Novel Data-driven Emulator for Predicting Microstructure Evolutions

by

Peichen Wu

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved January 2024 by the
Graduate Supervisory Committee:

Kumar Ankit, Chair
Yang Jiao
Houlong Zhuang
Ashif Iquebal

ARIZONA STATE UNIVERSITY

May 2024

ABSTRACT

Phase-field (PF) models are one of the most powerful tools to simulate microstructural evolution in metallic materials, polymers, and ceramics. However, existing PF approaches rely on rigorous mathematical model development, sophisticated numerical schemes, and high-performance computing for accuracy. Although recently developed surrogate microstructure models employ deep-learning techniques and reconstruction of microstructures from lower-dimensional data, their accuracy is fairly limited as spatio-temporal information is lost in the pursuit of dimensional reduction. Given these limitations, a novel data-driven emulator (DDE) for extrapolation prediction of microstructural evolution is presented, which combines an image-based convolutional and recurrent neural network (CRNN) with tensor decomposition, while leveraging previously obtained PF datasets for training. To assess the robustness of DDE, the emulation sequence and the scaling behavior with phase-field simulations for several noisy initial states are compared. In conclusion, the effectiveness of the microstructure emulation technique is explored in the context of accelerating runtime, along with an emphasis on its trade-off with accuracy.

Meanwhile, an interpolation DDE has also been tested, which is based on obtaining a low-dimensional representation of the microstructures via tensor decomposition and subsequently predicting the microstructure evolution in the low-dimensional space using Gaussian process regression (GPR). Once the microstructure predictions are obtained in the low-dimensional space, a hybrid input-output phase retrieval algorithm will be employed to reconstruct the microstructures. As proof of concept, the results on microstructure prediction for spinodal decomposition are presented, although the method itself is agnostic of the material parameters. Results show that GPR-based DDE model are able to predict microstructure evolution sequences that closely resemble the true microstructures (average normalized mean square of 6.78×10^{-7})

at time scales half of that employed in obtaining training data. This data-driven microstructure emulator opens new avenues to predict the microstructural evolution by leveraging phase-field simulations and physical experimentation where the time resolution is often quite large due to limited resources and physical constraints, such as the phase coarsening experiments previously performed in microgravity.

Future work will also be discussed and demonstrate the intended utilization of these two approaches for 3D microstructure prediction through their combined application.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Kumar Ankit, for his support and guidance over the course of my time at Arizona State University. I would also like to thank my committee members, Dr. Yang Jiao, Dr. Houlong Zhuang and Dr. Ashif Iquebal, for their help, time and consideration. I gratefully acknowledge Research Computing at Arizona State University for providing HPC and storage resources that have contributed to the computational research efforts. I would like to thank all the members of the 4D ICE Research Group for their help and encouragement. Financial support from the National Science Foundation (NSF) under Grant No. NSF CMMI-1763128 (Drs. Alexis Lewis and Thomas Kuech, Program Managers) and CAREER-2145812 (Dr. Jonathan Madison, Program Manager) are gratefully acknowledged. Finally, I would like to extend my gratitude to my family for their unwavering support and my girlfriend for her unending patience.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

PREFACE

The numerical modeling and analysis of microstructural evolution represent a subject of widespread scientific fascination. The phase-field method, recognized for its versatility, is a popular approach within the materials science community for modeling phase transformations. However, existing phase field approaches rely on rigorous mathematical model development, sophisticated numerical schemes, and high-performance computing for accuracy. In inspiration of recently developed surrogate data-driven models employ deep-learning techniques with traditional molecular dynamics methods, two novel data-driven emulator combined with phase field approach have been proposed.

I primarily developed the first data-driven emulator, utilizing convolutional and recurrent neural networks. Dr. Ashif took the lead in developing the second approach, as detailed in Article 2 listed below. Article 2 primarily utilizes a data-driven emulator that integrates tensor decomposition, Gaussian process regression, and the Phase Reconstruction algorithm. It then provides corresponding interpolation predictions for spinodal decomposition. In this article, my primary responsibility involves furnishing appropriate images illustrating the evolution of spinodal decomposition, crucial for constructing the training, validation, and test databases. Additionally, I contribute the corresponding code to quantify the microstructure's average grain size, serving as a metric to evaluate the performance of the data-driven emulator.

