Machine Learning with Applications 16 (2024) 100544

Contents lists available at ScienceDirect

Machine Learning with Applications

journal homepage: www.elsevier.com/locate/mlwa

Efficient surrogate models for materials science simulations: Machine learning-based prediction of microstructure properties

Binh Duong Nguyen [a], Pavlo Potapenko [a], Aytekin Demirci [a], Kishan Govind [a], Sébastien Bompas [a], Stefan Sandfeld [a,b,*]

[a] Institute for Advanced Simulations – Materials Data Science and Informatics (IAS-9), Forschungszentrum Jülich GmbH, 52425 Jülich, Germany
[b] Chair of Materials Data Science and Materials Informatics, Faculty 5 – Georesources and Materials Engineering, RWTH Aachen University, 52056 Aachen, Germany

ARTICLE INFO

Dataset link: https://gitlab.com/computational-materials-science/public/publication-data-and-code/2024_Nguyen_et_al_MLWA_ML-based-predictions-of-microstructure-properties, https://doi.org/10.5281/zenodo.10821847

Keywords:
Structure-properties relation
Forward model
Feature engineering
Power spectrum density
Convolutional neural network
Support vector regression
Ising model
Cahn–Hilliard model

ABSTRACT

Determining, understanding, and predicting the so-called structure–property relation is an important task in many scientific disciplines, such as chemistry, biology, meteorology, physics, engineering, and materials science. *Structure* refers to the spatial distribution of, e.g., substances, material, or matter in general, while *property* is a resulting characteristic that usually depends in a non-trivial way on spatial details of the structure. Traditionally, forward simulations models have been used for such tasks. Recently, several machine learning algorithms have been applied in these scientific fields to enhance and accelerate simulation models or as surrogate models. In this work, we develop and investigate the applications of six machine learning techniques based on two different datasets from the domain of materials science: data from a two-dimensional Ising model for predicting the formation of magnetic domains and data representing the evolution of dual-phase microstructures from the Cahn–Hilliard model. We analyze the accuracy and robustness of all models and elucidate the reasons for the differences in their performances. The impact of including domain knowledge through tailored features is studied, and general recommendations based on the availability and quality of training data are derived from this.

1. Introduction

Studying the (micro)structure-properties relation is an important task for many different scientific fields and on many different length scales, e.g., for meteorology with up to kilometer-sized features, for materials science on the nanometer scale or for biological or chemical systems on various length scales (Kohn, Underwood, & Cooper, 2018). Mathematically, the task is to find the map from a (one-, two-, or three-dimensional) spatial distribution of values to a single (scalar, vectorial, or tensorial) value. For example, geological measurements of the three-dimensional structural details of the earth's crust are accompanied by displacement measurements which represents an average, i.e., an effective property, and can help to understand the general mechanism for shallow earthquakes (Tarasov, 2019). In the field of weather forecasting, spatial details such as the structure of clouds or streamlines of the airflow determine properties such as cloud top temperature and particle effective radius (Rosenfeld, Woodley, Lerner, Kelman, & Lindsey, 2008). Biological structures consist of molecules and cells, that are permanently evolving. Their subcellular interactions give rise to complex properties such as transport properties or how cells age (Li, Tang, & Guo, 2021). In the domain of material science and in particular, with regards to metallic materials, the notion of microstructure refers to any phenomena that "lives" on a small scale (i.e., small relative to the specimen size) and that disturbs the otherwise perfect crystal lattice (Callister, 2007). The property is typically the result of the interplay of many different physical or chemical mechanisms; a property is an averaged quantity where the details of the underlying microstructural length scale usually are no longer directly observable. Two typical examples are: (i) interstitial point defects on the atomic scale that gives rise to hardening behavior in alloys observed during mechanical testing of centimeter-sized samples (Baker, 2022), or (ii) the property of strength, which increases considerably with a decrease in grain size (Opiela, Fojt-Dymara, Grajcar, & Borek, 2020).

To predict such properties based on microstructures of different length scales, dedicated simulation models have been developed

* Corresponding author at: Institute for Advanced Simulations – Materials Data Science and Informatics (IAS-9), Forschungszentrum Jülich GmbH, 52425 Jülich, Germany.
E-mail addresses: bi.nguyen@fz-juelich.de (B.D. Nguyen), p.potapenko@fz-juelich.de (P. Potapenko), a.demirci@fz-juelich.de (A. Demirci), k.govind@fz-juelich.de (K. Govind), s.bompas@fz-juelich.de (S. Bompas), s.sandfeld@fz-juelich.de (S. Sandfeld).

https://doi.org/10.1016/j.mlwa.2024.100544
Received 27 August 2023; Received in revised form 14 November 2023; Accepted 9 March 2024
Available online 11 March 2024

(Nguyen, Choi, Udaykumar, & Baek, 2023; Seif, 2023; Sharma et al., 2023). On the (sub)nanometer scale, one of the commonly utilized methods is density functional theory calculations, which investigate the electronic structure of many-body systems to acquire the properties of the electron system based on the fundamental laws of quantum mechanics (Dreizler & Gross, 2012). Another popular method, molecular dynamics simulation, predicts the trajectory of each atom based on Newton's laws (Hospital, Goñi, Orozco, & Gelpí, 2015). These two methods can calculate very accurately the structures and properties of a material on a microscopic scale, but the computational cost for large-scale problems is prohibitive. If the problem can be written in terms of continuous field equations governed by partial differential equations, numerical methods such as the finite element method (Huebner, Dewhirst, Smith, & Byrom, 2001) are often used. However, the computational cost can still be high, with single simulations taking up to several days. Additionally, the numerical solution sometimes suffers from numerical instabilities, which makes performing simulations still a challenging task.

Recent development of statistical ML and deep learning (DL) algorithms have the potential to act as surrogate models and/or provide alternatives for predicting the structure–property relation in science and engineering. This helps to overcome the limitations of classical methods in terms of computational cost and robustness (Gupta & Zhang, 2024; Jung, Yoon, Park, Kim, & Kim, 2019; Wei et al., 2019). For example, DL approaches have been applied to weather forecasting to predict the likelihood of weather conditions at a given time and location based on numerous atmospheric and oceanic properties such as pressure, humidity, wind velocity and temperature from radar or weather satellite images (Espeholt et al., 2022). ML-based short-term forecasting of earthquakes using remote-sensing (image) data was demonstrated to outperform conventional approaches (Xiong et al., 2021). The data augmentation process with discrete waveform transforms (DWT) and singular value decomposition (SVD) helps to increase the variety of earthquake data for training ML models. These models are then used to predict the response of nonlinear systems for unseen earthquakes or to replicate non-linear FE model prediction (Parida, Bose and Apostolakis, 2023; Parida, Bose, Butcher, Apostolakis and Shekhar, 2023). Additionally, such methods have been applied to computational biology (Sapoval et al., 2022) for protein structure prediction from its amino acid sequences (Jumper et al., 2021; Tunyasuvunakool et al., 2021), for predicting the melting temperature of proteins based on their amino acid sequence (Gorania, Seker, & Haris, 2010), or for predicting the ligand binding sites in the protein structures (Kandel, Tayara, & Chong, 2021). The field of materials science also benefits from ML methods. In the context of surrogate models, Nakka, Harursampath, and Ponnusami (2023) created a DL-based model for encoding material properties into the microstructure image so that the model learns material information. Messner (2020) use Convolutional Neural Networks (CNN) as "sufficiently accurate surrogate models" for solving the inverse design problem that produces optimal structures with required mechanical properties. Further example applications in the context of structure–property relations are the ML-based prediction of the hardness and relative mass density of nanocomposites based on microstructural texture variance produced by different laser parameters (Yu, Mo, Chen, & Yao, 2021), the diffusivity and permeability based on the geometry of the pore space utilizing artificial neural networks (Prifling, Röding, Townsend, Neumann, & Schmidt, 2021), the effective heat conductivity for highly heterogeneous microstructured materials (Lißner & Fritzen, 2019), the surrogate modeling of the mechanical response of elasto-viscoplastic grain microstructures (Khorrami et al., 2023) or the prediction of mechanical properties of two-phase microstructures of epoxy-carbon fiber aerospace composite (Ford, Maneparambil, Rajan, & Neithalath, 2021). Most of these approaches are concerned with mapping an input to an output. A variety of approaches have incorporated more general physics knowledge into the model, mostly into the loss function, such as Zhang, Liu, and Sun (2020a, 2020b) who used a physics-guided convolutional neural network and physics-informed multi-LSTM networks as surrogate models for structural response prediction. Raissi (2018) introduced physics-informed neural networks for solving nonlinear partial differential equations, and Eghbalian, Pouragha, and Wan (2023) develops an Elasto-Plastic Neural Network for replacing the conventional yield function, plastic potential, and the plastic flow rule.

Many of these examples, however, consider highly specialized scientific situations where the focus is on solving a particular domain-scientific problem with an as high as possible accuracy. Systematic studies with an emphasis on aspects of the training behavior, the ability to generalize, or the performance w.r.t. to the amount of data are rare. This makes comparison between the work of different groups difficult and it is far from being trivial to estimate if a model could be reused for a different problem class as well.

In this work, we investigate the benefits and drawbacks of a range of machine learning approaches for predicting properties from structures of two fundamentally different, materials science datasets. Besides different ML model complexities, we also investigate the importance of incorporating domain knowledge through feature engineering. The datasets are obtained from materials scientific simulations and cover two extremes: The first governs the self-organized magnetization of a domain and relates the spatial magnetization structure to the temperature. It is based on randomness and stochastic processes that are simulated using a Monte Carlo method, resulting in structures with very sharp interfaces and discrete changes in time. The other dataset is obtained from a simulation of the evolution of two different phases, which is a smooth and continuous process given by a set of coupled partial differential equations. In both cases, the structure can be represented as an image and the property is a scalar number. These two models are representative for many problems encountered in physics and engineering.

The following machine learning approaches are investigated: (i) a piecewise-constant regression model together with simple features, used as a baseline model; (ii) a support vector regression model with physics-based features; (iii) a non-standard setup of a convolutional neural network approach with three input channels where the original data is accompanied by Fourier and wavelet transformations as additional features; (iv) a combination of a pretrained ResNet and a principal component analysis used as input features for a support vector regression model; (v) several "off-the-shelf" CNNs with different types of pre-training.

2. Data generation: The materials scientific problems and the simulation methods

For all investigations, two datasets will be used that represent the evolution of two different types of microstructures. The datasets are obtained from materials scientific simulations and cover two extremes: while the first is based on stochastic processes where randomness and self-organization are important, the second dataset is obtained from the solution of coupled partial differential equations and describes flow-like smooth and continuous processes. Fig. 1 shows examples of microstructures (the insets A–F) together with the respective property (shown on the vertical axes of the two curves). The underlying theory and the implementation of the simulation models are summarized in Appendix A. In what follows we only describe the datasets.

2.1. The Ising dataset

The two-dimensional microstructure represents the magnetization of a domain with two different magnetic spin directions, indicated by the black and white color in Fig. 1a. The microstructure depends on the chosen temperature, which is considered as the property. The evolution of the system is determined by a Metropolis Monte Carlo algorithm. For a given temperature, the simulation starts with a random distribution of

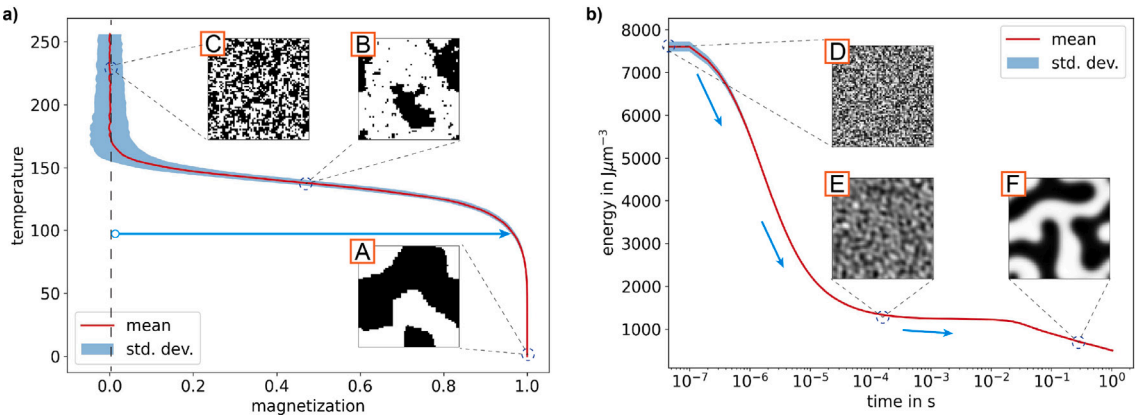

Fig. 1. (a) Visualization of the final state of magnetization from each given temperature. The vertical blue arrow shows an example simulation trajectory of a specific temperature $T = 100$. (b) The evolution of energy and the change of microstructure over time on various simulation run in Cahn–Hilliard model. The blue arrows illustrate the simulation trajectory.

