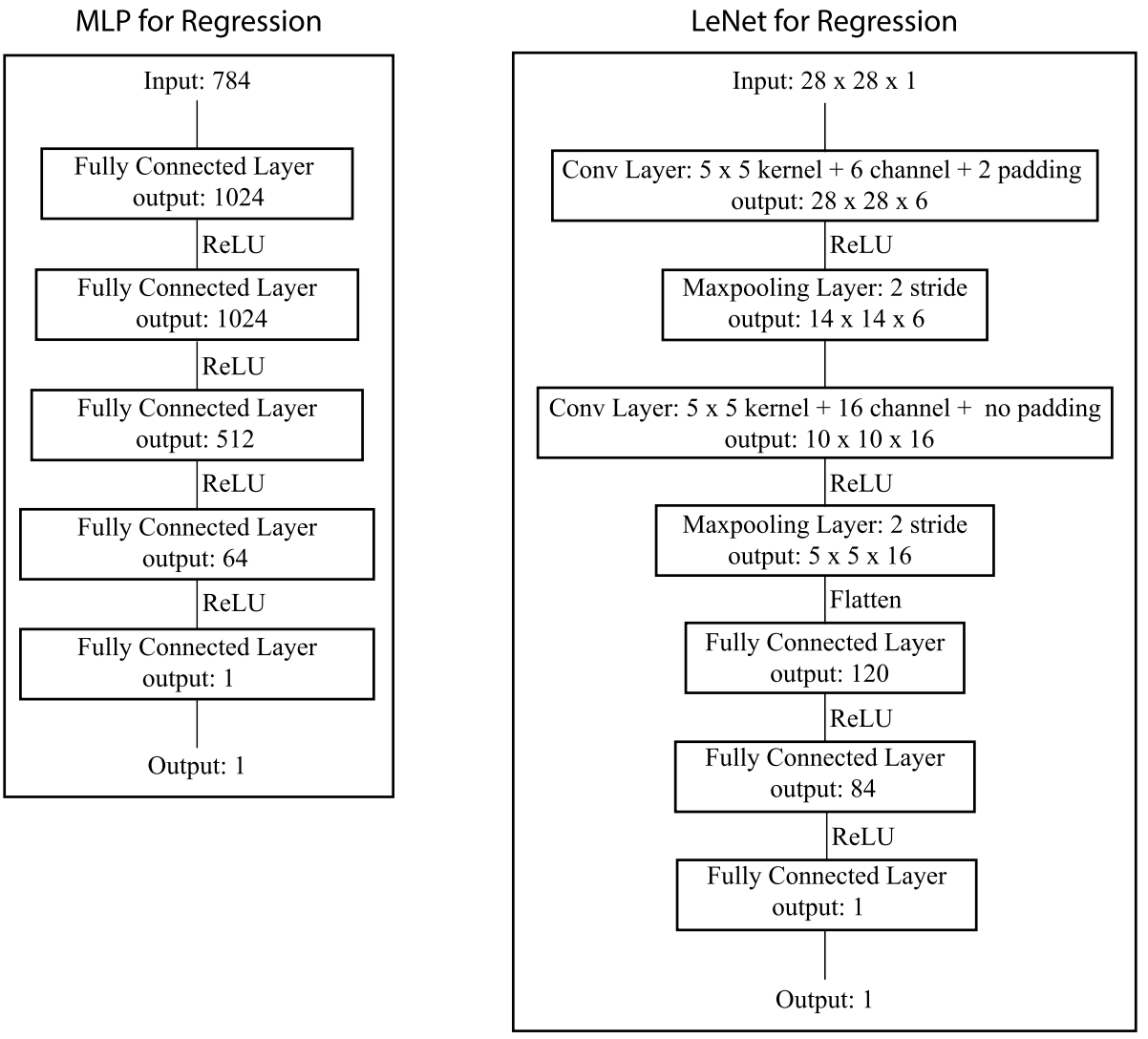

Fig. 12. Illustration of model structures of the MLP model and the modified LeNet introduced in Section 3.2.

Table 2
Hyperparameter selection results where λ is the penalty weight for each OOD generalization method (IRM, REx, IGA), and t is the anneal step (or epoch) after which the penalty weight will be introduced during training.

Source data		Mechanical MNIST				Mechanical MNIST - EMNIST letters			
OOD problems		Covariate shift mechanism shift		Sampling bias		Covariate shift mechanism shift		Sampling bias	
Hyperparameters		λ	t	λ	t	λ	t	λ	t
MLP	IRM	1e−5	15 000	1e−6	10 000	1e−6	15 000	1e−5	10 000
MLP	REx	1e−1	15 000	1e−1	10 000	1e−2	15 000	1e−1	10 000
MLP	IGA	1e−2	15 000	1e−1	10 000	1e−2	15 000	1e0	10 000
LeNet	IRM	1e−6	15 000	1e−5	15 000	1e−5	15 000	1e−6	15 000
LeNet	REx	1e0	15 000	1e−1	15 000	1e−2	15 000	1e−1	15 000
LeNet	IGA	1e−4	15 000	1e−1	15 000	1e−5	15 000	1e−3	15 000

Table 3
The performance of four algorithms (ERM, IRM, REx, IGA) on the **covariate shift** data from the **Mechanical MNIST Collection**. The RMSE on each dataset is calculated using both Eqs. (13) and (14) based on 15 predictions given by the corresponding ML model trained with 15 different initialization. In addition, the mean and standard deviation for the change in strain energy from the train, validation, and test datasets are also given.

Data		Train		Valid		Test-1		Test-2	
Statistics of strain energy		y_mean = 559.12 y_std = 45.99		y_mean = 559.35 y_std = 45.74		y_mean = 575.96 y_std = 48.17		y_mean = 562.21 y_std = 44.90	
Evaluation method		RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))
MLP	ERM	2.01	2.77	20.85	26.60	55.53	61.50	92.52	95.04
MLP	IRM	6.82	8.51	16.48	19.54	39.85	41.40	44.23	45.07
MLP	REx	4.39	6.45	17.17	21.16	43.11	45.60	53.44	54.55
MLP	IGA	2.72	3.87	19.50	22.08	49.11	50.34	60.33	60.98
LeNet	ERM	1.51	3.01	10.04	15.72	22.46	25.85	35.75	38.69
LeNet	IRM	2.91	5.04	7.03	10.78	11.74	14.55	18.21	21.40
LeNet	REx	3.75	6.11	7.10	10.94	8.77	12.82	7.94	15.13
LeNet	IGA	2.15	4.09	7.11	11.52	10.89	14.55	17.12	21.61

Table 4
The performance of four algorithms (ERM, IRM, REx, IGA) on the **covariate shift** data from the **Mechanical MNIST - EMNIST Letters Collection**. The RMSE on each dataset is calculated using both Eqs. (13) and (14) based on 15 predictions given by the corresponding ML model trained with 15 different initialization. In addition, the mean and standard deviation for the change in strain energy from the train, validation, and test datasets are also given.

Data		Train		Valid		Test-1		Test-2	
Statistics of strain energy		y_mean = 703.01 y_std = 86.26		y_mean = 706.98 y_std = 88.45		y_mean = 695.56 y_std = 86.30		y_mean = 671.97 y_std = 79.58	
Evaluation method		RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))
MLP	ERM	9.72	12.40	36.21	43.34	87.24	90.10	167.57	168.80
MLP	IRM	8.70	13.35	32.72	44.84	72.41	77.36	136.24	138.15
MLP	REx	8.94	12.70	27.67	36.51	58.89	63.02	111.22	113.05
MLP	IGA	6.01	8.89	26.68	33.14	52.24	55.32	95.04	96.73
LeNet	ERM	12.33	16.89	28.47	39.44	80.77	84.93	162.48	164.70
LeNet	IRM	11.32	20.10	23.84	39.84	52.78	60.54	106.13	111.77
LeNet	REx	9.41	16.34	19.45	30.85	42.58	50.08	77.88	85.92
LeNet	IGA	6.54	12.60	17.47	30.45	30.27	40.41	54.33	65.97

different environments and approaches. Here, we present identical results in the more detailed form of violin plots. In each violin plot, there are 15 white points which represent the RMSE performance of the MLP model (or the modified LeNet model) with 15 different initialization, and one color-filled point which represents the aggregated mean prediction calculated through Eq. (13) based on the predicting quantity of interest (i.e., the change in strain energy after a equibiaxial extension) value given by these 15 models. Specifically, Fig. 13 shows the performance of all algorithms (ERM, IRM, REx and IGA) on the training, validation, and test environments from the covariate shift dataset created with the methods described in Section 2.2 on both Mechanical MNIST (Fig. 13a) and Mechanical EMNIST-Letters (Fig. 13b). In Fig. 13a, we show a violin plot for each of the four algorithms implemented on the MLP model (Fig. 13a-i), and the modified LeNet model (Fig. 13a-ii). Similarly, we plot the evaluation results of all the algorithms on the covariate shift dataset from Mechanical EMNIST-Letters through violin plots in Fig. 13b. Following the same format, Fig. 14 shows the performance of all algorithms on the mechanism shift dataset described in Section 2.3 for both Mechanical MNIST (Fig. 14a) and Mechanical EMNIST-Letters (Fig. 14b).