Data-driven emulator methods and prediction results reported in Chapter 2 (Convolutional and recurrent neural network, Gaussian process regression) and Chapter 3 (Microstructure image predictions via Data-driven emulator) were published in the following peer-reviewed articles:

***Article 1**: Wu, P*., Iquebal, A., & Ankit, K. (2023). Emulating microstructural evolution during spinodal decomposition using a tensor decomposed convolutional and recurrent neural network. Computational Materials Science 224: 112187*

***Article 2**: Iquebal, A., Wu, P*., & Ankit, K. (2023). Emulating the evolution of phase separating microstructures using low-dimensional tensor decomposition and nonlinear regression. MRS Bulletin 48: 602-613*

***Article 3**: Wu, P*., Farmer, W., & Ankit, K. (2023). A Novel Data-Driven Emulator for Predicting Electromigration-Mediated Damage in Poly-crystalline Interconnects. Journal of Electronic Materials 52: 2746-2761*

The dissertation work also resulted in additional research that was published in the following articles:

***Article 4**: Raghavan, R., Wu, P*., & Ankit, K. (2022). Phase-field modeling of nanostructural evolution in physical vapor deposited phase-separating ternary alloy films. Modelling and Simulation in Materials Science and Engineering 30(8), 084004*

***Article 5**: Glicksman, M. E., Ankit, K., & Wu, P*. (2022). Capillary effects on curved solid–liquid interfaces: An overview. Journal of Crystal Growth 126871*

***Article 6**: Glicksman, M. E., Wu, P*., & Ankit, K. (2022). Periodic Grain Boundary Grooves: Analytic Model, Formation Energies, and Phase-Field Comparison. Journal of Phase Equilibria and Diffusion 1-20*

***Article 7**: Glicksman, M. E., Wu, P*., & Ankit, K. (2021). Surface Laplacian of interfacial thermochemical potential: its role in solid-liquid pattern formation. Nature Microgravity 7(1), 41*

Chapter 1

INTRODUCTION

The materials science domain has recently experienced a swift surge in the adoption of data-driven practices, resulting in the creation of efficient, widely applicable, and precise methods for various applications. These applications encompass material property prediction, the exploration of microstructure-property and microstructure-processing relationships, and the characterization of material microstructures. At the heart of materials science is the crucial task of connecting microstructure to properties and performance. An essential element in establishing these connections is gaining a profound understanding of how microstructures undergo changes in response to environmental exposure or processing conditions, such as time, temperature, applied stress or strain, and irradiation.

Enhancements in computational capabilities have been driven by the integration of deep neural networks, improved hardware, and the availability of openly accessible software packages. Computational materials science spans a broad spectrum, employing numerous methods that cover length scales from the atomic to the continuum. Techniques like phase field modeling find extensive application for predicting the evolution of microstructures in both two- and three-dimensional systems. Furthermore, density functional theory (DFT), a quantum-based approach, has played a pivotal role in uncovering new materials, pinpointing dopants for alloy reinforcement, elucidating diffusion mechanisms, and more.

Nevertheless, any computational methodology encounters limitations related to the length and timescales of simulations, accuracy, and the generalizability or transferability across diverse material systems. Advancements in materials science hinge