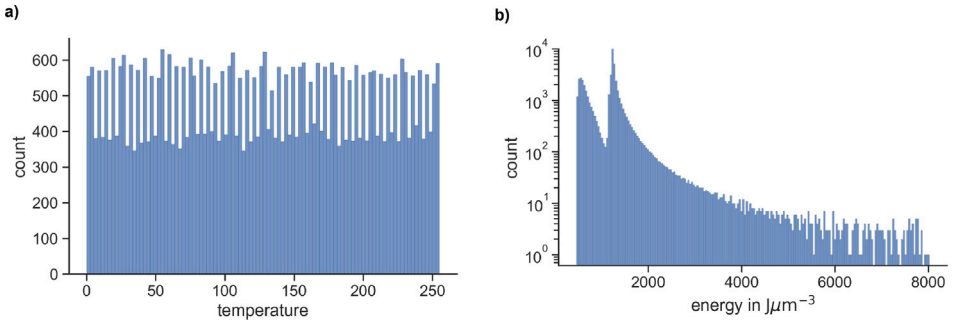

Fig. 2. Histograms of training data distribution for the Ising datasets (left panel) and the Cahn–Hilliard dataset (right panel).

spin values and evolves for a number of steps. The final microstructure is saved as a black and white image, where black (0) represents a negative spin and white (255) represents a positive spin. A single simulation requires a few seconds of computational time and results in a single image. At increasing temperature T above a critical temperature $T_c \approx 2.269$ (in non-dimensional units) the resulting structures become increasingly random. They start the transition towards an ordered state when the temperature is decreasing below the critical temperature T_c; where larger features become visible and the randomness vanishes. Fig. 1a shows example images at three different stages, labeled as A, B, and C together with the (dimensionless) temperature values of 11, 102 and 222, respectively. The temperature values are converted from $0..2T_c$ to 0..255 for the dataset. The whole dataset consists of 50,000 images. The distribution of the corresponding temperature values can be seen in Fig. 2a – a mainly uniform distribution which results in a balanced training dataset.

2.2. The Cahn–Hilliard dataset

The microstructure of the Cahn–Hilliard model represents two phases, e.g., the two different chemical elements of an alloy. The evolution is governed by a set of coupled partial differential equations. This system is more complex than the Ising model because the number of different physical phenomena considered is significantly larger (see Appendix A for further details). This also results in a high computational cost of several hours for a single simulation which might make machine learning-based approaches good candidates as replacements. Three microstructure snapshots are shown in Fig. 1b. Each image represents the microstructure corresponding to the system's energy with values of 7054 J μm^{-3}, 1348 J μm^{-3}, and 589 J μm^{-3}. 20 simulations are performed with random initial microstructure and values in between 0 and 1. In order to reduce the computing time, we use a non-constant time stepping with exponentially increasing time steps. The simulation exhibits two parts: a first part where the energy decays rapidly and a second part, where the energy decreases only slowly. 2000 steps are taken in the first part where the time step increases; the microstructure data is stored at every step. The second part consists of 10,000 steps with a constant step size. The image data are exported at every 10th step of this part of the simulation. Altogether, the dataset contains $\approx 60,000$ images. The distribution of the energy values of the dataset can be seen in Fig. 2b which shows a strong imbalance with significantly more data for the low energy regime. We have chosen to use this kind of sampling because it is close to a "real world" dataset. Creating a uniformly distributed dataset requires significantly more simulation time and is, in most situations, not feasible.

3. Methods

Learning to predict properties from (micro)structures is a regression-type of a problem. For such problems, a large range of different statistical and deep learning methods exist. Our selection of investigated methods is guided by the following considerations: (i) A simple statistical learning method with as simple as possible features should be used as a baseline method; (ii) statistical learning methods can perform very well together with appropriate features; (iii) deep learning approaches typically do not require sophisticated features but sometimes requires larger training datasets. In the following, we start by introducing and deriving the used features. Subsequently, the ML models used in this study are selected.

3.1. Feature engineering

Three different types of features are used throughout this work: a very simplistic feature that does not require any domain knowledge and two physics-based features of different complexity.

3.1.1. A minimalistic, domain-agnostic feature ("grad")

For the statistical learning methods, we start by creating a set of generic features that do not require knowledge of the scientific details behind the data generation. By taking a look at the six microstructure images A–F in Fig. 1 we observe that differences could be related to the total length of the boundary or interfaces between black and white regions. In image analysis, a gradient filter would be the most simple way of extracting such information. Therefore, as a simple feature X, we use the following function

$$X = \frac{1}{n \cdot m} \sum_{i,j} \|\nabla u(i,j)\| = \frac{1}{n \cdot m} \sum_{i,j} \sqrt{\left(\frac{\partial u[i,j]}{\partial x}\right)^2 + \left(\frac{\partial u[i,j]}{\partial y}\right)^2} \tag{1}$$

where $u[i,j]$ is the value of the pixel in row i and column j of the image array u, and m and n are the number of rows and columns, respectively. The gradient of u is approximated by a central finite difference scheme. This feature can be easily used together with a broad range of other datasets as well and does not require further domain-specific knowledge. In the remainder of this work, we abbreviate this feature as "grad".

3.1.2. Feature engineering: The power spectral density ("physFeat1")

Due to the aspects of the underlying physics problem, the microstructure images often exhibit periodicity, symmetry, or rotational invariance. For CNNs, this can be considered through hard or soft constraints. Hard constraints are induced by architecture modifications, such as using group equivariant convolution instead of regular convolutions (Cohen & Welling, 2016). This strategy cannot be transferred to statistical learning methods. Soft constraints, on the other hand, are imposed through training with specific augmentation techniques, e.g., applying specific translations to images to mimic periodicity, random flipping, or rotation of images. However, this does not guarantee that the constraints are exactly enforced, and the physics problem may be violated. As an alternative to applying soft or hard constraints, we will incorporate microstructural constraints using physics-inspired feature engineering.

Which physics-based features are suitable to relate the microstructures to their properties? Both datasets were obtained by enforcing periodic boundary conditions. The resulting properties are both invariant under mirroring and rotation by 90 degrees. Furthermore, the Ising model represents a situation in which a microstructure can undergo a phase transition, i.e., a small change in the temperature results in a significant and qualitative change of the structure. Such a phase transition manifests itself in long-range, collective behavior which is caused by short-range interaction. Roughly speaking, this is related to the fact that at the critical point the system transitions from small fluctuations (the size of the black and white patterns) to large fluctuations, cf. Fig. 1, and there are fluctuations of all wavelengths directly at the critical temperature. As a consequence, physics-based feature engineering should result in descriptors that are able to capture characteristics of the distribution of various wavelengths.

A suitable measure is the PSD, a "multi-scale measure" used in signal processing to describe the distribution of wavelengths. The PSD is obtained via Fourier transformations of an image from which the Fourier amplitudes can be extracted; they are by definition translation invariant. This is followed by radial averaging, making the resultant features invariant to 90-degree rotation and flipping. As a result, two equivalent images from a physics perspective also result in identical PSDs (see, e.g., the applications in the context of quality assessment of fingerprints (Shen et al., 2022) or for statistical analysis of shear bands in Sandfeld and Zaiser (2014)).

Fig. 3 shows the PSD for three different microstructures obtained from the Ising and the Cahn–Hilliard dataset. E.g., in the left column, we observe that a significant portion of the power is located in features with wavenumbers $k \leq 3$, i.e., patterns that are $\geq 1/3$ of the image size. Furthermore, the PSDs roughly exhibits a linear behavior in the double logarithmic plots (with the exception of the second Cahn–Hilliard example). Even though it is obvious that a linear fit is only a rough approximation, it is well able to differentiate between microstructures at different temperatures and energy values. The slope and intercept of the fitted lines after back-transformation from the double logarithmic scale will serve as the features that are used to characterize each microstructure image. The two left scatter plots in Fig. 4 show the temperature and energy, respectively, as a function of these two features for each training dataset. Visual inspection shows that even though the points are clearly localized, the features of the dataset also contain significant scatter. Additionally, the Cahn–Hilliard dataset exhibits an unexpected drop at an energy value of $\approx 1200\,\mathrm{J/\mu m^3}$. This is a side effect of the above line fit which indicates, in this region, a bad representation of the data (cf. the second row of Fig. 3b). In the remainder of this work, we abbreviate these two features describing the microstructure through the approximated PSD as "physFeat1".

3.1.3. Feature engineering: Extended physics-based features ("physFeat2")

The second physics-based set of features is also based on the PSD but is further fine-tuned for accuracy. Fig. 5 schematically shows the feature engineering pipeline that is now explained. A side effect of the strong dimensionality reduction of the previous two features is loss of information. We attempt to make up for this by considering the whole PSD curve and not only a fit of a straight line. Furthermore, additional features are introduced that capture new aspects: instead of working with the PSD of only the original image, we create an additional image array, which includes information about the interfaces, similar to the "grad" feature:

$$\|\nabla u[i,j]\| = \sqrt{\left(\frac{\partial u[i,j]}{\partial x}\right)^2 + \left(\frac{\partial u[i,j]}{\partial y}\right)^2}, \tag{2}$$

where $u[i,j]$ is, as before, the value of the pixel in row i and column j. For each image, the PSD is obtained, which consists of a one-dimensional array of 32 values. The two arrays are stacked, resulting in a 2×32 array. An issue that can arise during computing the PSD are numerical errors due to high PSD values, causing extreme value ranges. As a remedy, feature normalization by logarithmic scaling is performed:

$$\mathrm{PSD}_{\mathrm{scaled}}(x) = \log_{10}(\mathrm{PSD} + 1), \tag{3}$$

where 1 is added to avoid computational problems if the PSD has a value of 0.

3.1.4. Combined image embedding and dimensionality reduction ("CNN-PCA")

The aim of this last set of features is twofold: (i) to achieve high accuracy without having to rely on domain knowledge, (ii) to allow for a regression model that is computationally cheap and fast to train. The CNN-PCA features are obtained by first performing image embedding, followed by a dimensionality reduction, as summarized in Fig. 6a. For the image embedding a ResNet18 with pretrained ImageNet weights (Deng et al., 2009) is used. The CNN is then trained with the images, but for the sake of achieving a low training time, we did not do any fine-tuning of the network. The temporary features are the weights taken from the average pooling layer of the already trained model in the form of a vector with 512 elements. To reduce the dimensionality, a principal component analysis (PCA) is then performed, and the principal components are used as features for the regression model. Using PCA without the image embedding would result in bad performance as the microstructural information is not well represented (see Fig. B.12). The number of components is a hyperparameter that

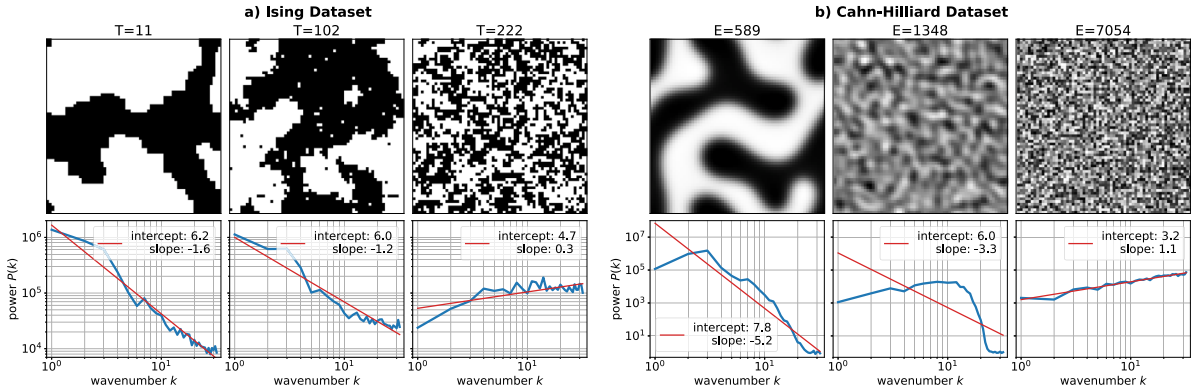

Fig. 3. Microstructure and corresponding PSDs for the Ising dataset (left) and the Cahn–Hilliard dataset (right). The PSD is shown on a double logarithmic scale, where k is the wavelength and $P(k)$ is the normalized power. The thin red line is a linear fit on the double logarithmic scale while the values of intercept and slope of the fitted line are given after backtransformation to the linear scale. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