Table 5
The performance of four algorithms (ERM, IRM, REx, IGA) on the **mechanism shift** data from the **Mechanical MNIST Collection**. The RMSE on each dataset is calculated using both Eqs. (13) and (14) based on 15 predictions given by the corresponding ML model trained with 15 different initialization. In addition, the mean and standard deviation for the change in strain energy from the train, validation, and test datasets are also given.

Data		Train		Valid		Test-1		Test-2	
Statistics of strain energy		y_mean = 559.12 y_std = 45.99		y_mean = 559.35 y_std = 45.74		y_mean = 543.16 y_std = 37.94		y_mean = 504.45 y_std = 31.37	
Evaluation method		RMSE (Eq. (10))	RMSE (Eq. (11))	RMSE (Eq. (10))	RMSE (Eq. (11))	RMSE (Eq. (10))	RMSE (Eq. (11))	RMSE (Eq. (10))	RMSE (Eq. (11))
MLP	ERM	2.01	2.77	20.85	26.60	110.19	112.06	71.55	76.02
MLP	IRM	6.82	8.51	16.48	19.54	56.31	58.13	35.43	64.90
MLP	REx	4.39	6.45	17.17	21.16	63.50	65.08	37.13	58.78
MLP	IGA	2.72	3.87	19.50	22.08	62.55	63.36	40.64	42.85
LeNet	ERM	1.51	3.01	10.04	15.72	42.88	45.93	31.15	38.28
LeNet	IRM	2.91	5.04	7.03	10.78	23.96	31.38	19.89	32.28
LeNet	REx	3.75	6.11	7.10	10.94	7.70	23.12	10.59	28.94
LeNet	IGA	2.15	4.09	7.11	11.52	21.70	31.23	15.79	32.66

Table 6
The performance of four algorithms (ERM, IRM, REx, IGA) on the **mechanism shift** data from the **Mechanical MNIST - EMNIST Letters Collection**. The RMSE on each dataset is calculated using both Eqs. (13) and (14) based on 15 predictions given by the corresponding ML model trained with 15 different initialization. In addition, the mean and standard deviation for the change in strain energy from the train, validation, and test datasets are also given.

Data		Train		Valid		Test-1		Test-2	
Statistics of strain energy		y_mean = 703.01 y_std = 86.26		y_mean = 706.98 y_std = 88.45		y_mean = 629.56 y_std = 65.91		y_mean = 567.98 y_std = 48.67	
Evaluation method		RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))
MLP	ERM	9.72	12.40	36.21	43.34	206.02	207.21	163.66	165.31
MLP	IRM	8.70	13.35	32.72	44.84	174.03	175.42	149.32	153.51
MLP	REx	8.94	12.70	27.67	36.51	151.76	153.21	131.10	134.32
MLP	IGA	6.01	8.89	26.68	33.14	133.05	134.75	139.39	140.81
LeNet	ERM	12.33	16.89	28.47	39.44	215.98	217.78	222.95	224.47
LeNet	IRM	11.32	20.10	23.84	39.84	140.05	147.29	131.84	141.74
LeNet	REx	9.41	16.34	19.45	30.85	100.36	111.68	98.68	114.34
LeNet	IGA	6.54	12.60	17.47	30.45	57.96	115.96	31.49	161.78

And, Fig. 15 shows the performance of all algorithms on the sampling bias dataset described in Section 2.4 for both Mechanical MNIST (Fig. 14a) and Mechanical EMNIST-Letters (Fig. 14b).

C.2. Results tables

Tables 3–8 present the RMSE performance calculated by both Eqs. (13) and (14) for all methods introduced in Section 3.1 on all OOD datasets introduced in Section 2. Note that in these tables, the RMSE calculated by Eq. (13) corresponds to the RMSE performance presented and discussed in Section 4. In addition, for each table, we also show the mean and the standard deviation of the change in strain energy for each group of data: training, validation and testing datasets. Note that typically datasets with larger standard deviation of strain energy tends to correspond to a larger RMSE which makes the RMSE from Mechanical MNIST-EMNIST Letters generally larger than the RMSE from Mechanical MNIST. Tables 3 and 4 show the RMSE performance for covariate shift datasets based on Mechanical MNIST and Mechanical MNIST-EMNIST Letters respectively. Tables 5 and 6 show the RMSE

Table 7
The performance of four algorithms (ERM, IRM, REx, IGA) on the **sampling bias** data from the **Mechanical MNIST Collection**. The RMSE on each dataset is calculated using both Eqs. (13) and (14) based on 15 predictions given by the corresponding ML model trained with 15 different initialization. In addition, the mean and standard deviation for the change in strain energy from the train, validation, and test datasets are also given.

Data		Train		Valid		Test-1		Test-2		Test-3	
Statistics of strain energy		y_mean = 545.99 y_std = 42.55		y_mean = 545.99 y_std = 41.23		y_mean = 565.68 y_std = 22.45		y_mean = 564.82 y_std = 22.27		y_mean = 567.02 y_std = 47.57	
Evaluation method		RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))
MLP	ERM	4.99	5.16	11.96	13.36	29.54	30.38	26.63	27.53	29.21	30.05
MLP	IRM	5.20	5.28	11.05	12.56	29.00	29.92	25.51	26.51	28.08	29.05
MLP	REx	3.55	5.00	7.77	9.39	25.85	26.90	21.29	22.38	22.99	24.17
MLP	IGA	3.58	4.77	7.15	9.59	16.25	20.39	15.91	19.05	18.24	21.31
LeNet	ERM	3.90	4.59	5.41	6.10	10.97	11.42	10.89	11.33	13.22	13.60
LeNet	IRM	4.65	5.30	5.74	6.36	10.96	11.35	10.92	11.32	13.02	13.37
LeNet	REx	3.89	4.56	5.30	5.91	11.17	11.54	11.06	11.42	13.53	13.85
LeNet	IGA	5.49	6.73	6.21	7.41	10.28	11.16	10.08	10.99	11.74	12.59

Table 8
The performance of four algorithms (ERM, IRM, REx, IGA) on the **sampling bias data** from the **Mechanical MNIST - EMNIST Letters Collection**. The RMSE on each dataset is calculated using both Eqs. (13) and (14) based on 15 predictions given by the corresponding ML model trained with 15 different initialization. In addition, the mean and standard deviation for the change in strain energy from the train, validation, and test datasets are also given.

Data		Train		Valid		Test-1		Test-2		Test-3	
Statistics of strain energy		y_mean = 686.22 y_std = 80.05		y_mean = 686.08 y_std = 80.41		y_mean = 704.81 y_std = 37.24		y_mean = 705.21 y_std = 37.75		y_mean = 713.32 y_std = 87.12	
Evaluation method		RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))	RMSE (Eq. (13))	RMSE (Eq. (14))
MLP	ERM	3.25	3.99	21.05	23.21	36.48	38.01	36.69	38.24	42.42	43.70
MLP	IRM	8.58	10.83	17.40	20.15	30.49	32.43	30.78	32.77	37.05	38.83
MLP	REx	5.03	6.51	15.39	17.65	27.73	29.35	27.83	29.56	34.42	35.85
MLP	IGA	9.92	12.59	15.42	18.24	26.91	28.88	27.08	29.46	33.28	35.40
LeNet	ERM	10.32	12.26	13.68	15.64	21.84	23.24	22.25	23.67	28.27	29.43
LeNet	IRM	9.91	11.51	13.48	15.03	22.34	23.41	22.59	23.67	28.89	29.72
LeNet	REx	10.03	11.82	13.36	14.96	22.46	23.54	22.69	23.79	28.76	29.67
LeNet	IGA	12.10	13.70	14.74	16.28	21.69	22.91	21.98	23.17	27.91	28.85

performance for mechanism shift datasets based on Mechanical MNIST and Mechanical MNIST-EMNIST Letters respectively. Tables 7 and 8 show the RMSE performance for sampling bias datasets based on Mechanical MNIST and Mechanical MNIST-EMNIST Letters respectively.