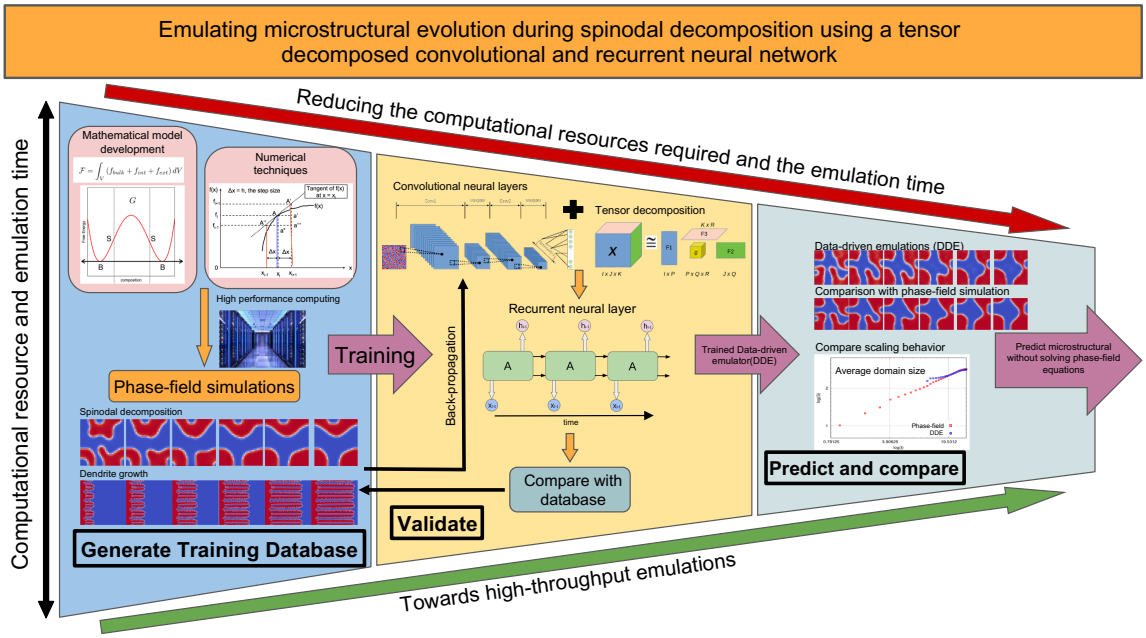

Figure 1.1: Data-driven emulator working flow, as demonstrated by Peichen et al. (2023)

significantly on feedback from both computational models and validation through experiments. However, the expansion of such methods to handle larger datasets, particularly in three dimensions, poses challenges due to the considerable time and computational resources involved.

Phase-field methods are one of the most powerful and versatile tools to simulate microstructural evolution in metallic materials, polymers, and ceramic. Relevant studies showcasing this capability include works by Boettinger et al. (2002); Chen (2002); Moelans et al. (2008); Singer-Loginova and Singer (2008). As an illustration, the phase-field method has proven highly effective in simulating phenomena such as dendrite growth, grain coarsening, and eutectic growth, as depicted in fig. 1.2, fig. 1.3, and fig. 1.4, respectively. The popularity of the phase-field method's in modeling morphological evolution is due to the elegance with which it treats moving boundary

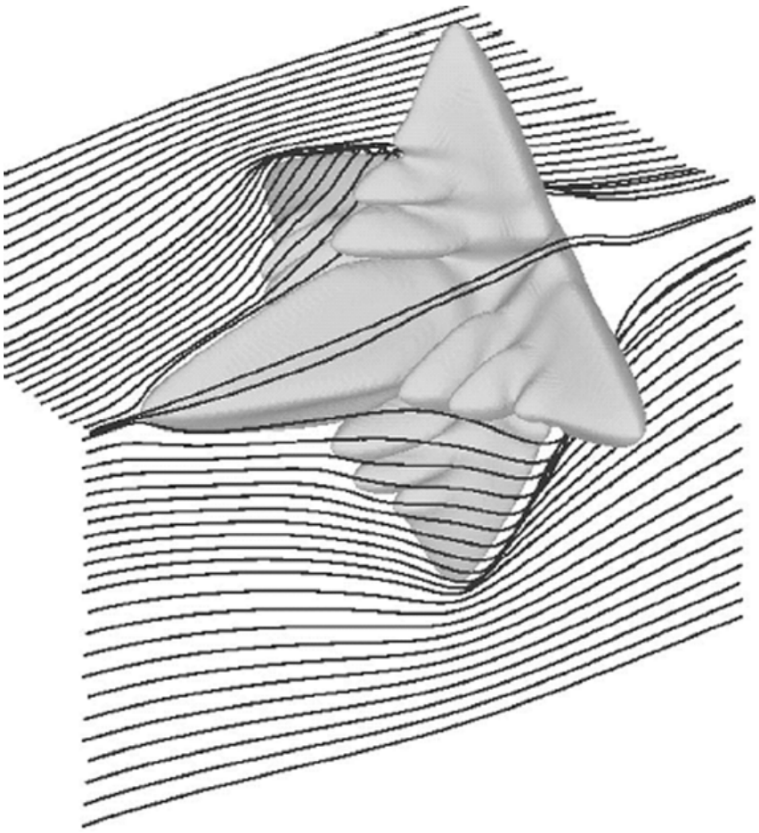

Figure 1.2: A freely growing dendrite in a forced flow. The lines represent computed stream traces directed from left to right. Reprinted with permission from Singer-Loginova and Singer (2008). Copyright 2008 by the IOP Publishing, Ltd.