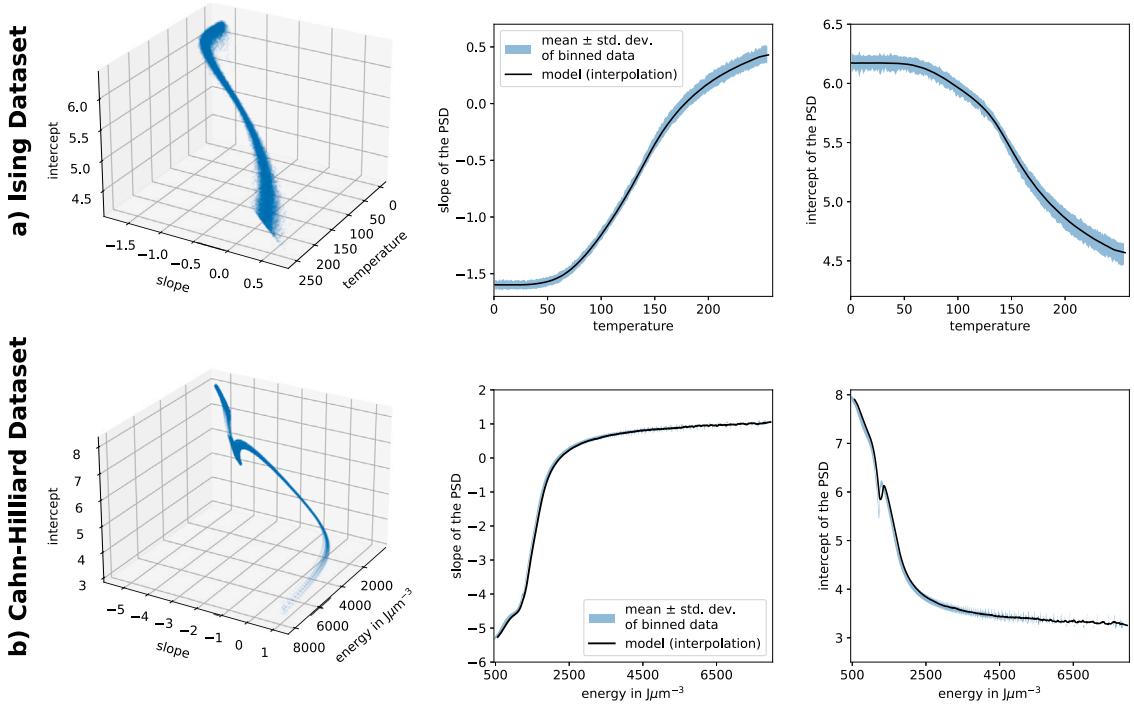

Fig. 4. Visualization of the two scalar features as a function of the investigated property. The top row shows the Ising dataset, the bottom row shows the Cahn–Hilliard dataset. For getting a better idea of the data distributions the blue data points are plotted slightly translucent. The middle and right columns show projections of the feature space and fitted curves, representing the model response. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

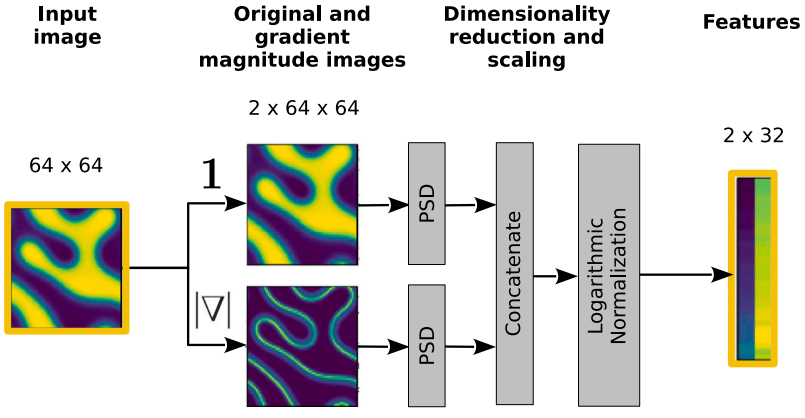

Fig. 5. PSD-based feature engineering pipeline, specifically tailored to the considered type of data.

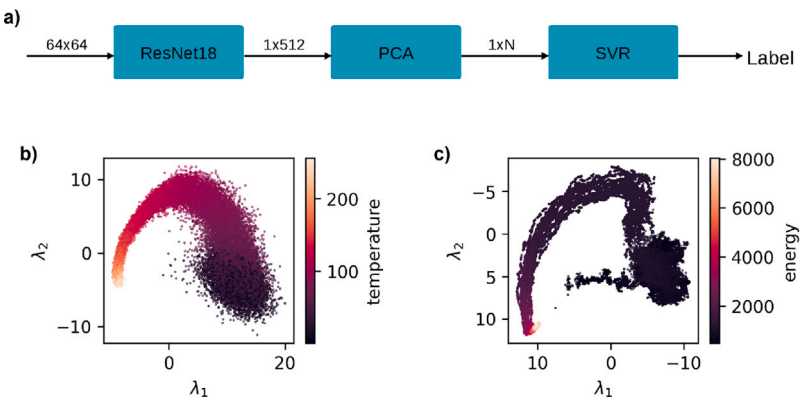

Fig. 6. Summary of the CNN-PCA-SVR model in (a) and visualization of the first two principal components of the ResNet18 weights for the Ising dataset (b) and the Cahn–Hilliard dataset (c).

is investigated in Fig. B.13; the first two components of the Ising and the Cahn–Hilliard dataset are shown in the appendix and in particular in Fig. 6b and c. The data distribution in (b) can be divided into three zones of low, mid, and high temperatures. For low temperatures, the data distribution has a larger variance than that obtained for the mid and high temperature zones. In the high temperature zone, the two components become strongly localized. The principal components for the Ising dataset are shown in Fig. 6b while Fig. 6c shows the components of the Cahn–Hilliard dataset.

3.2. Overview of the used machine learning models

Altogether four fundamentally different ML-based approaches are studied and introduced subsequently. An overview together with the used abbreviations is given in Table 1. The first two methods are statistical ML methods and use the above introduced features; grad-PCR will be used as a baseline model. Note, that the second type of physics-based feature, physFeat2, requires a regression method that is able to make use of the complex features. physFeat2 together with the very simple piecewise constant regression showed a performance below the baseline model and therefore was not considered here. The last two ML methods are CNN-based learning approaches where the first method, CNN, does not require any preprocessed features. The second method, MuCha-CNN, is tailored to the specifics of images and indirectly makes use of some of the information that is also contained in the PSD. Subsequently, all models are briefly introduced.

3.2.1. Piecewise Constant Regression (PCR)

Piecewise constant regression (PCR) is a particular type of regression tree and is one of the simplest ML methods for regression. Here, it is used either with the gradient-based feature (grad) or with the slope and intercept from the PSD (physFeat1). The training consists of (a) binning the property data with a bin size of $\Delta = 1$ (i.e., temperature for the Ising dataset and energy for the Cahn–Hilliard dataset) and (b) "fitting" piece-wise constant approximations to the binned feature data by computing the mean values. The motivation for choosing this method in combination with the grad features is to use it as a baseline model (grad-PCR). The thin solid line in the middle and right panels of Fig. 4 shows the model responses. Predicting temperatures or energies works as follows: for a given microstructure, compute the PSD, and obtain slope and intercept values of the fitted line, as explained above. Then, search for the nearest point in the learned slope-intercept space and get the corresponding temperature or energy. Slope and intercept have different physical dimensions, and therefore, dimensionless scaling of the data to the range $[0, 1]$ is performed before the distance can be computed.

Table 1
Overview of the investigated features and ML methods.

Abbreviation	Features	ML model
grad-PCR	Simple sum of the norm of the gradient, Section 3.1.1	Piecewise constant regression (PCR), Section 3.2.1
physFeat1-PCR	Intercept & slope of the fitted to the power spectral density	
grad-SVR	Simple sum of the norm of the gradient, Section 3.1.1	Support vector regression (SVR), Section 3.2.2
physFeat1-SVR	Intercept and slope of the line that was fitted to the power spectral density	
physFeat2-SVR	Highly-specialized, physics-based preprocessing pipeline for obtaining feature, Section 3.1.3	
CNN-PCA-SVR	Image embedding (ResNET) and principal components analysis for automated feature extraction, Section 3.1.4	
CNN	(input for CNN: the image w/o further preprocessing or feature extraction)	Several CNN architectures; 2 pretraining approaches, Section 3.2.3
MuCha-CNN	(input for CNN: the image itself, as well as a Fourier and a wavelet transformation of the image)	ResNet34 with simple training protocol, Section 3.2.4

3.2.2. Support Vector Regression (SVR)

Support vector regression (SVR) (Vapnik, 2000) is one of the commonly used methods for regression in statistical learning, which performs well as long as the dataset is not too large (several tens of thousands of data records are still feasible). We used the implementation of the epsilon-insensitive SVM provided by scikit-learn (Pedregosa et al., 2011). For the two simpler features, grad and physFeat1, determining the most suitable hyperparameter was done manually by trying 3–5 different values, starting from the scikit-learn default values. For the combination with the more complex feature physFeat2 the goal was to achieve an as high as possible accuracy. Therefore, a systematic hyperparameter search was performed. The ranges of the hyperparameters and final selected values are available in Appendix B.1.

3.2.3. Convolutional Neural Network approaches (CNN)

As DL networks of different depth and degrees of complexity – without further alterations or adaptions – we used a ResNet18, a ResNet152, a DenseNet121, and an EfficientNet-B0. All of these architectures have achieved very good results for ImageNet (Deng et al., 2009) classification problems, where EfficientNet-B0 performed particularly well (Tan

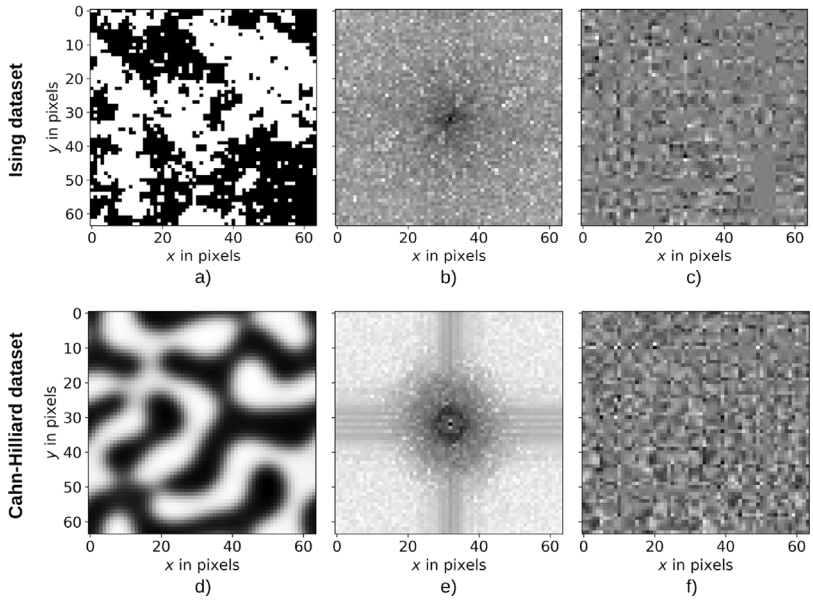

Fig. 7. An example of three channel images of the Ising dataset (top row) and Cahn–Hilliard dataset (bottom row), respectively: (a) and (d) original image; (b) and (e) fast-Fourier transform of the image; (c) and (f) wavelet transform of the image.

& Le, 2019). Each of these models will be used with two types of initial weights: (i) we use pretrained models with weights taken from ImageNet, (ii) we train the models that were initialized with random weights from scratch. Transfer learning is used by freezing the whole pretrained model and then training only the final layer. Once the layer has been optimized, we train the whole model using a learning rate of 10^{-4}. The training data itself was standardized by subtracting the mean and dividing by the variance. Both datasets are split into training and test data, and 20% of the training data is kept as validation data. Once the model has been optimized, we evaluate the model performance on the test data. During the training, we use early stopping such that if the validation loss does not improve for 20 epochs, we stop the training. Furthermore, the training starts with a learning rate of 10^{-3} and is reduced to half of its value with a minimum of 10^{-5} when the monitored metric does not change anymore. All architectures are implemented using the Pytorch framework (Paszke et al., 2019).

3.2.4. A Multichannel Convolutional Neural Network (MuCha-CNN)

Convolutional neural networks (LeCun, Bengio, & Hinton, 2015) are a special type of deep neural network that have been widely used to process image data. One of their main aspects is convolutional layers, which act as feature detectors, making classical feature engineering obsolete. The input for CNNs may consist of image data with three intensity channels (one for each of the colors red, green, and blue). Instead of the three color channels, we use one layer for the grayscale image; the second and third channels contain the magnitude of the Fast Fourier transformation (FFT) and the wavelet transformation of the grayscale image, respectively. Since the PSD is also based on FFT, the information contained in the additional images is related to the PSD and could be considered as an additional feature. Further information is given in Appendix C. Examples of the content of the three channels are shown in Fig. 7.