C.3. ML model prediction vs. ground truth visualization

Here we provide an additional supplement to Figs. 6–8 from Section 4. in Figs. 16–17, we visualize example comparisons between the the ground truth and the aggregated mean prediction of the change in strain energy through implementing OOD generalization algorithms (IRM, REx, IGA) and ERM on LeNet model and predicting on test environment 1 from each OOD dataset drawn from Mechanical MNIST and Mechanical MNIST - EMNIST Letters. In addition, the RMSE between the aggregated mean prediction and the ground truth for each method is shown in the legend of each plot. These values of RMSE can also be found in Tables 3–8. As shown in Figs. 16ab and 17ab, the prediction given by ERM underestimates the ground truth for the test data from the covariate and mechanism

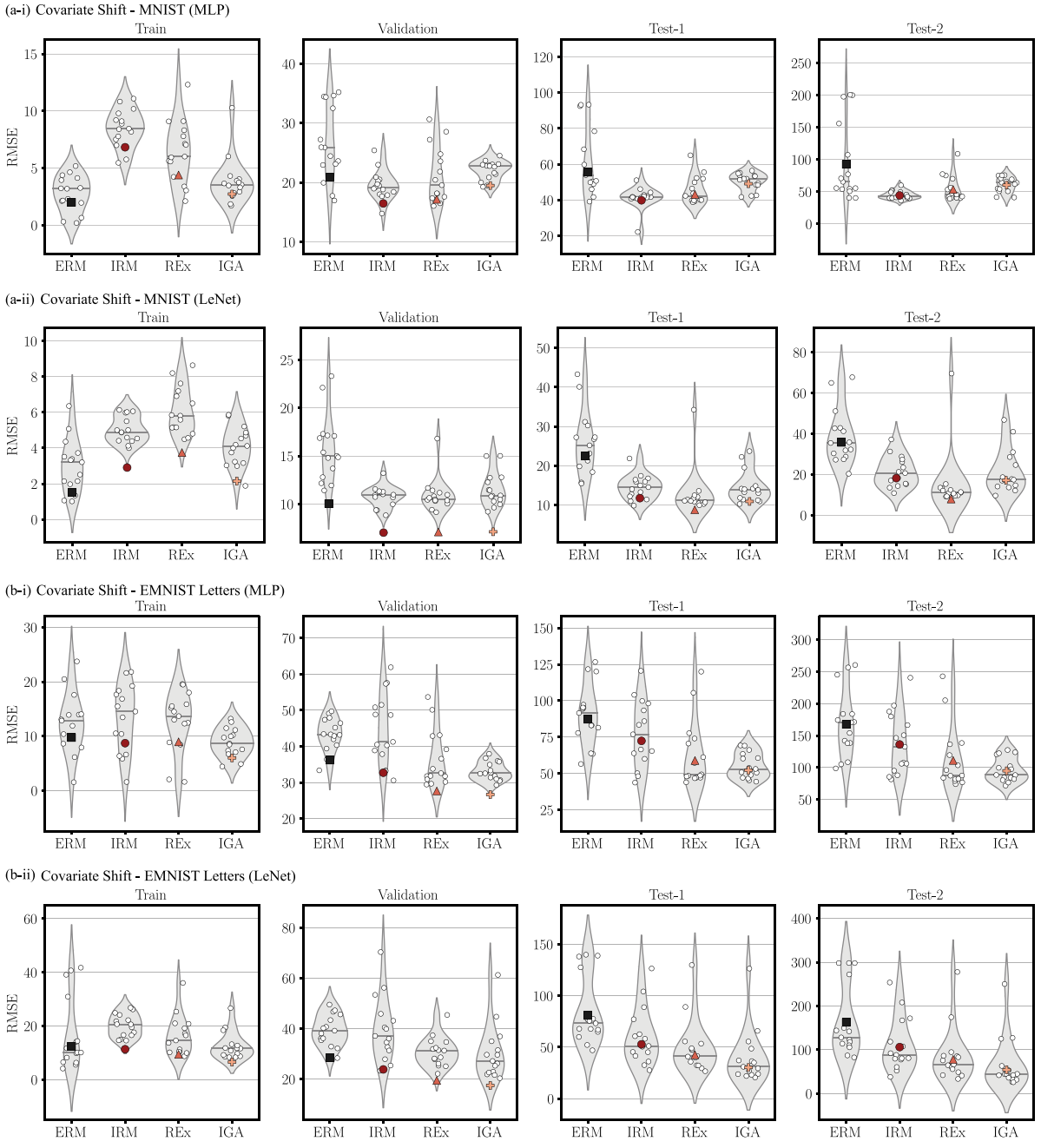

Fig. 13. The performance of the four algorithms (ERM, IRM, REx, IGA) on the covariate shift data defined in Section 2.2. Every white point represents the RMSE given by a single model initialized with different seed. The color-filled points show the RMSE of the aggregated mean prediction calculated by Eq. (13). (a) The performance of a MLP model (a-i) and the modified LeNet model (a-ii) trained by the four algorithms on training, validation, and testing data from the Mechanical MNIST Collection. (b) The performance of a MLP model (b-i) and a modified LeNet model (b-ii) trained by the four algorithms on training, validation, and testing data from the Mechanical MNIST - EMNIST Letters Collection.

shift datasets. While the prediction given by OOD generalization methods (IRM, REx and IGA) sometimes also underestimates the ground truth, it is closer to the ground truth (i.e., the change in strain energy calculated by FEM) and sometimes has no underestimation (e.g., prediction given by REx for covariate shift dataset from Mechanical