problems by obviating the necessity to explicitly track the interfaces. However, they remain computationally intensive due to the strict limits on the maximum time and length scales imposed by the numerical methods, as highlighted Kautz (2021). For instance, the computational complexity associated with simulating spinodal decomposition, as elucidated by Provatas and Elder (2010); Moelans et al. (2008), warrants the application of supercomputers for solving fourth-order Cahn-Hilliard partial differential equations (PDEs). The finite difference and the finite element schemes have been extensively employed, although there are challenges related to numerical imple-

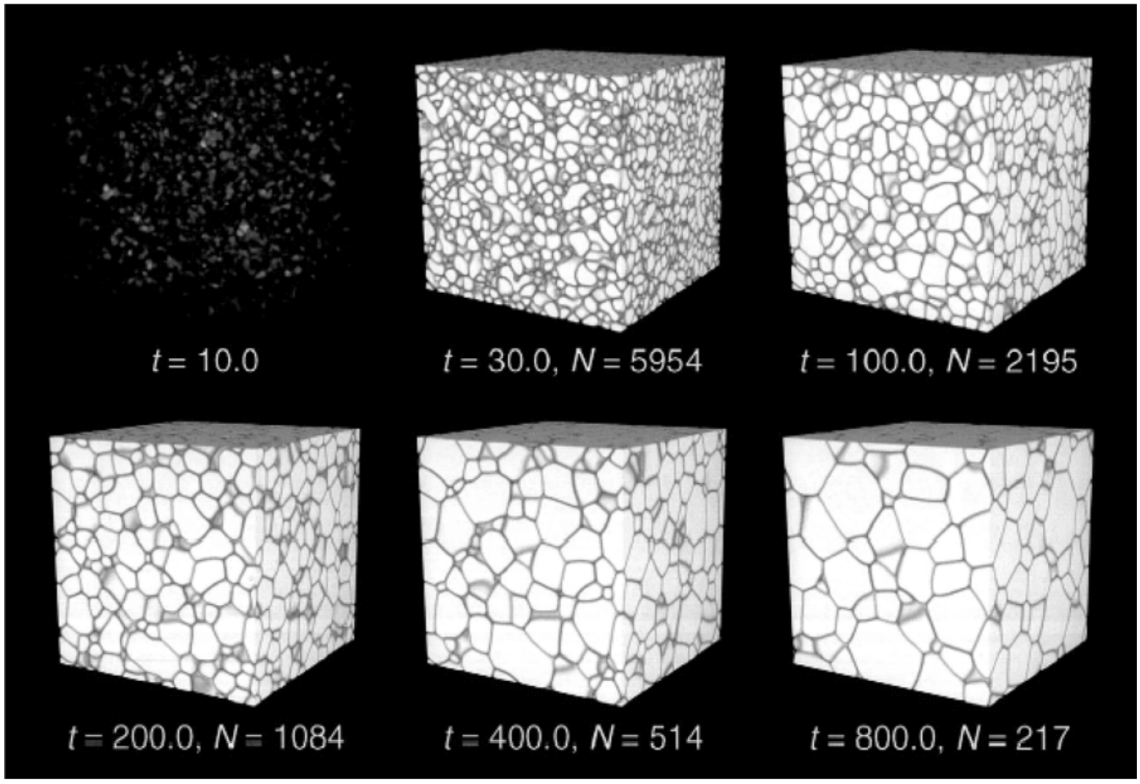

Figure 1.3: Grain coarsening process obtained from 3D simulations of grain growth. In the model, the grain boundary energy and mobility is isotropic. The elapsed time t and the number of grains N are specified for every image. The microstructure at t = 10.0 illustrates the homogeneous nucleation of crystallites from the undercooled liquid initial state. Reprinted with permission from Singer-Loginova and Singer (2008). Copyright 2008 by the IOP Publishing, Ltd.