We used residual networks with 34 layers (ResNet34) (He, Zhang, Ren, & Sun, 2016), which is implemented using the Pytorch framework (Paszke et al., 2019). The size of each input channel is 64×64 pixels, and the respective ranges are scaled such that all values are in between 0 and 255. Fig. 7b shows the magnitude-spectrum from a complex array obtained by applying the Fourier transform using Numpy (Harris et al., 2020). Fig. 7c shows the resulted image obtained by applying the wavelet transform using the Python package PyWavelets (Lee, Gommers, Waselewski, Wohlfahrt, & O'Leary, 2019). This image is the so-called diagonal detail which results from vertical and horizontal highpass filtering. Since our goal is not to study the effect of various wavelet functions, we pragmatically chose one of the default functions from the "sym-family" functions. For training, the weights of the multichannel CNN are randomly initialized. Then, the network is trained for 100 epochs with a fixed learning rate of 0.001 and momentum equal 0.9. As an optimizer, we used stochastic gradient descent and the L1 loss.

4. Results and discussion

All models were trained on a training dataset and evaluated on a separate testing dataset. The testing dataset was identical for all models and comprises for the Ising dataset of 1000 images covering the whole temperature range (≈2% of the total dataset) and 6000 images for the Cahn–Hilliard dataset obtained from two full, individual simulations (≈10% of the total dataset). These test datasets are kept entirely separated from the training process. The performance of the models was assessed by the RMSE and the R^2 score. Results for different models are shown in Table 2. For the Ising dataset, all approaches show prediction accuracies that are roughly in the same order of magnitude (RMSE $= 12.15 \pm 4.15$). The baseline model (grad-PCR) has only a slightly worse RMSE than the other models, and the best model(s) achieve half of the RMSE of the baseline model. For the Cahn–Hilliard dataset, the differences between models are more pronounced (RMSE $= 171.5 \pm 160.5$). In particular, the gradient-based baseline model grad-PCR, as well as grad-SVR, perform rather badly with RMSE values of 332 and 262. SVR with the more detailed, physics-based features, physFeat2-SVR, and the multichannel CNN perform best, followed by the vanilla ResNet18. Note that the values cannot be directly compared to those of the Ising model, but for both datasets, the best performing models are physFeat2-SVR, the CNN (ResNet18) with training from scratch, and MuCha-CNN.

To understand which details of the datasets are more difficult to predict than the others, a confusion matrix for all models, and both problems is shown in Fig. 8 for the true vs. the predicted properties of the testing data. In the following, we first discuss the general model performance and then discuss the advantages and disadvantages of all individual models.

Table 2
Root mean square error (RMSE) and the R^2 score of the predictions of all models and for both datasets.

	grad-PCR	physFeat1-PCR	grad-SVR	physFeat1-SVR	physFeat2-SVR	CNN-PCA-SVR	CNN (ResNet18)	MuCha-CNN
Ising (RMSE):	16.3	14.5	14.1	12.3	8.5	10.0	8.0	8.9
Ising (R^2):	0.951	0.961	0.963	0.972	0.987	0.981	0.988	0.985
CH (RMSE):	337	44	262	151	12	72	22	11
CH (R^2):	0.674	0.994	0.802	0.934	1.0	0.985	0.999	1.0

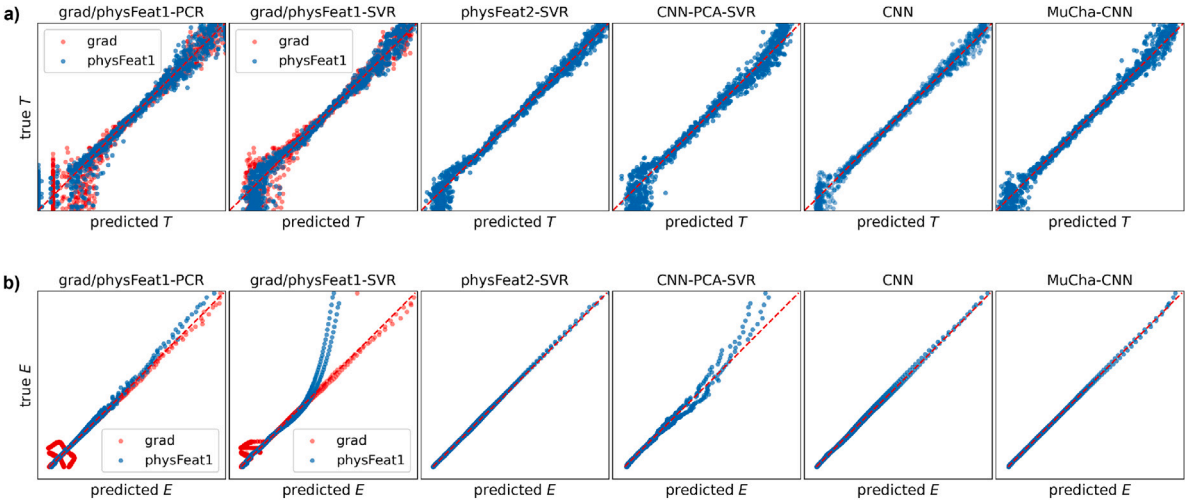

Fig. 8. Confusion matrices of all models and used futures for (a) the Ising dataset and (b) the Cahn–Hilliard dataset. The red marker illustrate the results when the simple norm of the gradient was used as a feature. The dashed red line indicates the values for a perfect prediction. Vertical and horizontal range from each subfigure of (a) and (b) are 0..256 and 0..8000, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4.1. General analysis of the prediction errors

Analysis of the Ising dataset. Fig. 8a shows the ground truth temperature vs. the predicted temperature for the Ising dataset. We observe for the baseline model that the accuracy for intermediate temperatures is better than that for low or high temperatures. For these extreme temperatures, the magnetization values of the corresponding microstructures are either 0 or 1. Taking a look at how all models perform in the low temperature regime at $T \lessapprox 10$, we see that MuCha-CNN performs best, however, all models' predictions are affected by errors where predicted temperatures of low temperature microstructures are too high. In the high temperature regime, MuCha-CNN exhibits a slightly larger scatter as, e.g., compared to the regular CNN approach which performs better than all other models for high temperatures.

The low temperature regime is more difficult to predict because the changes in the images that correspond to changes in the temperature are smaller than those in the higher temperature regime. The underlying physical reason for this is that low temperature results in considerably slowed down dynamics of the system. The difficulties of the predictions in the high temperature regime have a different reason. There, the microstructure tends to become increasingly random, and the temperature differences are related to increasingly small remnants of the ordered structure.

What is the influence of using the "physical features" (i.e., the power spectral density)? In the first two confusion matrices from the left, we used both the simple gradient-type features as well as the PSD-features (shown as the red and blue markers). There, we see that the use of physical features has the most pronounced influence on low temperature predictions. Furthermore, the complex SVR model benefits well from more complex features; in particular, physFeat2-SVR qualitatively shows one of the best confusion matrices with balanced performance for all temperatures.

Analysis of the Cahn–Hilliard dataset. Models trained with the Cahn–Hilliard dataset have the worst RMSE for the two gradient-feature-based methods which also shows in the confusion matrix in Fig. 8b (the red marker in the two leftmost plots). In particular for low energy values one observes artifacts that result in different distributions for the two simulations of which the testing dataset consists. The simplistic features are not able to capture the microstructural differences at low energy values properly and are not able to generalize to microstructures from other simulation runs. Using the physical PSD features, this behavior changes, and in particular the physFeat2-SVR model performs nearly perfectly.

In fact, there is only one model that performs better: the MuCha-CNN, which operates with two additional types of images obtained from a wavelet and Fourier transform. These two additional channels contain information about different wavelengths and spatial structures and, therefore, are related to the physical features. The CNN (ResNet18) achieves nearly identical prediction performance.

From the confusion matrix, we see that the CNN-PCA-SVR model with 150 principal components performs worse than all other models. Generally, CNN-PCA-SVR is more accurate in the low energy regime as compared to the higher energies. We also can clearly see differences between the two simulation runs contained in the testing dataset. This also might be an indicator that the training dataset is not sufficiently large enough for this method, and the model fails to generalize properly. Since the physFeat2-SVR model with highly specialized features shows nearly perfect predictions, it follows that the cause of the bad performance is the feature extraction by CNN and PCA. Principal component analysis is a linear method and, therefore, might be limited in terms of the features that can be represented.

A particular challenge of this dataset is that it is strongly imbalanced with regards to the energy distribution, cf. Fig. 2b. This is the reason why the RMSE value of CNN-PCA-SVR is better than those for the grad-PCR and grad-SVR, even though the confusion matrix of these two methods suggests the contrary. The reason for this is the high amount of data for lower energies, where these two models perform particularly badly. The imbalance of data is also the reason why almost all models perform better in the low energy range, as there is a large amount of data. Furthermore, also the microstructural patterns in the images at

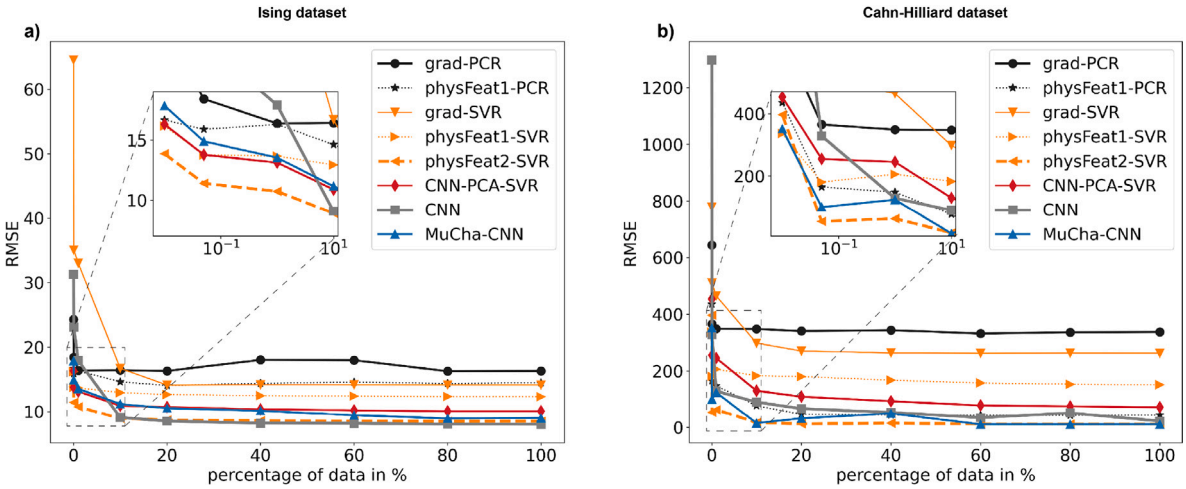

Fig. 9. Prediction performance for different sizes of training datasets for the Ising datasets (left) and the Cahn–Hilliard dataset (right). The x-axis of the insets are plotted in log-scale to reveal the low percentage area. Markers show the performance for 0.1%, 0.5%, 1%, 10%, 20%, 40%, 60%, 80% and 100% of the total training data for both datasets.

low energies are clearly developed and distinct. Predictions in the high energy regime are slightly less accurate, which is partially due to the smaller amount of data. Additionally, the variance among the training examples is larger in this energy regime.

4.2. Influence of the size of the training dataset

An important goal is that models are able to achieve a high prediction accuracy. However, in many practical situations, it can be equally important to achieve good results with a small dataset, e.g., because more data might not be available or computationally too costly to obtain. The training dataset of the Ising and Cahn–Hilliard model contains 49,000 images and 54,000 images, respectively. To see the impact of the size of the datasets, the RMSE results for training with different size percentages are compared in Fig. 9. For both datasets, there is a pronounced drop in the RMSE values when the percentage of training data is increased from 0.1% to 10%; from 10% to 100%, there is only a slight decrease. The corresponding confusion matrices for all sizes can be seen in Appendix C. In general, using at least 10% of the original datasets is a good compromise. These are approximately 5000 images. For small datasets of 50 images, the SVR methods clearly outperform CNN-based approaches for the Ising model. For the Cahn–Hilliard model, at least 50 images are required, and only physFeat2-SVR is able to perform well. Additionally, the two CNN-based methods show acceptable but less robust performance as they are still dependent on the chosen testing samples.

4.3. Required computational and feature engineering effort

The time required for training (including computing the feature values) and for predicting for each of the ML approaches as a function of the size of the training dataset are shown in Fig. 10. All times were measured on an 8-core workstation with Intel Core i9-10850K CPU, 32 GB RAM, and a Nvidia GeForce RTX 3060 GPU with 12 GB RAM. We observe a huge difference in the times required for training the models: the baseline model grad-PCR is the simplest model; it only requires binning and averaging of the training data and, therefore, is the fastest to train. Obtaining the gradient of the image is computationally cheaper than computing the PSD – for the used implementations, roughly by a factor of two. The PCR models require very little training time (even though the implementation was not optimized): it is almost the same for every fraction of the Ising dataset and takes around 1.2 s. For training, excluding the computation of the features and predicting it takes 0.5 s. For the Cahn–Hilliard dataset, the computational times show only small deviations from the times for 10% of the data: it is around 33 s for training and less than 10 s for predicting.