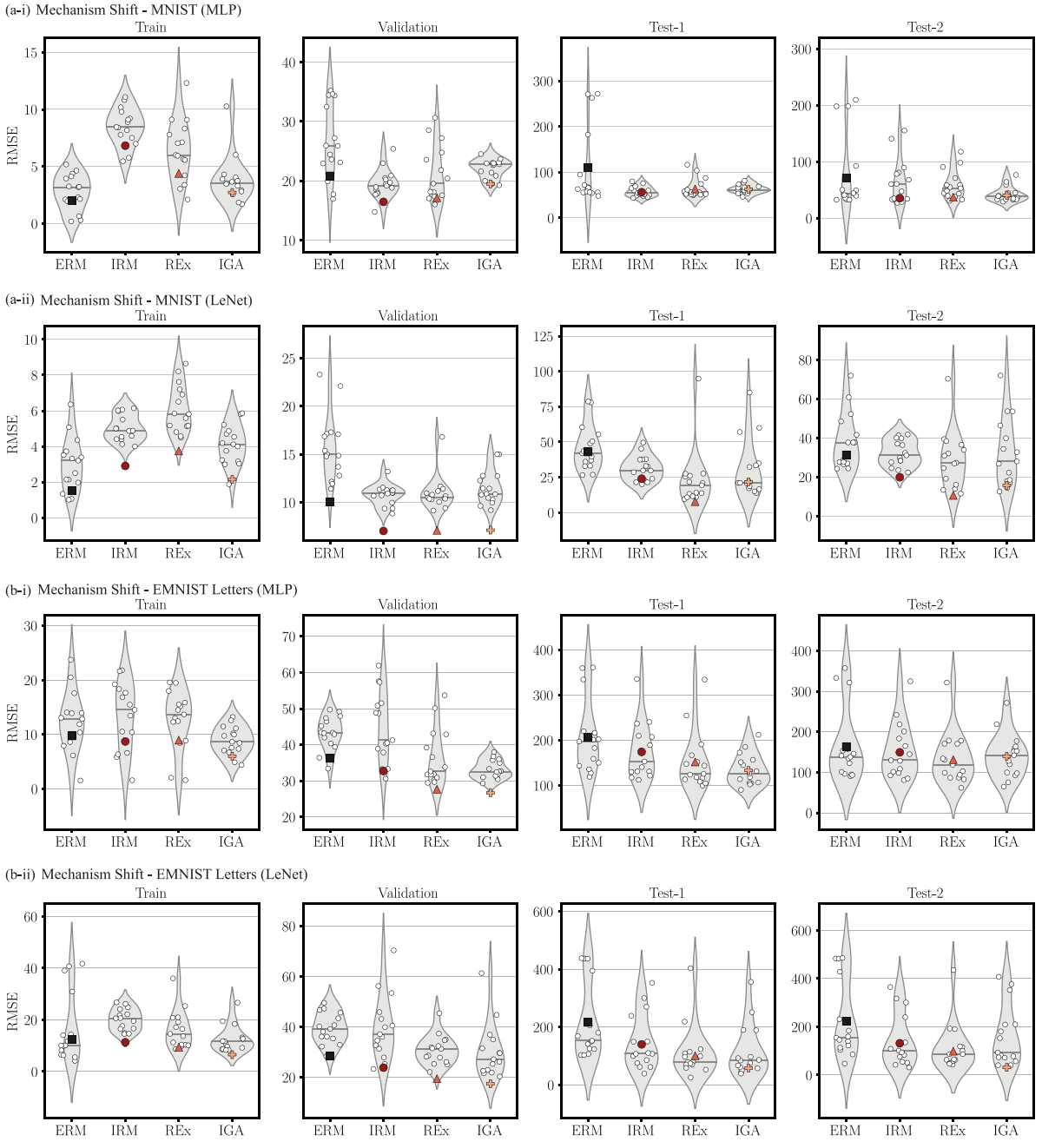

Fig. 14. The performance of the four algorithms (ERM, IRM, REx, IGA) on the mechanism shift data defined in Section 2.3. Every white point represents the RMSE given by a single model initialized with different seed. The color-filled points show the RMSE of the aggregated mean prediction calculated by Eq. (13). (a) The performance of a MLP model (a-i) and the modified LeNet model (a-ii) trained by the four algorithms on training, validation, and testing data from the Mechanical MNIST Collection. (b) The performance of a MLP model (b-i) and a modified LeNet model (b-ii) trained by the four algorithms on training, validation, and testing data from the Mechanical MNIST - EMNIST Letters Collection.

MNIST dataset). For the results from the sampling bias datasets shown in Fig. 16c and Fig. 17c, the prediction accuracy on the test environment is similar for ERM and OOD generalization methods, which is consistent with the discussion in Section 4.3.

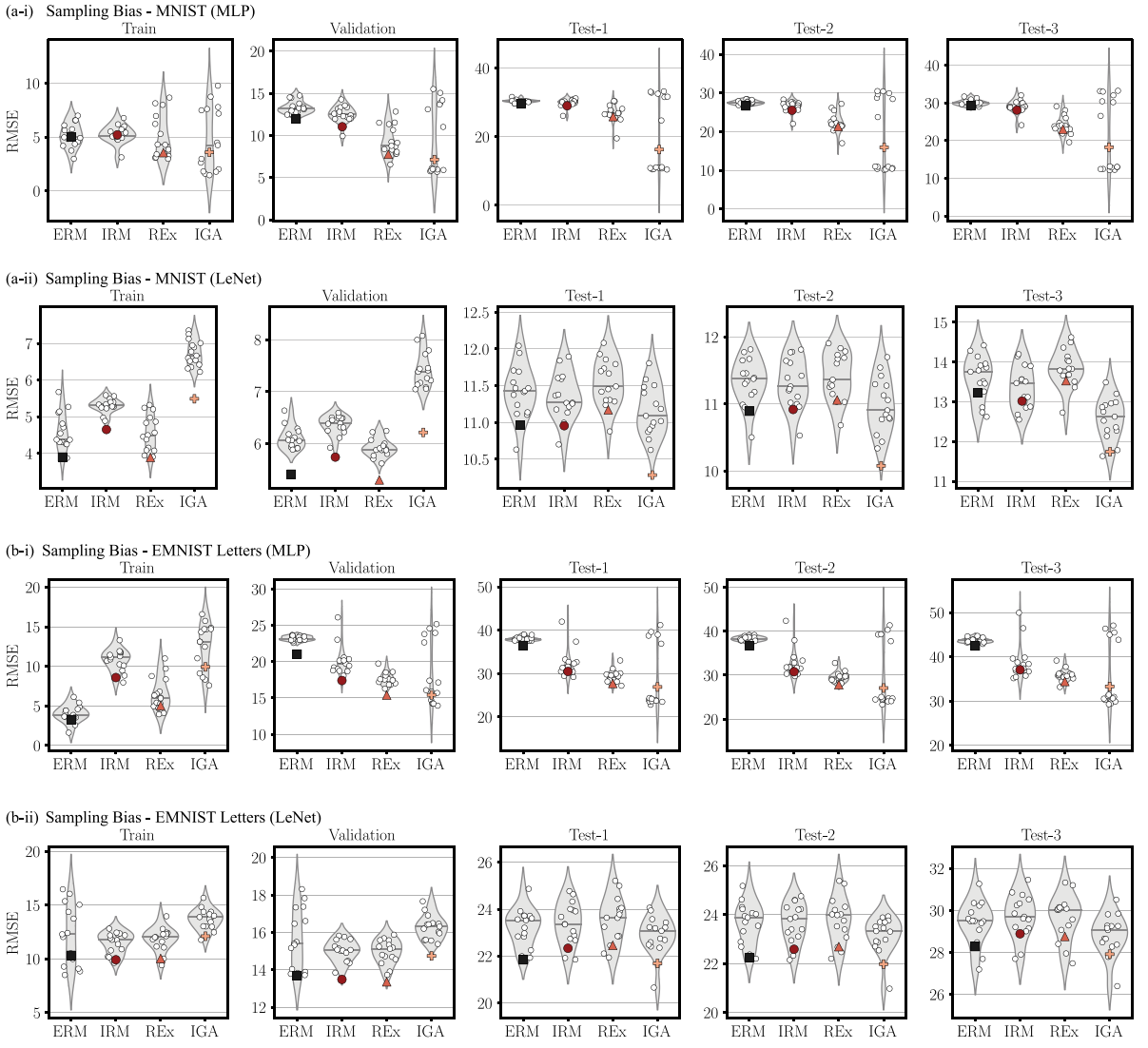

Fig. 15. The performance of the four algorithms (ERM, IRM, REx, IGA) on the sampling bias data defined in Section 2.4. Every white point represents the RMSE given by a single model initialized with different seed. The color-filled points show the RMSE of the aggregated mean prediction calculated by Eq. (13). (a) The performance of a MLP model (a-i) and the modified LeNet model (a-ii) trained by the four algorithms on training, validation, and testing data from the Mechanical MNIST Collection. (b) The performance of a MLP model (b-i) and a modified LeNet model (b-ii) trained by the four algorithms on training, validation, and testing data from the Mechanical MNIST - EMNIST Letters Collection.