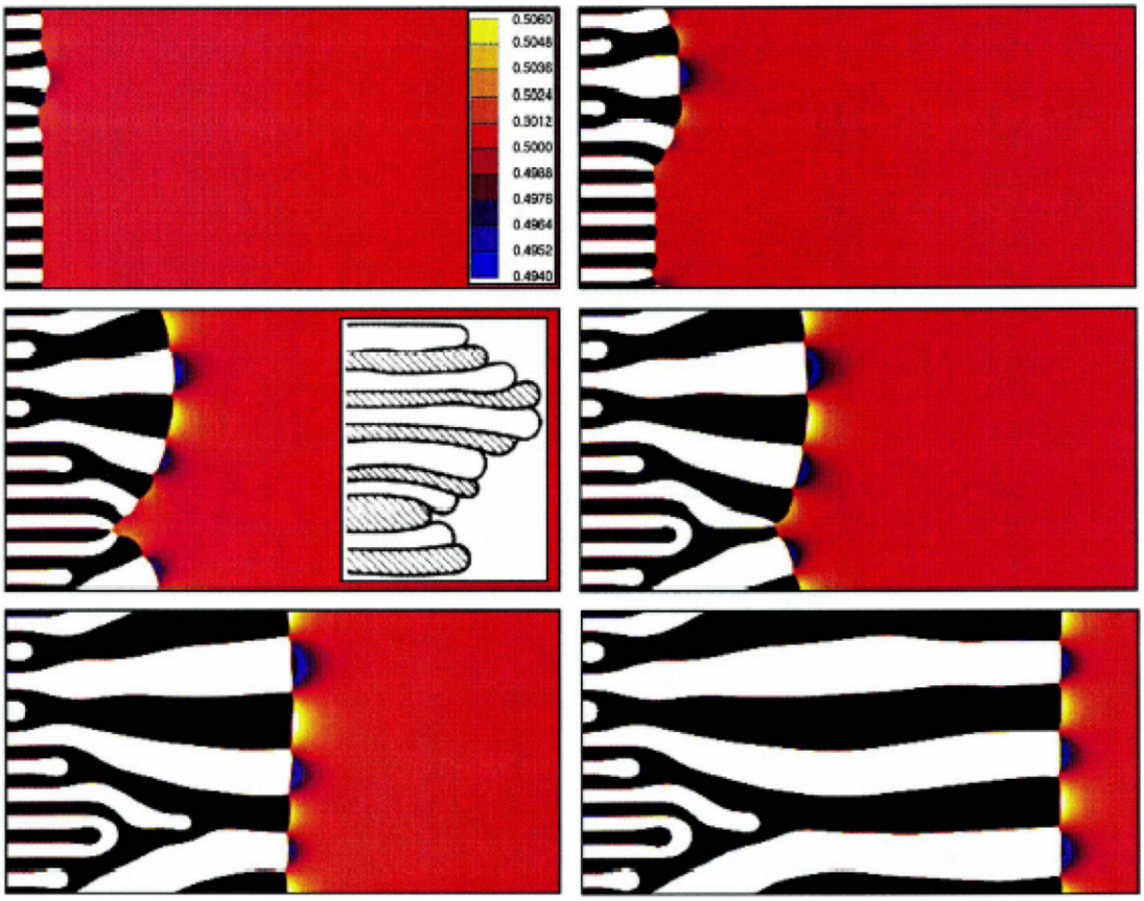

Figure 1.4: Evolution of 2D eutectic lamellae of a binary alloy and the selection process of lamellar spacing. The black and white colors represent the two solid phases, and the numerical values of the solute concentration in the liquid are indicated in the map in the first picture. Reprinted with permission from Singer-Loginova and Singer (2008). Copyright 2008 by the IOP Publishing, Ltd.

mentation, as outlined by Cheng et al. (2019). Various techniques have also been employed to shorten the simulation runtimes, such as adaptive mesh refinement as demonstrated by Provatas et al. (1998); Greenwood et al. (2018), load balancing as discussed by George and Warren (2002), and random walker presented by Plapp and Karma (2000).

However, depending on the spatial and time resolution warranted, simulating the evolution of microstructures in three dimensions, which is a 4D problem, often spans from several hours to weeks on a supercomputing cluster. Contemporary approaches for expediting phase-field calculations depend on high-performance GPUs and power-intensive computational resources, as highlighted by Shimokawabe et al. (2011), rather than utilizing past simulations to emulate new ones. As a result, a new simulation needs to be performed every time a simulation parameter is altered. Another limitation of the phase-field method is that it is not transferable across material systems without extensive parameter adaptation, as noted by DeWitt et al. (2020), even though the underlying phase transformation mechanism may be similar such as spinodally decomposing microstructures of polymers, as discussed by Glotzer (1995); Bruder and Brenn (1992); Mukherjee et al. (2016) and metallic materials, as demonstrated by Rundman and Hilliard (1967); Langer (1971); Miller et al. (1995), both of which entail up-hill diffusion of atoms.