The SVR models require substantially more training time as the size of the datasets increases. The training time additionally depends strongly on the features. For the simple grad-SVR approach, the training time without feature calculation increases exponentially with the size of the dataset; it takes around 5400 s and 2300 s for 100% of the Ising and the Cahn–Hilliard dataset, respectively. The reason is that the model complexity (the number of support vectors) increases with larger datasets. This also shows in the more than exponentially increasing prediction times.

CNNs and MuCha-CNN require, by comparison, the most time for training. Additionally, this also depends strongly on the complexity of the chosen model architecture. Furthermore, factors such as batch size can significantly impact the training and prediction times. E.g., even though the training process used in CNNs does not use any prepossessing that could increase the train time, we used a batch size of 32 during the training and employed early stopping where we waited for 20 epochs to see if the results were improved. These are two examples where the training time could have been reduced if a loss of accuracy would be acceptable.

The last aspect is the effort that is required to engineer the features for the PCR and SVR models. While the physFeat2-SVR model achieves for both datasets superior prediction accuracy, the identification of physically reasonable features and the concomitant hyperparameter study might outweigh the high accuracy. This also holds for MuCha-CNN, even though no hyperparameter tuning was performed. In case there are no obvious physical features, one of the relatively shallow CNNs is a good choice. However, the amount of required training data is somewhat higher than that of other models. A fast-to-train model with still acceptable accuracy is the CNN-PCA-SVR. It is also the model that makes predictions considerably faster than all other models (except for PCR for larger datasets).

4.4. General discussion

Feature engineering methods, which reduce features' dimensions and extract the essential information from data, depend strongly on the nature of data and may need particular care (Nguyen & Holmes, 2019). Therefore, studying, preprocessing the data, and choosing an appropriate technique requires dedicated effort. The PSD features that we choose for several approaches, such as physFeat1-PCR/-SVR, or physFeat2-SVR can be useful for various types of problems, as was shown based on the two datasets investigated in this work. Clearly, the next question is, if these approaches can also be used for data

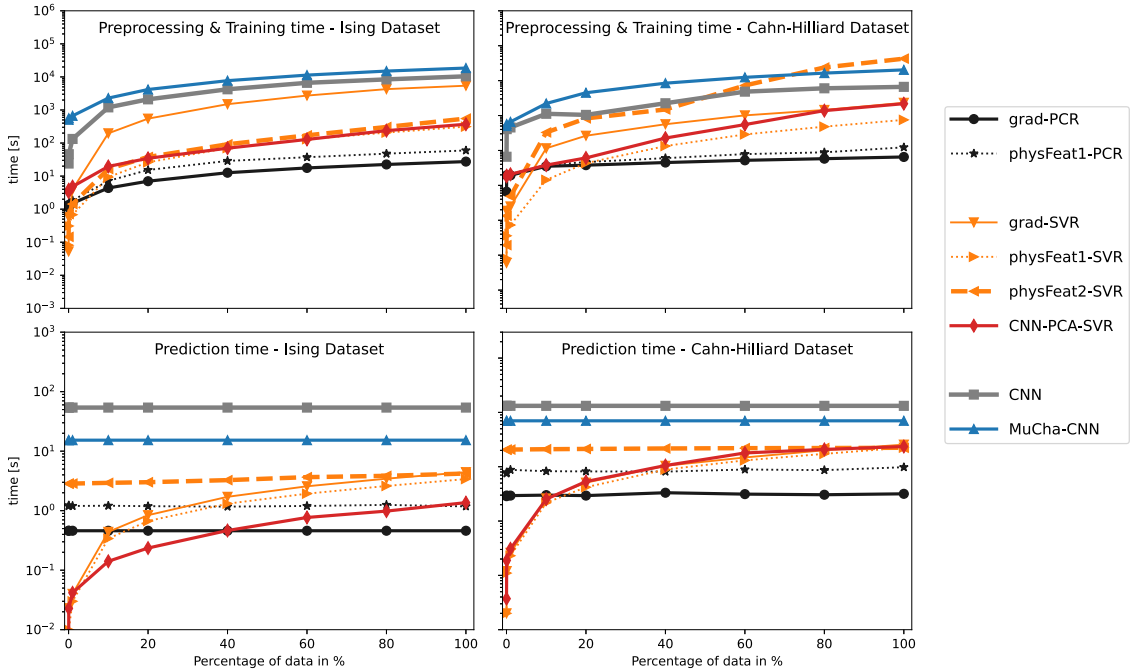

Fig. 10. Computational time for training (including computation of the input features for PCR and SVR) and predictions of all models and both datasets as a function of the size of the training dataset. The vertical axes are scaled logarithmically.

that visually looks very different. To answer this question a very different kind of simulation dataset was investigated, which is from the domain of computational fluid dynamics. There we predicted the Reynolds number (as "property") from simulated images of fluid flow (as "structure"). In Fig. E.20 the results are shown. The overall accuracy is slightly lower as compared to the two mainly investigated datasets. The MuCha-CNN method is independent of the nature of data since it can extract the most important features of the image data without having to rely on engineered features (see Figs. 8 and E.21) and therefore performed considerably better. Nonetheless, most of the methods still perform rather well. Thus, using the PSD features for building machine learning models in situations where the data mostly consists of fluctuations and patterns of various wavelenghts is a reasonable approach.

5. Conclusion

We investigated the problem of learning and predicting the so-called structure–property relation, i.e., the mathematically non-trivial mapping from a two-dimensional structure to a scalar value. For this, we compared the predictive power of several machine learning models – statistical learners as well as deep learning-based models – for predicting the properties of microstructure from the Ising model and the Cahn–Hilliard model. One of the challenges was to cope with strongly imbalanced datasets in case of the Cahn–Hilliard model.

We found that statistical learning approaches that include a physics-based feature engineering may outperform more generic approaches, e.g., CNN-based models both in terms of accuracy and train/prediction time. The improved accuracy, in particular for smaller datasets, is partially due to the possibility of automatically introducing translational, 90-degree-rotational, and mirroring invariances through the engineered features, but additionally, the reduction of the dimensionality of the feature space is beneficial for the computational efficiency. However, the feature engineering requires a certain amount of domain knowledge and the overall effort is generally large.

Comparing such ML approaches to classical forward simulations, we found that even predicting properties that usually require a large numerical simulation effort can be learned and reliably predicted. Nonetheless, generalizing the models such that they are applicable for different domain sizes or different boundary conditions is one of the current shortcomings and field of active research in various communities.

CRediT authorship contribution statement

Binh Duong Nguyen: Software, Writing, Formal analysis. **Pavlo Potapenko:** Software, Writing, Formal analysis. **Aytekin Demirci:** Software, Writing, Formal analysis. **Kishan Govind:** Software, Writing, Formal analysis. **Sébastien Bompas:** Software, Writing, Formal analysis. **Stefan Sandfeld:** Conceptualization, Software, Writing, Formal analysis, Supervision, Funding acquisition.

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 and code availability

The two datasets, the data of all shown plots together with the code for reproducing the plots, as well as supplementary materials, are available at https://gitlab.com/computational-materials-science/public/publication-data-and-code/2024_Nguyen_et_al_MLWA_ML-based-predictions-of-microstructure-properties An additional snapshot of the repository with data and code has been published at the permanent DOI https://doi.org/10.5281/zenodo.10821847.

Acknowledgments

Financial support from the European Research Council through the ERC Grant Agreement No. 759419 is gratefully acknowledged.

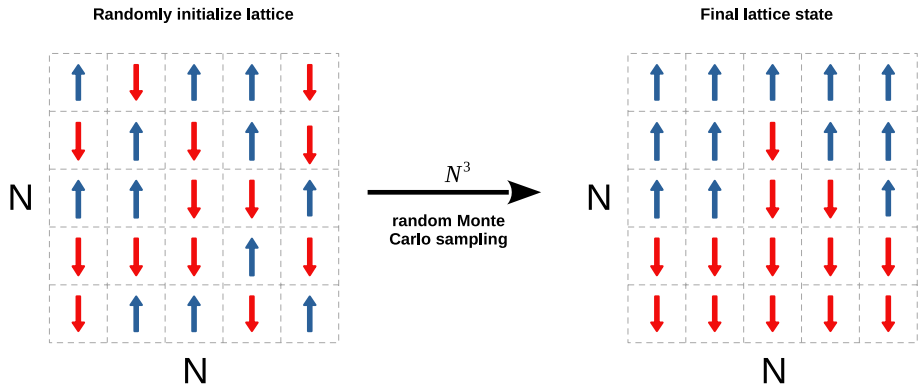

Fig. A.11. Graphical representation of the Ising model. The spins are first randomly placed in the $N \times N$ lattice. The evolution of the lattice follows the Monte Carlo algorithm for a fixed temperature T. In this example, the temperature is below the Curie temperature, and the final state is organized.

Appendix A. Explanation of the data generation by the simulation models

In the following, we introduce the used simulation models together with the underlying physical behavior. As the emphasis of this work is not on simulations, please refer to the given references for further details.

A.1. The Ising model – Statistical mechanics approach to ferro-magnetism

The Ising model is a theoretical model developed to describe ferro-magnetism by W. Lenz in Lenz (1920) and solved for the 1D case by E. Ising in Ising (1925). In the 2D case, magnetic dipoles are located in the center points of a $N \times N$ grid. The Hamiltonian governs the total energy of the system, which, in the case of periodic boundary conditions is given by:

$$H = -\sum_{\langle i,j \rangle} J_{ij}\sigma_i\sigma_j - \mu \sum_i h_i\sigma_i \,, \tag{A.1}$$

where $\langle i, j \rangle$ is the set of all the nearest neighbors of i, j, and J_{ij} is the coupling force between the ith and jth magnetic dipole, $\sigma \in \{-1, 1\}$ is the sign of the magnetic dipole at a given site, μ is the magnetic moment and h is an external field apply to the lattice. In our case, we simplify Eq. (A.1) by setting $h = 0$ and $J = 1$ for all magnetic dipoles

$$H = -J \sum_{\langle i,j \rangle} \sigma_i\sigma_j \,. \tag{A.2}$$

The magnetization of the system, which is a quantitative measurement of the excess dipole signs is given as,

$$\langle M \rangle = \frac{1}{N^2} \sum_i \sigma_i \,. \tag{A.3}$$

Simulation set up. To simulate this system, we use the Metropolis algorithm. First, we randomly initialize the $N \times N$ dipoles, where we chose $N = 64$. Then, we randomly select one site $s(i, j)$, $i, j \in [1, N]$ and flip the corresponding magnetic dipole by changing its sign. We then calculate the energy contribution of the new configuration. Due to the periodic boundary conditions, the nearest neighbors of the first and last magnetic dipole in the lattice are the Nth and first magnetic dipole, respectively. If the energy of the new configuration is smaller than the previous one, we keep the new configuration. Alternatively, if the energy of the new configuration is greater than the previous configuration, we only keep it with a probability $p = e^{-\beta E}$, $E = H_p - H_n$ and $\beta = \frac{1}{k_b T}$ where T is the temperature of the system and k_b the Boltzmann constant, set to 1 for simplicity. The critical temperature, or Curie temperature (T_c), is defined as,

$$T_c = \frac{2J}{k_B \ln(1+\sqrt{2})} \approx 2/\ln(1+\sqrt{2}) \approx 2.269 \,. \tag{A.4}$$

We repeat those steps until we reach a stopping criterion. An example of such process is given in Fig. A.11. We choose N^3 as a stopping criterion for the simulation, where N is the size of the lattice. The idea behind this is that every site of the $N \times N$ lattice is visited approximately N times so that the information has sufficient time to travel through all the lattices.