C.4. Best and worst cases pattern visualization

To further support the results shown in Figs. 6–8 from Section 4 and the prediction vs. ground truth results shown in Figs. 16–17, here we present visualizations of representative samples from the Mechanical MNIST dataset that lead to different error levels. As shown in Fig. 18, which contains visualizations of the data shown in Fig. 16, we present randomly selected samples from test environment 1 for each OOD dataset from Mechanical MNIST. Specifically, for each algorithms (ERM, IRM, REx and IGA) on each test environment 1 from covariate shift (Fig. 18a), mechanism shift (Fig. 18b) and sampling bias dataset (Fig. 18c), we show examples at different levels of RMSE. To achieve this, sorted the data based on RMSE and sampled three cases from three groups: lowest error (bottom 10%), median error (45% − 55%) and highest error (top 10%). For each example, we report the ground

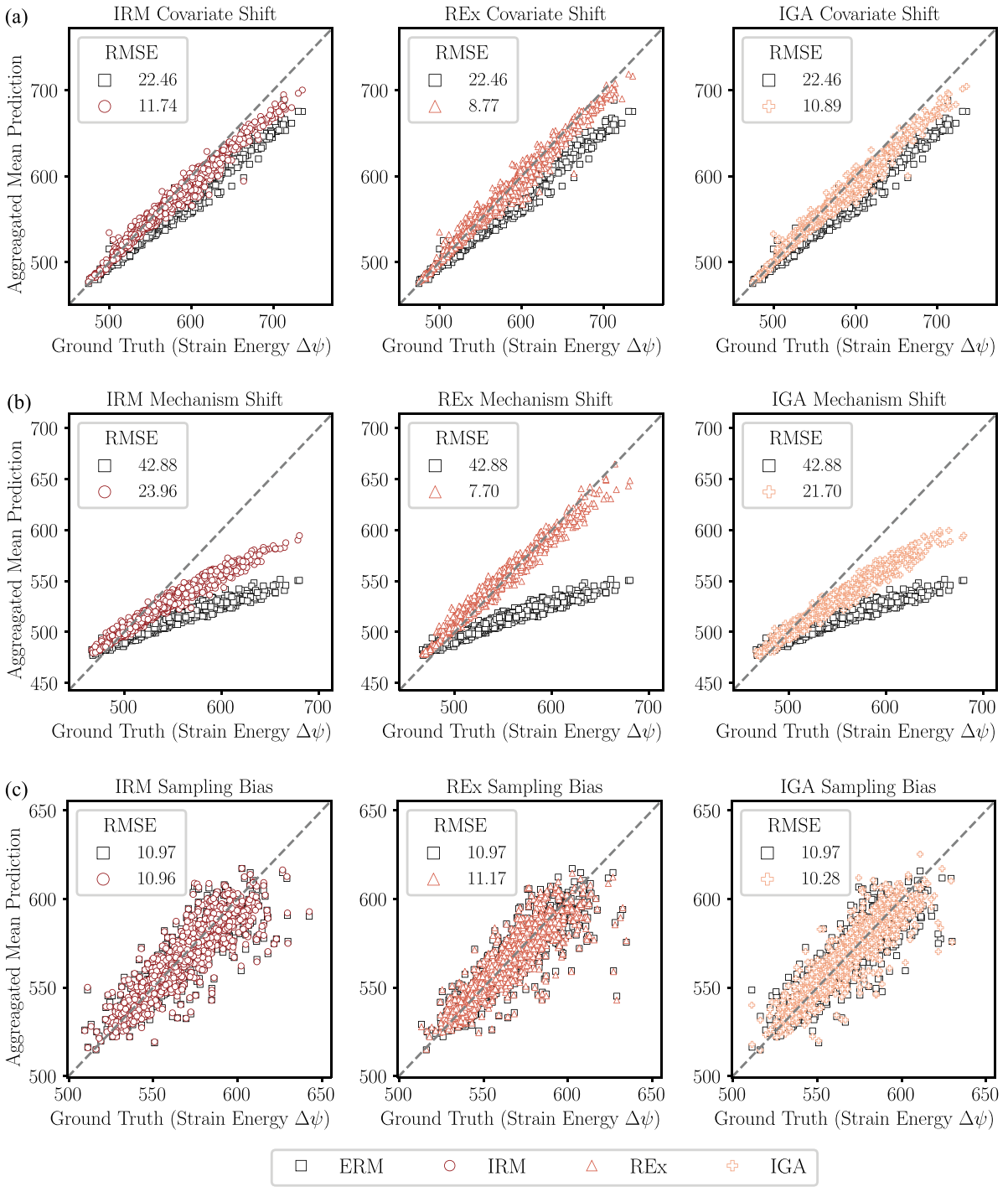

Fig. 16. The aggregated mean prediction versus the ground truth of the change in strain energy for OOD generalization algorithms (IRM, REx, IGA) and ERM on the test environment 1 of each OOD dataset on Mechanical MNIST.

truth of the change in strain energy, the predicted change in strain energy, and the corresponding RMSE calculated through Eq. (13). We note that for covariate shift and mechanism shift dataset, both the best (lowest error) and

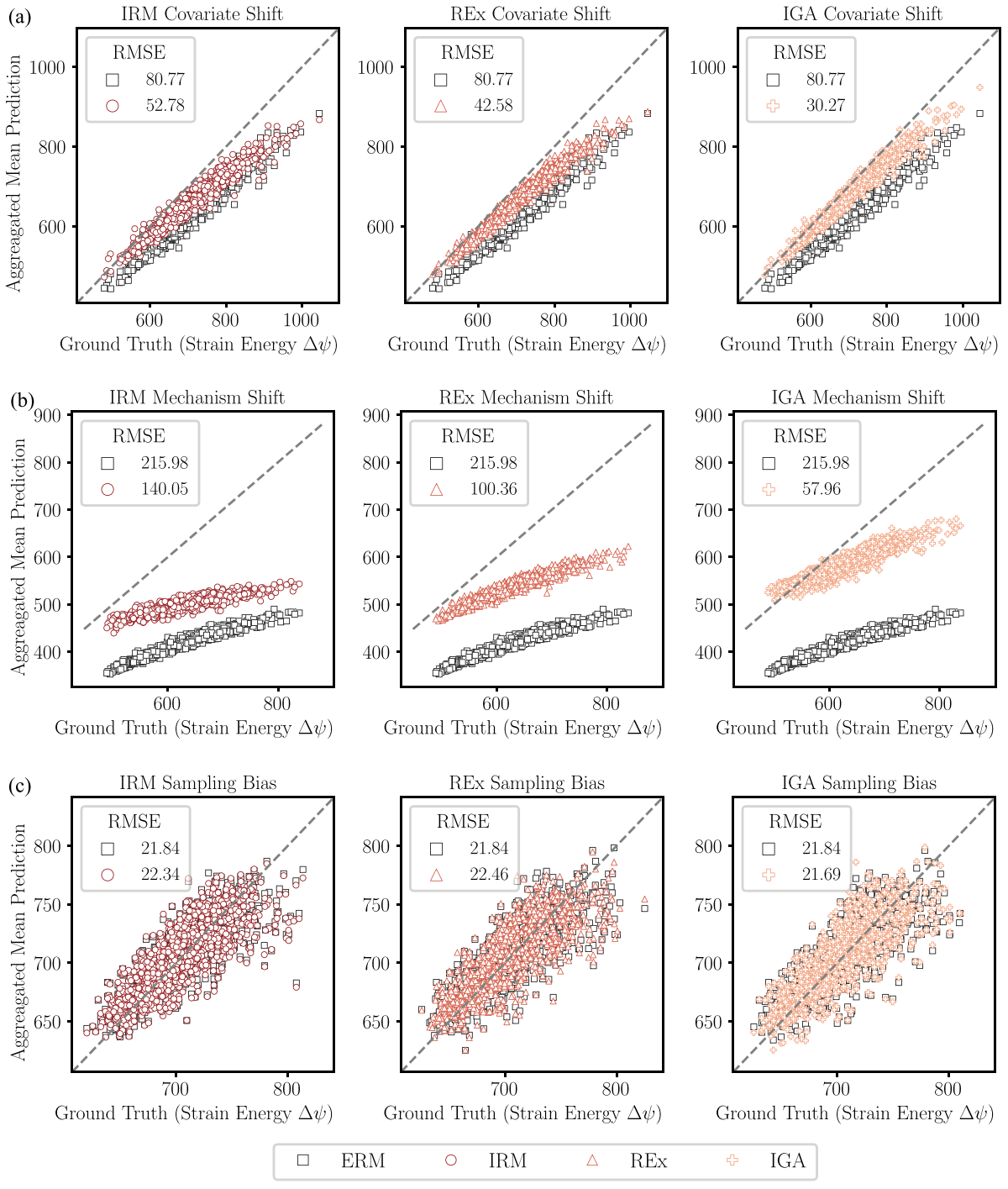

Fig. 17. The aggregated mean prediction versus the ground truth of the change in strain energy for OOD generalization algorithms (IRM, REx, IGA) and ERM on the test environment 1 of each OOD dataset on Mechanical MNIST - EMNIST Letters.

worst (highest error) cases of ERM lead to higher RMSE than other three OOD algorithms. We present the sample cases for Mechanical MNIST - EMNIST Letters on Fig. 19 with an identical format.

Fig. 18. The sampled cases of three groups: lowest error (bottom 10% RMSE), median error (45% − 55% RMSE) and highest error (top 10% RMSE) from the Mechanical MNIST collection and the corresponding ground truth and prediction of the change in strain energy. For each OOD dataset (a) Covariate Shift (b) Mechanism Shift and (c) Sampling Bias, the error (RMSE) is calculated and sorted based on the predicted value given by LeNet model on ERM, IRM, REx, and IGA respectively.

Fig. 19. The sampled cases of three groups: lowest error (bottom 10% RMSE), median error (45% − 55% RMSE) and highest error (top 10% RMSE) from the Mechanical MNIST - EMNIST Letters collection and the corresponding ground truth and prediction of the change in strain energy. For each OOD dataset (a) Covariate Shift (b) Mechanism Shift and (c) Sampling Bias, the error (RMSE) is calculated and sorted based on the predicted value given by LeNet model on ERM, IRM, REx, and IGA respectively.

References

[1] J.-H. Lee, H.S. Park, D.P. Holmes, Elastic instabilities govern the morphogenesis of the optic cup, Phys. Rev. Lett. 127 (13) (2021) 138102.

[2] L. Svolos, H.M. Mourad, G. Manzini, K. Garikipati, A fourth-order phase-field fracture model: Formulation and numerical solution using a continuous/discontinuous Galerkin method, J. Mech. Phys. Solids (2022) 104910.

[3] W.K. Liu, S. Li, H.S. Park, Eighty years of the finite element method: birth, evolution, and future, Arch. Comput. Methods Eng. (2022) 1–23.

[4] E. Süli, Lecture Notes on Finite Element Methods for Partial Differential Equations, Mathematical Institute, University of Oxford, 2012.

[5] H.S. Park, K. Gall, J.A. Zimmerman, Deformation of FCC nanowires by twinning and slip, J. Mech. Phys. Solids 54 (9) (2006) 1862–1881.

[6] J. Bian, L. Nicola, On the lubrication of rough copper surfaces with graphene, Tribol. Int. 156 (2021) 106837.

[7] K. Guo, Z. Yang, C.-H. Yu, M.J. Buehler, Artificial intelligence and machine learning in design of mechanical materials, Mater. Horiz. 8 (4) (2021) 1153–1172.

[8] M. Alber, A. Buganza Tepole, W.R. Cannon, S. De, S. Dura-Bernal, K. Garikipati, G. Karniadakis, W.W. Lytton, P. Perdikaris, L. Petzold, et al., Integrating machine learning and multiscale modeling—perspectives, challenges, and opportunities in the biological, biomedical, and behavioral sciences, NPJ Digit. Med. 2 (1) (2019) 1–11.

[9] G.C. Peng, M. Alber, A. Buganza Tepole, W.R. Cannon, S. De, S. Dura-Bernal, K. Garikipati, G. Karniadakis, W.W. Lytton, P. Perdikaris, et al., Multiscale modeling meets machine learning: What can we learn? Arch. Comput. Methods Eng. 28 (3) (2021) 1017–1037.

[10] J. Huang, J. Liew, A. Ademiloye, K.M. Liew, Artificial intelligence in materials modeling and design, Arch. Comput. Methods Eng. 28 (5) (2021) 3399–3413.

[11] F.E. Bock, R.C. Aydin, C.J. Cyron, N. Huber, S.R. Kalidindi, B. Klusemann, A review of the application of machine learning and data mining approaches in continuum materials mechanics, Front. Mater. (2019) 110.

[12] D. Morgan, R. Jacobs, Opportunities and challenges for machine learning in materials science, Annu. Rev. Mater. Res. 50 (2020) 71–103.

[13] G. Pilania, Machine learning in materials science: From explainable predictions to autonomous design, Comput. Mater. Sci. 193 (2021) 110360.

[14] G. Hadash, E. Kermany, B. Carmeli, O. Lavi, G. Kour, A. Jacovi, Estimate and replace: A novel approach to integrating deep neural networks with existing applications, 2018, arXiv preprint arXiv:1804.09028.

[15] S. Mohammadzadeh, E. Lejeune, Predicting mechanically driven full-field quantities of interest with deep learning-based metamodels, Extreme Mech. Lett. 50 (2022) 101566.

[16] Y. Kim, G.H. Gu, P. Asghari-Rad, J. Noh, J. Rho, M.H. Seo, H.S. Kim, Novel deep learning approach for practical applications of indentation, Mater. Today Adv. 13 (2022) 100207.

[17] X. Zhang, K. Garikipati, Machine learning materials physics: Multi-resolution neural networks learn the free energy and nonlinear elastic response of evolving microstructures, Comput. Methods Appl. Mech. Engrg. 372 (2020) 113362.

[18] J.R. Mianroodi, N. H. Siboni, D. Raabe, Teaching solid mechanics to artificial intelligence—a fast solver for heterogeneous materials, Npj Comput. Mater. 7 (1) (2021) 1–10.

[19] S. Saha, Z. Gan, L. Cheng, J. Gao, O.L. Kafka, X. Xie, H. Li, M. Tajdari, H.A. Kim, W.K. Liu, Hierarchical deep learning neural network (HiDeNN): An artificial intelligence (AI) framework for computational science and engineering, Comput. Methods Appl. Mech. Engrg. 373 (2021) 113452.

[20] P. Prachaseree, E. Lejeune, Learning mechanically driven emergent behavior with message passing neural networks, 2022, arXiv preprint arXiv:2202.01380.

[21] M. Mozaffar, R. Bostanabad, W. Chen, K. Ehmann, J. Cao, M. Bessa, Deep learning predicts path-dependent plasticity, Proc. Natl. Acad. Sci. 116 (52) (2019) 26414–26420.

[22] J.N. Fuhg, L. van Wees, M. Obstalecki, P. Shade, N. Bouklas, M. Kasemer, Machine-learning convex and texture-dependent macroscopic yield from crystal plasticity simulations, Materialia (2022) 101446.

[23] C. Wang, S. Li, D. Zeng, X. Zhu, Quantification and compensation of thermal distortion in additive manufacturing: a computational statistics approach, Comput. Methods Appl. Mech. Engrg. 375 (2021) 113611.

[24] G. Chen, T. Li, Q. Chen, S. Ren, C. Wang, S. Li, Application of deep learning neural network to identify collision load conditions based on permanent plastic deformation of shell structures, Comput. Mech. 64 (2) (2019) 435–449.

[25] M. Su, Q. Zhong, H. Peng, S. Li, Selected machine learning approaches for predicting the interfacial bond strength between FRPs and concrete, Constr. Build. Mater. 270 (2021) 121456.

[26] Q. Chen, Y. Xie, Y. Ao, T. Li, G. Chen, S. Ren, C. Wang, S. Li, A deep neural network inverse solution to recover pre-crash impact data of car collisions, Transp. Res. C 126 (2021) 103009.

[27] C.-T. Chen, G.X. Gu, Generative deep neural networks for inverse materials design using backpropagation and active learning, Adv. Sci. 7 (5) (2020) 1902607.

[28] P.Z. Hanakata, E.D. Cubuk, D.K. Campbell, H.S. Park, Forward and inverse design of kirigami via supervised autoencoder, Phys. Rev. Res. 2 (4) (2020) 042006.

[29] B. Sanchez-Lengeling, A. Aspuru-Guzik, Inverse molecular design using machine learning: Generative models for matter engineering, Science 361 (6400) (2018) 360–365.