A.2. The Cahn–Hilliard equation – Evolution of phase separation in binary systems

In this work, we present a simple case of phase separation evolution in a binary system, including the elasticity by a coupling approach between a phase field model and an eigenstrain problem. The phase field model, which is motivated by Cahn (1961) for spinodal decomposition in binary alloys, is computed by solving the Cahn–Hilliard equation. There, the evolution of the composition field c is governed by the minimization of the free energy,

$$\frac{\partial c}{\partial t} = M_c \nabla^2 \frac{\delta E}{\delta c}, \tag{A.5}$$

where E is the free energy of the system and M_c is a homogeneous and isotropic interface mobility coefficient. The free energy density consists of the potential energy density (Φ^{bulk}), gradient energy density (Φ^{grad}), and elastic energy density (Φ^{el}) and is given as,

$$\Phi = \Phi^{\text{bulk}} + \Phi^{\text{grad}} + \Phi^{\text{el}}, \tag{A.6}$$

where $\Phi^{\text{bulk}} = c_0 c^2 (1-c)^2$, $\Phi^{\text{grad}} = \frac{1}{2} k_c |\nabla c|^2$ and $\Phi^{\text{el}} = \frac{1}{2}\boldsymbol{\sigma} : \boldsymbol{\varepsilon}^{\text{el}}$. The two constants c_0 and k_c are the density scale and the gradient energy density, respectively. $\boldsymbol{\sigma}$ is the stress tensor and $\boldsymbol{\varepsilon}_{\text{el}}$ is the elastic strain tensor. The energy functional is then

$$E = \int_\Omega (\Phi^{\text{bulk}} + \Phi^{\text{grad}} + \Phi^{\text{el}}) \mathrm{d}\Omega = \int_\Omega (c_0 c^2 (1-c)^2 + \frac{1}{2} k_c |\nabla c|^2 + \frac{1}{2}\boldsymbol{\sigma} : \boldsymbol{\varepsilon}^{\text{el}}) \mathrm{d}\Omega \,. \tag{A.7}$$

The eigenstrain problem is fulfilled through the mechanical equilibrium,

$$\nabla . \boldsymbol{\sigma} = 0, \tag{A.8}$$

with the stress tensor $\boldsymbol{\sigma} = \boldsymbol{C} : \boldsymbol{\varepsilon}^{\text{el}}$ and the stiffness tensor $\boldsymbol{C}$. The elastic strain tensor is defined as

$$\boldsymbol{\varepsilon}^{\text{el}}(\boldsymbol{u}, c) = \boldsymbol{\varepsilon}(\boldsymbol{u}) - \boldsymbol{\varepsilon}^{\text{iel}}(c) \,, \tag{A.9}$$

where $\boldsymbol{\varepsilon} = \frac{1}{2}(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T)$ is the total strain tensor from displacement $\boldsymbol{u}$ caused by lattice distortion and $\boldsymbol{\varepsilon}^{\text{iel}}$ is the non-elastic part that causes the eigenstrain.

Simulation set up. The coupling partial differential equations are implemented and solved in FEniCS, an open-source Python library package (Langtangen & Logg, 2017). The width and height of the domain are both 20 μm. The elastic stiffness constants for the isotropic material model are $C_{11} = 198\,\text{GPa}$, $C_{12} = 138\,\text{GPa}$ and $C_{44} = 97\,\text{GPa}$. The density scale is $c_0 = 50 \times 10^{-6}\ \text{J}\ \mu\text{m}^{-3}$ and the gradient energy density is $k_c = 10\ \text{J}\ \mu\text{m}^{-1}$. Periodic boundary conditions are used.

We start with the initial values drawn from a uniform random distribution from the range between 0 and 1. Before the two phases are separated clearly into 0 and 1, there is a "mixing state" (also called binodalor unstable state) where two phases are mixing (the values of concentration c are neither 0 nor 1 yet but in the range between 0 and 1) which cause the non-convexity of the energy, and this is where we see a kink in the energy curve. This "mixing state" has a duration that depends on how the parameters are chosen (the strong or weak influence of the gradient term and/or the chemical term on the total energy of the whole system). After the kink, the two phases become clearly separate, where the values are either 0 or 1. It then gradually lowers the energy by merging the phases. The similar behavior of the energy curve and phenomenon can be referred to Kim and Lee (2021).

The simulation is divided into two parts: the first part, where the energy decays rapidly, and the second part, where the energy decreases only slowly. We run the simulation in the former with 2000 steps with Δt values spaced evenly on a log scale (minimum and equal to 10^{-7} s at the beginning, maximum and equal to roughly 10^{-4} s at the end); and in the latter with 10,000 steps with constant $\Delta t = 10^{-4}$ s. The image data are exported at every step of the first part and every 10th step of the second part of the simulation. Therefore, there are roughly 60 000 images in the produced dataset.

Appendix B. Additional information and discussion of the machine learning models

B.1. Hyperparameter search for the SVR model with "physFeat2" features

In this work, the support vector regression (SVR) implementation of scikit-learn (Pedregosa et al., 2011) is used, which is based on radial basis function kernels. It requires three hyperparameters: a regularization term C, a kernel coefficient γ, and the distance within the epsilon-tube, ϵ, where points are not penalized. To find the optimal values of these hyperparameters, we used the tree-structured Parzen Estimator (TPE) (Bergstra, Bardenet, Bengio, & Kégl, 2011), where the authors also mention how crucial the tuning of SVR hyperparameters is for the model performance. The ranges of SVR hyperparameters are provided in Table B.3, where C is the regularization term (box constraint), γ is the kernel coefficient and ϵ is the distance within the epsilon-tube where points are not penalized. The TPE search is performed 1024 times, followed by hand-tuning. The model performance is evaluated by performing 5-fold cross-validation on training data, consequently no test data was seen throughout the hyperparameter search. The resultant final parameters are provided in the last two columns in Table B.3. The box constraint C provides insight into the characteristic differences between the two datasets: the larger C, the stronger the penalty for not reaching the ϵ-tube region. At the same time, a smaller C leads to a stronger regularization. Theoretically, it is impossible to exactly predict the temperature of an image from the Ising dataset based on only one snapshot of the whole simulation trajectory because the temperature information is related to the probability of flipping the sign of the spin of one of the elementary magnets. As a result, a non-uniqueness could be observed: there may exist near-identical images at different temperature values. The non-uniqueness is countered by the regularization of the SVR model, which, therefore, has to be higher for the Ising dataset as compared to the Cahn–Hilliard dataset.

Table B.3
PSD-SVR model hyperparameters: the columns "Range" and "logarithmic distribution" denote the parameter range/values that constrain the hyperparameter optimization, the last two columns show the final parameter for the two datasets.

Hyperparameter	Range	Logarithmic distribution	Ising	Cahn–Hilliard
C	$(10^{-5}, 10^{6})$	True	10^2	10^5
γ	Scale, auto	False	1/64	1/64
ϵ	(0.1, 0.5)	False	0.12	0.4

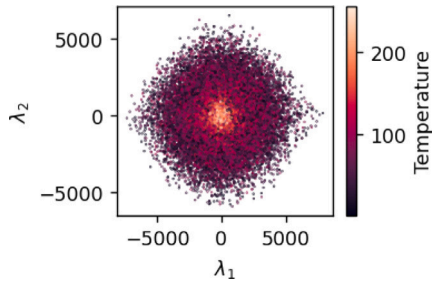

Fig. B.12. The first two principal components directly obtained by using the pixel values of the input images as features for PCA.

B.2. Support vector regression with image embedding and PCA (CNN-PCA-SVR)

The combination of a ResNet18 and the principal component analysis serves as a problem-agnostic way of computing features that can then be used for support vector regression. Using *only* PCA turned out to result in features (the principal components) that do not contain sufficient information. The first two components are shown in Fig. B.12. The strong localization for high-temperature data in the latent space is caused by the strong randomness contained in the images. For lower temperatures, this is different, but the radial symmetric distribution mainly captures the randomness of the images and not the patterns with different wavelengths. Therefore, a CNN was used to extract features (i.e., the weights of the last hidden layer). Those were then used as input for a PCA. The number of used principal components is a hyperparameter that was chosen to keep the computational cost as low as possible and at the same time to achieve an as good as possible prediction performance. We start by taking a look at the variance explained as a function of the number of principal components in Fig. B.13. The plot shows that the variance explained quickly increases for up to 15 principal components. This explains the rapid decrease in the RMSE when the number of components is increased from 1 to 15 (the influence of the number of components for both datasets are shown in the right figure of Fig. B.13).

In Fig. B.14, confusion matrices are given for changing the number of principal components for both datasets.

Ising dataset. When there is only one principal component, the model is not able to distinguish images from each other in the high and low temperature zones. In the intermediate temperature zone, the predictions are slightly better. However, the model can predict the images around the critical temperature, T_c, successfully by reducing the information from 64 × 64 arrays to only one scalar value. Therefore, if the objective is to predict the phase transformation, this would be a highly efficient model. Already starting from two principal components, the confusion matrix looks very similar to Fig. 8.

Cahn-Hilliard dataset. For this dataset, the behavior is different: about 25 components are required until a similar behavior as in Fig. B.14 is observed. For all numbers of components, the low-energy regime is predicted significantly better due to the presence of clearly developed patterns in the images. This can also be deducted from Fig. 6c: the low energy zone is very extended in the latent space, which implies better

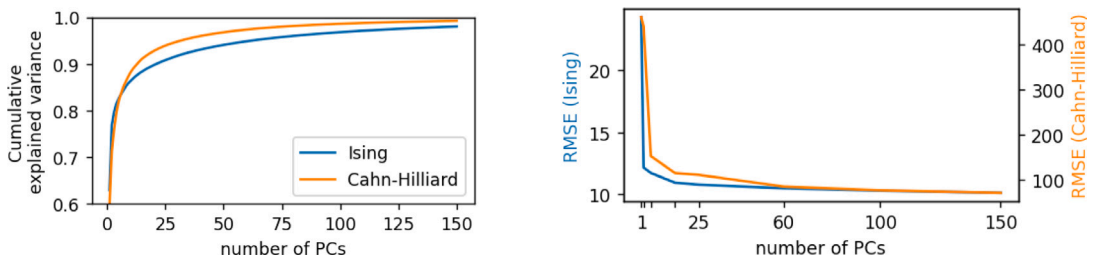

Fig. B.13. Scree plot showing the cumulative explained variance (left) and the RMSE error (right), both as a function of the number of principal components.

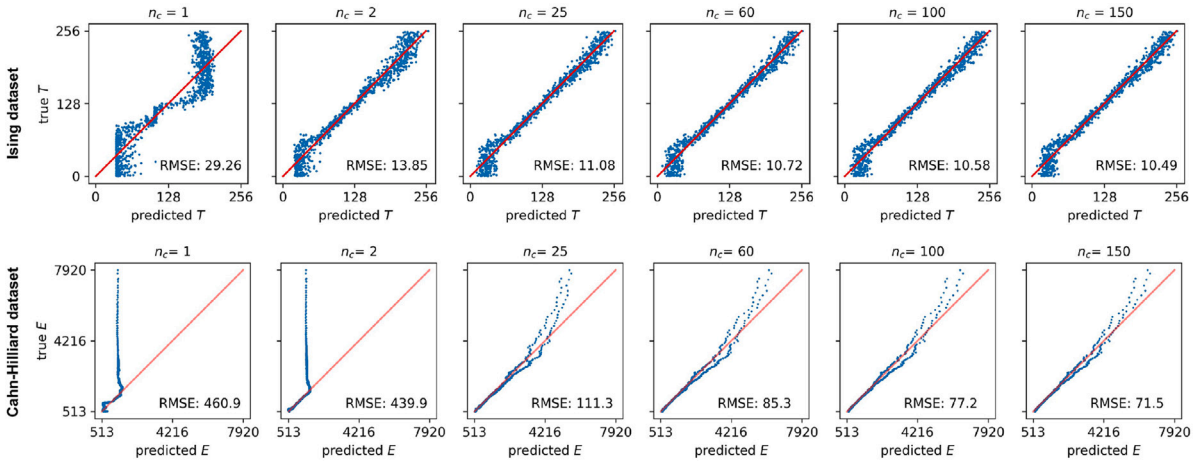

Fig. B.14. Confusion matrices of both datasets with changing number of principal components that are used as features for the SVR model.

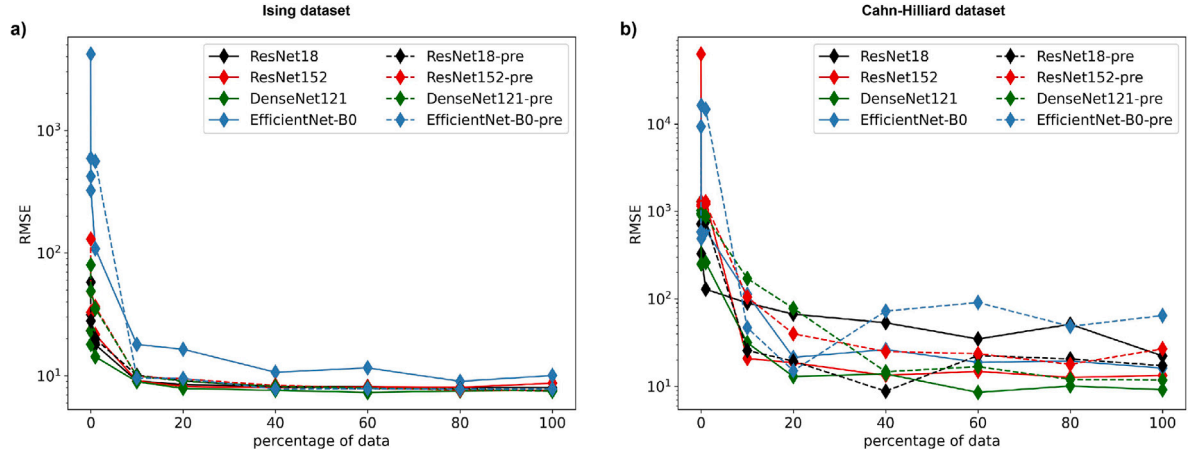

Fig. B.15. Prediction performance for different CNNs with random initialization of model weights (solid lines) and with pretrained weights (dashed lines) for different sizes of training sets for (a) the Ising datasets and (b) the Cahn–Hilliard datasets.

predictions in this regime. However, there are component pairs that do not follow the clockwise curve, leading to poor predictions for certain images in the low energy zone. The mid and high energy zones are very condensed in the latent space such that minor changes may lead to strongly different predictions, making the model in this region more error-prone.

B.3. Additional information for the multichannel CNN

Fourier transformations are often used in image processing, decomposing an image into its sine and cosine components. After the transformation, the image is represented in the Fourier or frequency domain where only the magnitude of the Fourier transform is used. For a square image of size $N \times N$, the two-dimensional Discrete Fourier Transform is given by:

$$F(u,v) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f(a,b) \exp^{-i2\pi*(\frac{ui}{N}+\frac{vj}{N})}, \tag{B.1}$$

where $f(a,b)$ is the image in the spatial domain, and the exponential term is the basis function corresponding to each point $F(u,v)$ in the Fourier space. The Discrete Wavelet Transform is given as,

$$W(u,v) = \sum_{a=0}^{N-1} \sum_{b=0}^{N-1} f(a,b)\phi_{(u,v)}(a,b), \tag{B.2}$$

where $\phi_{(u,v)}(a,b)$ is the basic wavelet function. The Fourier Transform produces a complex number-valued output image which can be

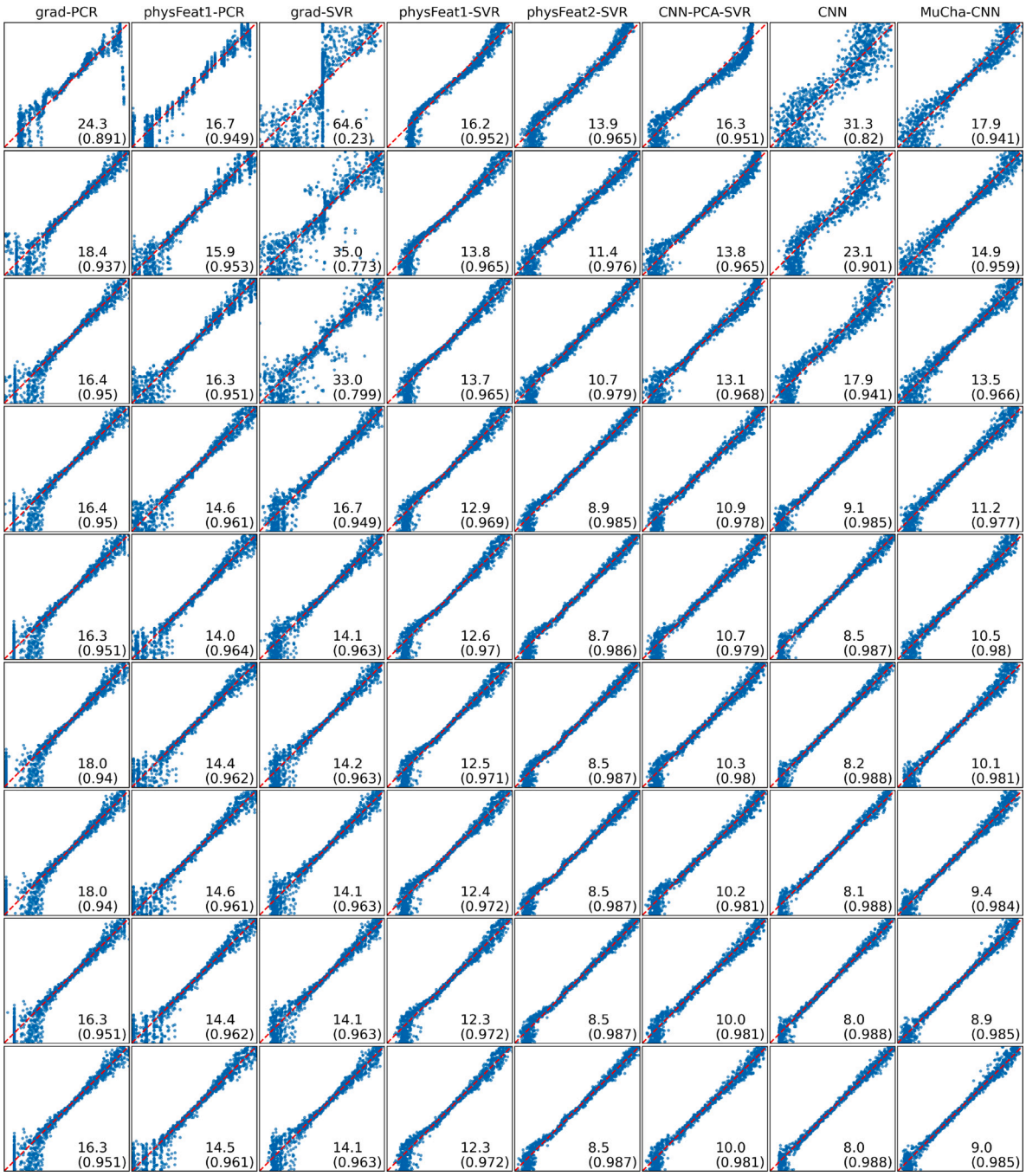

Fig. C.16. Confusion matrix of all models for training with different sizes of the Ising dataset. The vertical and horizontal data ranges (temperatures) for each sub-plot are 0..256. The number inside the sub-plot represents the RMSE value (without brackets) and R^2 score (with brackets).

displayed with two images, either with the real and imaginary part or with magnitude and phase. In image processing, often only the magnitude of the Fourier Transform is displayed, as it contains most of the information of the geometric structure of the spatial domain image. Basically, the frequency domain represents the rate of change in spatial pixels, which is advantageous when the investigated problem relates to the rate of change of pixels.

B.4. Additional information for the CNN-only approaches

Decreasing the dataset size to less than $\approx 10,000$ images results in a more pronounced loss of prediction accuracy for the CNN approaches as compared to the other models, cf. Fig. B.15. For the Ising dataset, the training with both weight initialization methods shows similar behavior (see Fig. B.15a). Difference occur only if less than 10% of the training data was used. Then the weight initialization from ImageNet results in higher RMSE values. For the Cahn–Hilliard dataset, a similar trend (with more scatter) can be observed (see Fig. B.15b). For both datasets, we find that training with random initialization of model weights gives a better performance. This is most likely due to the differences in the images used in the present study and the images of the ImageNet dataset used to obtain the pretrained weights. These images require different features than the ones learned by the pretrained weights, and this is why we did not find any benefit from the transfer learning approach. The small ResNet18 gives the best results for both datasets. This is not entirely surprising as similar results have also been obtained

Fig. C.17. Confusion matrix of all models for training with different sizes of the Cahn–Hilliard dataset. The vertical and horizontal data ranges (energies) for each sub-plot are 0..8000. The number inside the sub-plot represents the RMSE value (without brackets) and R^2 score (with brackets).

by other researchers where shallow models can provide better results compared to complex and deeper models (Bressem et al., 2020).

Appendix C. Visualization of confusion matrix from studying various percentages of dataset

Figs. C.16 and C.17 illustrate the confusion matrices for all models and for different sizes of the training datasets.

Appendix D. Comparing our approaches with various regression models from sklearn library package

The performance comparison of our approaches with various regression models from the sklearn library package (Nearest Neighbors, Linear SVM, RBF SVM, Decision Tree, Random Forest, Neural Net, AdaBoost, and Naive Bayes) are shown in Figs. D.18 and D.19. It is interesting to see that the performance of the Nearest Neighbors method is quite good for the Cahn–Hilliard dataset (with RMSE and R^2 score equal to 37 and 0.996), which is close to the performance of physFeat1-PCR.

Appendix E. Computational fluid dynamics (CFD) dataset: additional information and our approaches performance

The CFD dataset is obtained based on large eddy simulations of Kolmogorov flows, in which the image results in the strongly "curled" velocity field. The Reynolds numbers are chosen from the range of

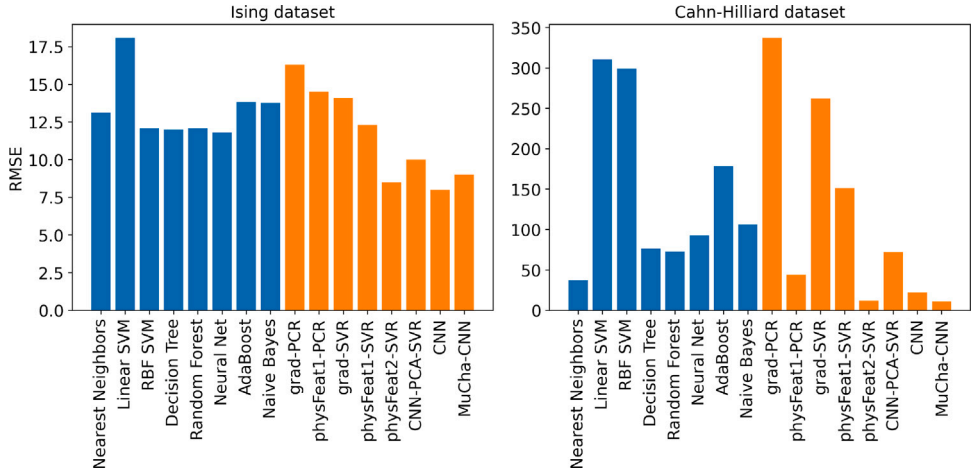

Fig. D.18. Visualization the RMSE of various approaches.

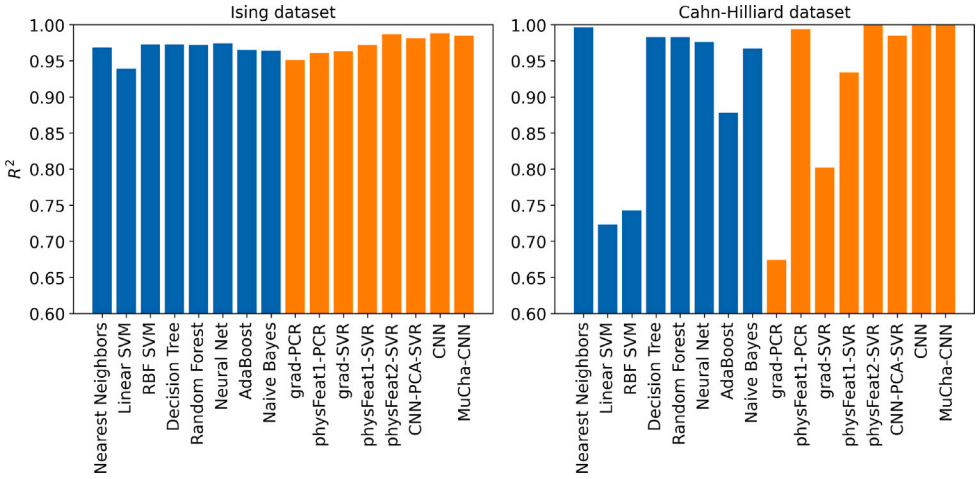

Fig. D.19. Visualization the R^2 score of various approaches.

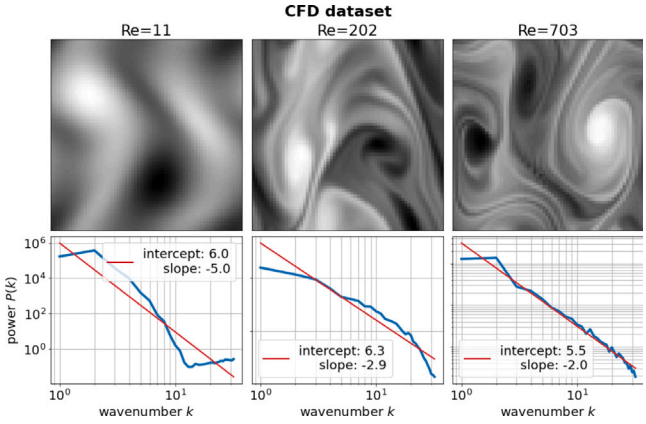

Fig. E.20. Microstructure and corresponding PSDs for the CFD dataset.

1..800. The velocity is obtained by solving the incompressible Navier–Stokes equations. More details about the simulation and the simulation software are described in Kochkov et al. (2021). Higher values of R result in smaller structures, requiring a higher spatial resolution. This

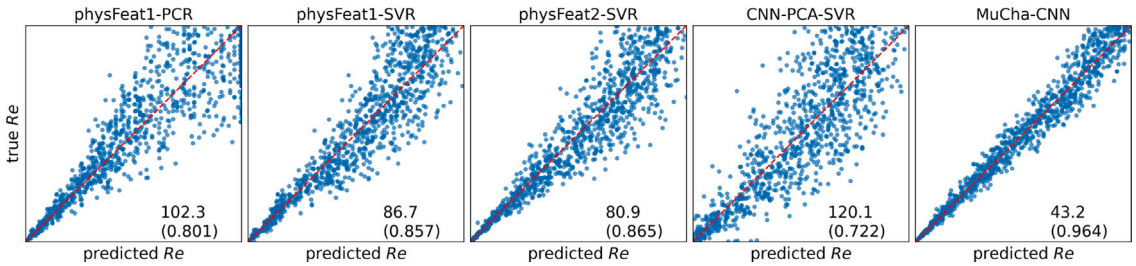

Fig. E.21. Confusion matrix of our approaches for the uniformed CFD dataset. The number inside the sub-plot represents the RMSE value (without brackets) and the R^2 score (with brackets).

partially explains why learning data for higher values of R is more difficult. The PSD features that are calculated from the images of this dataset are visualized in Fig. E.20.