[30] H.T. Kollmann, D.W. Abueidda, S. Koric, E. Guleryuz, N.A. Sobh, Deep learning for topology optimization of 2D metamaterials, Mater. Des. 196 (2020) 109098.
[31] A.E. Forte, P.Z. Hanakata, L. Jin, E. Zari, A. Zareei, M.C. Fernandes, L. Sumner, J. Alvarez, K. Bertoldi, Inverse design of inflatable soft membranes through machine learning, Adv. Funct. Mater. (2022) 2111610.
[32] F. Liu, X. Jiang, X. Wang, L. Wang, Machine learning-based design and optimization of curved beams for multistable structures and metamaterials, Extreme Mech. Lett. 41 (2020) 101002.
[33] A. Challapalli, D. Patel, G. Li, Inverse machine learning framework for optimizing lightweight metamaterials, Mater. Des. 208 (2021) 109937.
[34] A.E. Gongora, B. Xu, W. Perry, C. Okoye, P. Riley, K.G. Reyes, E.F. Morgan, K.A. Brown, A Bayesian experimental autonomous researcher for mechanical design, Sci. Adv. 6 (15) (2020) eaaz1708.
[35] B. Ni, H. Gao, A deep learning approach to the inverse problem of modulus identification in elasticity, MRS Bull. 46 (1) (2021) 19–25.
[36] H. Kobeissi, S. Mohammadzadeh, E. Lejeune, Enhancing mechanical metamodels with a generative model-based augmented training dataset, J. Biomech. Eng. 144 (12) (2022) 121002.
[37] Q. Liang, A.E. Gongora, Z. Ren, A. Tiihonen, Z. Liu, S. Sun, J.R. Deneault, D. Bash, F. Mekki-Berrada, S.A. Khan, et al., Benchmarking the performance of Bayesian optimization across multiple experimental materials science domains, Npj Comput. Mater. 7 (1) (2021) 1–10.
[38] X. Liu, C.E. Athanasiou, N.P. Padture, B.W. Sheldon, H. Gao, Knowledge extraction and transfer in data-driven fracture mechanics, Proc. Natl. Acad. Sci. 118 (23) (2021).
[39] V. François-Lavet, P. Henderson, R. Islam, M.G. Bellemare, J. Pineau, et al., An introduction to deep reinforcement learning, Found. Trends Mach. Learn. 11 (3–4) (2018) 219–354.
[40] F. Sui, R. Guo, Z. Zhang, G.X. Gu, L. Lin, Deep reinforcement learning for digital materials design, ACS Mater. Lett. 3 (10) (2021) 1433–1439.
[41] K. Wang, W. Sun, Meta-modeling game for deriving theory-consistent, microstructure-based traction–separation laws via deep reinforcement learning, Comput. Methods Appl. Mech. Engrg. 346 (2019) 216–241.
[42] K. Wang, W. Sun, Q. Du, A cooperative game for automated learning of elasto-plasticity knowledge graphs and models with AI-guided experimentation, Comput. Mech. 64 (2) (2019) 467–499.
[43] E. Lejeune, B. Zhao, Exploring the potential of transfer learning for metamodels of heterogeneous material deformation, J. Mech. Behav. Biomed. Mater. 117 (2021) 104276.
[44] L. Lu, M. Dao, P. Kumar, U. Ramamurty, G.E. Karniadakis, S. Suresh, Extraction of mechanical properties of materials through deep learning from instrumented indentation, Proc. Natl. Acad. Sci. 117 (13) (2020) 7052–7062.
[45] S. Goswami, K. Kontolati, M.D. Shields, G.E. Karniadakis, Deep transfer learning for partial differential equations under conditional shift with DeepONet, 2022, arXiv preprint arXiv:2204.09810.
[46] Z. Shen, J. Liu, Y. He, X. Zhang, R. Xu, H. Yu, P. Cui, Towards out-of-distribution generalization: A survey, 2021, arXiv preprint arXiv:2108.13624.
[47] S. Sagawa, P.W. Koh, T.B. Hashimoto, P. Liang, Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization, 2019, arXiv preprint arXiv:1911.08731.
[48] R. Guo, L. Cheng, J. Li, P.R. Hahn, H. Liu, A survey of learning causality with data: Problems and methods, ACM Comput. Surv. 53 (4) (2020) 1–37.
[49] M. Arjovsky, L. Bottou, I. Gulrajani, D. Lopez-Paz, Invariant risk minimization, 2019, arXiv preprint arXiv:1907.02893.
[50] S. Sagawa, A. Raghunathan, P.W. Koh, P. Liang, An investigation of why overparameterization exacerbates spurious correlations, in: International Conference on Machine Learning, PMLR, 2020, pp. 8346–8356.
[51] V. Nagarajan, A. Andreassen, B. Neyshabur, Understanding the failure modes of out-of-distribution generalization, 2020, arXiv preprint arXiv:2010.15775.
[52] S. Beery, G. Van Horn, P. Perona, Recognition in terra incognita, in: Proceedings of the European Conference on Computer Vision, ECCV, 2018, pp. 456–473.
[53] A. Kurakin, I.J. Goodfellow, S. Bengio, Adversarial examples in the physical world, in: Artificial Intelligence Safety and Security, Chapman and Hall/CRC, 2018, pp. 99–112.
[54] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F.A. Wichmann, W. Brendel, ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness, 2018, arXiv preprint arXiv:1811.12231.
[55] D. Tsipras, S. Santurkar, L. Engstrom, A. Turner, A. Madry, Robustness may be at odds with accuracy, 2018, arXiv preprint arXiv:1805.12152.
[56] H. Ye, C. Xie, T. Cai, R. Li, Z. Li, L. Wang, Towards a theoretical framework of out-of-distribution generalization, Adv. Neural Inf. Process. Syst. 34 (2021).
[57] R. Hu, J. Sang, J. Wang, C. Jiang, Understanding and testing generalization of deep networks on out-of-distribution data, 2021, arXiv preprint arXiv:2111.09190.
[58] P. Izmailov, P. Nicholson, S. Lotfi, A.G. Wilson, Dangers of Bayesian model averaging under covariate shift, Adv. Neural Inf. Process. Syst. 34 (2021).
[59] I.J. Goodfellow, J. Shlens, C. Szegedy, Explaining and harnessing adversarial examples, 2014, arXiv preprint arXiv:1412.6572.
[60] Y. Kim, Y. Kim, C. Yang, K. Park, G.X. Gu, S. Ryu, Deep learning framework for material design space exploration using active transfer learning and data augmentation, Npj Comput. Mater. 7 (1) (2021) 1–7.
[61] T. DeVries, G.W. Taylor, Learning confidence for out-of-distribution detection in neural networks, 2018, arXiv preprint arXiv:1802.04865.