The application of our approaches in this work for the CFD datasets is shown in Fig. E.21, respectively.

References

Baker, I. (2022). Interstitial strengthening in fcc metals and alloys. *Advanced Powder Materials*, *1*(4), Article 100034. http://dx.doi.org/10.1016/j.apmate.2022.100034.

Bergstra, J., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, & K. Weinberger (Eds.), *Vol. 24, Advances in neural information processing systems*. Curran Associates, Inc., URL https://proceedings.neurips.cc/paper_files/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf.

Bressem, K. K., Adams, L. C., Erxleben, C., Hamm, B., Niehues, S. M., & Vahldiek, J. L. (2020). Comparing different deep learning architectures for classification of chest radiographs. *Scientific Reports*, *10*(1), 13590.

Cahn, J. W. (1961). On spinodal decomposition. *Acta Metallurgica*, *9*(9), 795–801. http://dx.doi.org/10.1016/0001-6160(61)90182-1.

Callister, W. D., Jr. (2007). *Materials science and engineering an introduction*.

Cohen, T., & Welling, M. (2016). Group equivariant convolutional networks. In *International conference on machine learning* (pp. 2990–2999). PMLR.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248–255). IEEE.

Dreizler, R. M., & Gross, E. K. (2012). *Density functional theory: an approach to the quantum many-body problem*. Springer Science & Business Media.

Eghbalian, M., Pouragha, M., & Wan, R. (2023). A physics-informed deep neural network for surrogate modeling in classical elasto-plasticity. *Computers and Geotechnics*, *159*, Article 105472. http://dx.doi.org/10.1016/j.compgeo.2023.105472.

Espeholt, L., Agrawal, S., Sønderby, C., Kumar, M., Heek, J., Bromberg, C., et al. (2022). Deep learning for twelve hour precipitation forecasts. *Nature Communications*, *13*(1), 5145. http://dx.doi.org/10.1038/s41467-022-32483-x.

Ford, E., Maneparambil, K., Rajan, S., & Neithalath, N. (2021). Machine learning-based accelerated property prediction of two-phase materials using microstructural descriptors and finite element analysis. *Computational Materials Science*, *191*, Article 110328. http://dx.doi.org/10.1016/j.commatsci.2021.110328.

Gorania, M., Seker, H., & Haris, P. I. (2010). Predicting a protein's melting temperature from its amino acid sequence. In *2010 annual international conference of the IEEE engineering in medicine and biology* (pp. 1820–1823). IEEE.

Gupta, R., & Zhang, Q. (2024). Data-driven decision-focused surrogate modeling. *AIChE Journal*, e18338. http://dx.doi.org/10.1002/aic.18338.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., et al. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. http://dx.doi.org/10.1038/s41586-020-2649-2.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Hospital, A., Goñi, J. R., Orozco, M., & Gelpí, J. L. (2015). Molecular dynamics simulations: advances and applications. *Advances and Applications in Bioinformatics and Chemistry*, 37–47. http://dx.doi.org/10.2147/AABC.S70333.

Huebner, K. H., Dewhirst, D. L., Smith, D. E., & Byrom, T. G. (2001). *The finite element method for engineers*. John Wiley & Sons.

Ising, E. (1925). Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik*, *31*(1), 253–258. http://dx.doi.org/10.1007/BF02980577.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., et al. (2021). Highly accurate protein structure prediction with AlphaFold. *Nature*, *596*(7873), 583–589.

Jung, J., Yoon, J. I., Park, H. K., Kim, J. Y., & Kim, H. S. (2019). An efficient machine learning approach to establish structure-property linkages. *Computational Materials Science*, *156*, 17–25.

Kandel, J., Tayara, H., & Chong, K. T. (2021). PuResNet: prediction of protein-ligand binding sites using deep residual neural network. *Journal of Cheminformatics*, *13*(1), 1–14.

Khorrami, M. S., Mianroodi, J. R., Siboni, N. H., Goyal, P., Svendsen, B., Benner, P., et al. (2023). An artificial neural network for surrogate modeling of stress fields in viscoplastic polycrystalline materials. *Npj Computational Materials*, *9*(1), 37. http://dx.doi.org/10.1038/s41524-023-00991-z.

Kim, J., & Lee, H. G. (2021). Unconditionally energy stable second-order numerical scheme for the Allen–Cahn equation with a high-order polynomial free energy. *Advances in Difference Equations*, *2021*, 1–13. http://dx.doi.org/10.1186/s13662-021-03571-x.

Kochkov, D., Smith, J. A., Alieva, A., Wang, Q., Brenner, M. P., & Hoyer, S. (2021). Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, *118*(21), Article e2101784118. http://dx.doi.org/10.1073/pnas.2101784118.

Kohn, K. P., Underwood, S. M., & Cooper, M. M. (2018). Connecting structure–property and structure–function relationships across the disciplines of chemistry and biology: Exploring student perceptions. In J. Loertscher (Ed.), *CBE—Life Sciences Education*, *17*(2), ar33. http://dx.doi.org/10.1187/cbe.18-01-0004.

Langtangen, H. P., & Logg, A. (2017). *Solving PDEs in python*. Springer, http://dx.doi.org/10.1007/978-3-319-52462-7.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.

Lee, G., Gommers, R., Waselewski, F., Wohlfahrt, K., & O'Leary, A. (2019). Pywavelets: A python package for wavelet analysis. *Journal of Open Source Software*, *4*(36), 1237. http://dx.doi.org/10.21105/joss.01237.

Lenz, W. (1920). Beitrag zum verständnis der magnetischen erscheinungen in festen körpern. *Zeitschrift für Physik*, *21*, 613–615.

Li, Y., Tang, W., & Guo, M. (2021). The cell as matter: Connecting molecular biology to cellular functions. *Matter*, *4*(6), 1863–1891. http://dx.doi.org/10.1016/j.matt.2021.03.013.

Lißner, J., & Fritzen, F. (2019). Data-driven microstructure property relations. *Mathematical and Computational Applications*, *24*(2), 57. http://dx.doi.org/10.3390/mca24020057.

Messner, M. C. (2020). Convolutional neural network surrogate models for the mechanical properties of periodic structures. *Journal of Mechanical Design*, *142*(2), Article 024503. http://dx.doi.org/10.1115/1.4045040.

Nakka, R., Harursampath, D., & Ponnusami, S. A. (2023). A generalised deep learning-based surrogate model for homogenisation utilising material property encoding and physics-based bounds. *Scientific Reports*, *13*(1), 9079. http://dx.doi.org/10.1038/s41598-023-34823-3.

Nguyen, P. C., Choi, J. B., Udaykumar, H., & Baek, S. (2023). Challenges and opportunities for machine learning in multiscale computational modeling. *Journal of Computing and Information Science in Engineering*, *23*(6).

Nguyen, L. H., & Holmes, S. (2019). Ten quick tips for effective dimensionality reduction. *PLoS Computational Biology*, *15*(6), Article e1006907. http://dx.doi.org/10.1371/journal.pcbi.1006907.

Opiela, M., Fojt-Dymara, G., Grajcar, A., & Borek, W. (2020). Effect of grain size on the microstructure and strain hardening behavior of solution heat-treated low-C high-Mn steel. *Materials*, *13*(7), 1489.

Parida, S. S., Bose, S., & Apostolakis, G. (2023). Earthquake data augmentation using wavelet transform for training deep learning based surrogate models of nonlinear structures. *Vol. 55*, In *Structures* (pp. 638–649). Elsevier, http://dx.doi.org/10.1016/j.istruc.2023.05.122.

Parida, S. S., Bose, S., Butcher, M., Apostolakis, G., & Shekhar, P. (2023). SVD enabled data augmentation for machine learning based surrogate modeling of non-linear structures. *Engineering Structures*, *280*, Article 115600. http://dx.doi.org/10.1016/j.engstruct.2023.115600.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, R. Garnett (Eds.), *Advances in neural information processing systems 32* (pp. 8024–8035). Curran Associates, Inc., URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, *12*(Oct), 2825–2830.

Prifling, B., Röding, M., Townsend, P., Neumann, M., & Schmidt, V. (2021). Large-scale statistical learning for mass transport prediction in porous materials using 90,000 artificially generated microstructures. *Frontiers in Materials*, *8*, http://dx.doi.org/10.3389/fmats.2021.786502.

Raissi, M. (2018). Deep hidden physics models: Deep learning of nonlinear partial differential equations. *Journal of Machine Learning Research*, *19*(1), 932–955. http://dx.doi.org/10.48550/arXiv.1801.06637.

Rosenfeld, D., Woodley, W. L., Lerner, A., Kelman, G., & Lindsey, D. T. (2008). Satellite detection of severe convective storms by their retrieved vertical profiles of cloud particle effective radius and thermodynamic phase. *Journal of Geophysical Research: Atmospheres*, *113*(D4), http://dx.doi.org/10.1029/2007JD008600.

Sandfeld, S., & Zaiser, M. (2014). Deformation patterns and surface morphology in a minimal model of amorphous plasticity. *Journal of Statistical Mechanics: Theory and Experiment*, *2014*(3), P03014. http://dx.doi.org/10.1088/1742-5468/2014/03/P03014.

Sapoval, N., Aghazadeh, A., Nute, M. G., Antunes, D. A., Balaji, A., Baraniuk, R., et al. (2022). Current progress and open challenges for applying deep learning across the biosciences. *Nature Communications*, *13*(1), 1728. http://dx.doi.org/10.1038/s41467-022-29268-7.

Seif, M. N. (2023). Application of multi-scale computational techniques to complex materials systems.

Sharma, S., Joshi, S. S., Pantawane, M. V., Radhakrishnan, M., Mazumder, S., & Dahotre, N. B. (2023). Multiphysics multi-scale computational framework for linking process–structure–property relationships in metal additive manufacturing: a critical review. *International Materials Reviews*, 1–67.

Shen, T.-W., Li, C.-C., Lin, W.-F., Tseng, Y.-H., Wu, W.-F., Wu, S., et al. (2022). Improving image quality assessment based on the combination of the power spectrum of fingerprint images and prewitt filter. *Applied Sciences*, *12*(7), 3320. http://dx.doi.org/10.3390/app12073320.

Tan, M., & Le, Q. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning* (pp. 6105–6114). PMLR.

Tarasov, B. (2019). Dramatic weakening and embrittlement of intact hard rocks in the earth's crust at seismic depths as a cause of shallow earthquakes. In *Earth crust*. IntechOpen.

Tunyasuvunakool, K., Adler, J., Wu, Z., Green, T., Zielinski, M., Žídek, A., et al. (2021). Highly accurate protein structure prediction for the human proteome. *Nature*, *596*(7873), 590–596.

Vapnik, V. N. (2000). *The nature of statistical learning theory*. Springer New York, http://dx.doi.org/10.1007/978-1-4757-3264-1.

Wei, J., Chu, X., Sun, X.-Y., Xu, K., Deng, H.-X., Chen, J., et al. (2019). Machine learning in materials science. *InfoMat*, *1*(3), 338–358.

Xiong, P., Tong, L., Zhang, K., Shen, X., Battiston, R., Ouzounov, D., et al. (2021). Towards advancing the earthquake forecasting by machine learning of satellite data. *Science of the Total Environment*, *771*, Article 145256. http://dx.doi.org/10.1016/j.scitotenv.2021.145256.

Yu, T., Mo, X., Chen, M., & Yao, C. (2021). Machine-learning-assisted microstructure–property linkages of carbon nanotube-reinforced aluminum matrix nanocomposites produced by laser powder bed fusion. *Nanotechnology Reviews*, *10*(1), 1410–1424. http://dx.doi.org/10.1515/ntrev-2021-0093.

Zhang, R., Liu, Y., & Sun, H. (2020a). Physics-guided convolutional neural network (phycnn) for data-driven seismic response modeling. *Engineering Structures*, *215*, Article 110704. http://dx.doi.org/10.1016/j.engstruct.2020.110704.

Zhang, R., Liu, Y., & Sun, H. (2020b). Physics-informed multi-LSTM networks for metamodeling of nonlinear structures. *Computer Methods in Applied Mechanics and Engineering*, *369*, Article 113226. http://dx.doi.org/10.1016/j.cma.2020.113226.