[62] Y. Ming, H. Yin, Y. Li, On the impact of spurious correlation for out-of-distribution detection, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 36, 2022, pp. 10051–10059.
[63] J. Yang, K. Zhou, Y. Li, Z. Liu, Generalized out-of-distribution detection: A survey, 2021, arXiv preprint arXiv:2110.11334.
[64] C. Berger, M. Paschali, B. Glocker, K. Kamnitsas, Confidence-based out-of-distribution detection: a comparative study and analysis, in: Uncertainty for Safe Utilization of Machine Learning in Medical Imaging, and Perinatal Imaging, Placental and Preterm Image Analysis, Springer, 2021, pp. 122–132.
[65] X. Wang, L. Aitchison, Bayesian OOD detection with aleatoric uncertainty and outlier exposure, in: Fourth Symposium on Advances in Approximate Bayesian Inference, 2021.
[66] C. Henning, F. D'Angelo, B.F. Grewe, Are Bayesian neural networks intrinsically good at out-of-distribution detection? 2021, arXiv preprint arXiv:2107.12248.
[67] Y. Xie, C. Wu, B. Li, X. Hu, S. Li, A feed-forwarded neural network-based variational Bayesian learning approach for forensic analysis of traffic accident, Comput. Methods Appl. Mech. Engrg. 397 (2022) 115148.
[68] Y. Xie, S. Li, C. Wu, D. Lyu, C. Wang, D. Zeng, A generalized Bayesian regularization network approach on characterization of geometric defects in lattice structures for topology optimization in preliminary design of 3D printing, Comput. Mech. 69 (5) (2022) 1191–1212.
[69] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.
[70] M. Raissi, A. Yazdani, G.E. Karniadakis, Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations, Science 367 (6481) (2020) 1026–1030.
[71] S. Cuomo, V.S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, F. Piccialli, Scientific machine learning through physics-informed neural networks: Where we are and what's next, 2022, arXiv preprint arXiv:2201.05624.
[72] S. Cai, Z. Wang, S. Wang, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks for heat transfer problems, J. Heat Transfer 143 (6) (2021).
[73] F. Fuchs, D. Worrall, V. Fischer, M. Welling, Se (3)-transformers: 3d roto-translation equivariant attention networks, Adv. Neural Inf. Process. Syst. 33 (2020) 1970–1981.
[74] V.G. Satorras, E. Hoogeboom, M. Welling, E (n) equivariant graph neural networks, in: International Conference on Machine Learning, PMLR, 2021, pp. 9323–9332.
[75] T.E. Smidt, M. Geiger, B.K. Miller, Finding symmetry breaking order parameters with Euclidean neural networks, Phys. Rev. Res. 3 (1) (2021) L012002.
[76] T.S. Cohen, M. Geiger, M. Weiler, A general theory of equivariant cnns on homogeneous spaces, Adv. Neural Inf. Process. Syst. 32 (2019).
[77] S.L. Brunton, J.L. Proctor, J.N. Kutz, Discovering governing equations from data by sparse identification of nonlinear dynamical systems, Proc. Natl. Acad. Sci. 113 (15) (2016) 3932–3937.
[78] K. Champion, B. Lusch, J.N. Kutz, S.L. Brunton, Data-driven discovery of coordinates and governing equations, Proc. Natl. Acad. Sci. 116 (45) (2019) 22445–22451.
[79] M. Quade, M. Abel, J. Nathan Kutz, S.L. Brunton, Sparse identification of nonlinear dynamics for rapid model recovery, Chaos 28 (6) (2018) 063116.
[80] T. Lookman, P.V. Balachandran, D. Xue, R. Yuan, Active learning in materials science with emphasis on adaptive sampling using uncertainties for targeted design, Npj Comput. Mater. 5 (1) (2019) 1–17.
[81] M. Wang, W. Deng, Deep visual domain adaptation: A survey, Neurocomputing 312 (2018) 135–153.
[82] G. Csurka, A comprehensive survey on domain adaptation for visual applications, in: Domain Adaptation in Computer Vision Applications, Springer, 2017, pp. 1–35.
[83] A. Farahani, S. Voghoei, K. Rasheed, H.R. Arabnia, A brief review of domain adaptation, in: Advances in Data Science and Information Engineering, Springer, 2021, pp. 877–894.
[84] K. Zhou, Z. Liu, Y. Qiao, T. Xiang, C.C. Loy, Domain generalization in vision: A survey, 2021, arXiv preprint arXiv:2103.02503.
[85] J. Peters, P. Bühlmann, N. Meinshausen, Causal inference by using invariant prediction: identification and confidence intervals, J. R. Stat. Soc. Ser. B Stat. Methodol. 78 (5) (2016) 947–1012.
[86] B. Schölkopf, Causality for machine learning, in: Probabilistic and Causal Inference: The Works of Judea Pearl, 2022, pp. 765–804.
[87] P. Bühlmann, Invariance, causality and robustness, Statist. Sci. 35 (3) (2020) 404–426.
[88] S. Weichwald, J. Peters, Causality in cognitive neuroscience: concepts, challenges, and distributional robustness, J. Cogn. Neurosci. 33 (2) (2021) 226–247.
[89] M. Koyama, S. Yamaguchi, Out-of-distribution generalization with maximal invariant predictor, 2020.
[90] D. Krueger, E. Caballero, J.-H. Jacobsen, A. Zhang, J. Binas, D. Zhang, R. Le Priol, A. Courville, Out-of-distribution generalization via risk extrapolation (rex), in: International Conference on Machine Learning, PMLR, 2021, pp. 5815–5826.
[91] D. Mahajan, S. Tople, A. Sharma, Domain generalization using causal matching, in: International Conference on Machine Learning, PMLR, 2021, pp. 7313–7324.
[92] J.C. Duchi, H. Namkoong, Learning models with uniform performance via distributionally robust optimization, Ann. Statist. 49 (3) (2021) 1378–1406.
[93] K. Ahuja, K. Shanmugam, K. Varshney, A. Dhurandhar, Invariant risk minimization games, in: International Conference on Machine Learning, PMLR, 2020, pp. 145–155.
[94] S. Chang, Y. Zhang, M. Yu, T. Jaakkola, Invariant rationalization, in: International Conference on Machine Learning, PMLR, 2020, pp. 1448–1458.

[95] V. Piratla, P. Netrapalli, S. Sarawagi, Efficient domain generalization via common-specific low-rank decomposition, in: International Conference on Machine Learning, PMLR, 2020, pp. 7728–7738.
[96] I. Gulrajani, D. Lopez-Paz, In search of lost domain generalization, 2020, arXiv preprint arXiv:2007.01434.
[97] P.W. Koh, S. Sagawa, H. Marklund, S.M. Xie, M. Zhang, A. Balsubramani, W. Hu, M. Yasunaga, R.L. Phillips, I. Gao, et al., Wilds: A benchmark of in-the-wild distribution shifts, in: International Conference on Machine Learning, PMLR, 2021, pp. 5637–5664.
[98] J. Liu, Z. Hu, P. Cui, B. Li, Z. Shen, Heterogeneous risk minimization, in: International Conference on Machine Learning, PMLR, 2021, pp. 6804–6814.
[99] K. Kuang, R. Xiong, P. Cui, S. Athey, B. Li, Stable prediction with model misspecification and agnostic distribution shift, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 4485–4492.
[100] N. Tripuraneni, B. Adlam, J. Pennington, Overparameterization improves robustness to covariate shift in high dimensions, Adv. Neural Inf. Process. Syst. 34 (2021).
[101] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[102] E. Kussul, T. Baidyk, Improved method of handwritten digit recognition tested on MNIST database, Image Vis. Comput. 22 (12) (2004) 971–981.
[103] D.C. Cireşan, U. Meier, L.M. Gambardella, J. Schmidhuber, Deep, big, simple neural nets for handwritten digit recognition, Neural Comput. 22 (12) (2010) 3207–3220.
[104] S. An, M. Lee, S. Park, H. Yang, J. So, An ensemble of simple convolutional neural network models for MNIST digit recognition, 2020, arXiv preprint arXiv:2008.10400.
[105] E. Lejeune, Mechanical MNIST: A benchmark dataset for mechanical metamodels, Extreme Mech. Lett. 36 (2020) 100659.
[106] A. Logg, G.N. Wells, DOLFIN: Automated finite element computing, ACM Trans. Math. Softw. 37 (2) (2010) 1–28.
[107] A. Logg, K.-A. Mardal, G. Wells, Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book, Vol. 84, Springer Science & Business Media, 2012.
[108] D. Li, Y. Yang, Y.-Z. Song, T.M. Hospedales, Deeper, broader and artier domain generalization, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5542–5550.
[109] A. Tsymbal, The problem of concept drift: definitions and related work, Comput. Sci. Dep. Trinity College Dublin 106 (2) (2004) 58.
[110] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, G. Zhang, Learning under concept drift: A review, IEEE Trans. Knowl. Data Eng. 31 (12) (2018) 2346–2363.
[111] C. Winship, R.D. Mare, Models for sample selection bias, Annu. Rev. Sociol. 18 (1) (1992) 327–350.
[112] C. Winship, S.L. Morgan, The estimation of causal effects from observational data, Annu. Rev. Sociol. 25 (1) (1999) 659–706.
[113] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 248–255.
[114] J. Liu, Z. Hu, P. Cui, B. Li, Z. Shen, Kernelized heterogeneous risk minimization, 2021, arXiv preprint arXiv:2110.12425.
[115] G. Cohen, S. Afshar, J. Tapson, A. Van Schaik, EMNIST: Extending MNIST to handwritten letters, in: 2017 International Joint Conference on Neural Networks, IJCNN, IEEE, 2017, pp. 2921–2926.
[116] A.J. Smits, B.J. McKeon, I. Marusic, High–Reynolds number wall turbulence, Annu. Rev. Fluid Mech. 43 (2011) 353–375.
[117] Y.-c. Fung, Biomechanics: Mechanical Properties of Living Tissues, Springer Science & Business Media, 2013.
[118] E. Creager, J.-H. Jacobsen, R. Zemel, Environment inference for invariant learning, in: International Conference on Machine Learning, PMLR, 2021, pp. 2189–2200.