Merriman-Bence-Osher (MBO) threshold dynamics is another popular algorithm for simulating the mean curvature motion of interface, as proposed by Merriman et al. (1992, 1994). However, this technique is only first order accurate in a two-phase setting while the accuracy is known to rapidly degrade in a multiphase settings. A recent effort has rendered MBO dynamics to be second order accurate in problems limited to two-phases, as demonstrated by Zaitzeff et al. (2020), however, there are currently no accurate schemes available for 3D multiphase problems.

In fact, the dilemma of efficiency, computation cost, and accuracy is not a new topic of interest within the materials modeling community. Parallel computation, which utilizes multiple CPUs or GPUs, is one of the ways by which simulation runtimes can be reduced, as evidenced by its extensive use in molecular dynamics, phase-field, and finite element simulations as noted by Millán et al. (2017). However, the associated computation cost is still high and often requires powerful and expensive high-performance computing resources. Another pathway to dealing with this issue is the utilization of machine learning. Some of the recent examples include neural network-aided methodology to design modular meta-materials by Wu et al. (2020), potentials for molecular dynamics (DeePMD) as demonstrated by Wang et al. (2018), depicted in fig. 1.5, fig. 1.6 respectively, and Behler-Parrinello neural network (BPNN) by Behler and Parrinello (2007).

While the existing literature extensively covers techniques that concentrate on the fusion of machine learning with molecular dynamics and meta-materials, there is a scarcity of works specifically addressing the integration of machine learning with phase-field methods, as observed in a limited number of studies by Zapiain et al. (2021); Herman et al. (2020); Yang et al. (2021); Hu et al. (2022); Oommen et al. (2022); Desai and Dingreville (2022). These studies primarily rely on incorporating dimensional reduction, machine learning, and phase reconstruction, for microstructure emulations. For the dimensional reduction of microstructural images, Principal Component Analysis (PCA) is typically employed that has several limitations, as noted in Elhaik (2022). Since PCA requires unfolding an image data into a one-dimensional array which causes loss of spatio-temporal information, its use is to be best avoided when dealing with image sequences for e.g. an evolving microstructure. This spatio-temporal loss manifests as poor reconstruction of microstructures. This is evident from the mismatch between the emulated and simulated spinodal microstruc-

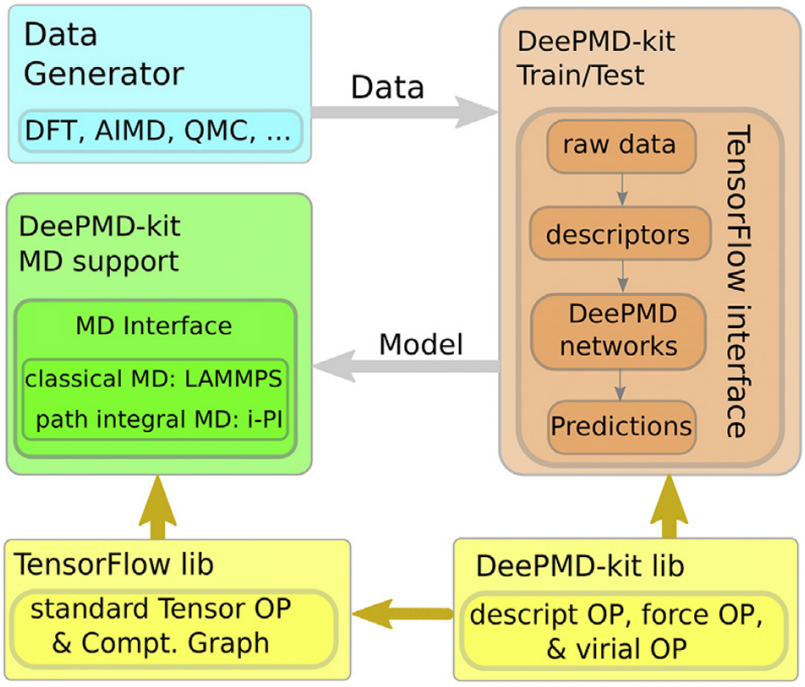

Figure 1.5: Schematic plot of the DeePMD-kit architecture and the workflow. The gray arrows present the workflow. The data, including energy, force, virial, box, and type, are passed from the Data Generator to the DeePMD-kit Train/Test module to perform training. After training, the DeePMD model is passed to the DeePMD-kit MD support module to perform MD. The TensorFlow and DeePMD-kit libraries are used for supporting different calculations. See text for detailed descriptions. Reprinted with permission from Wang et al. (2018). Copyright 2018 by the Elsevier.

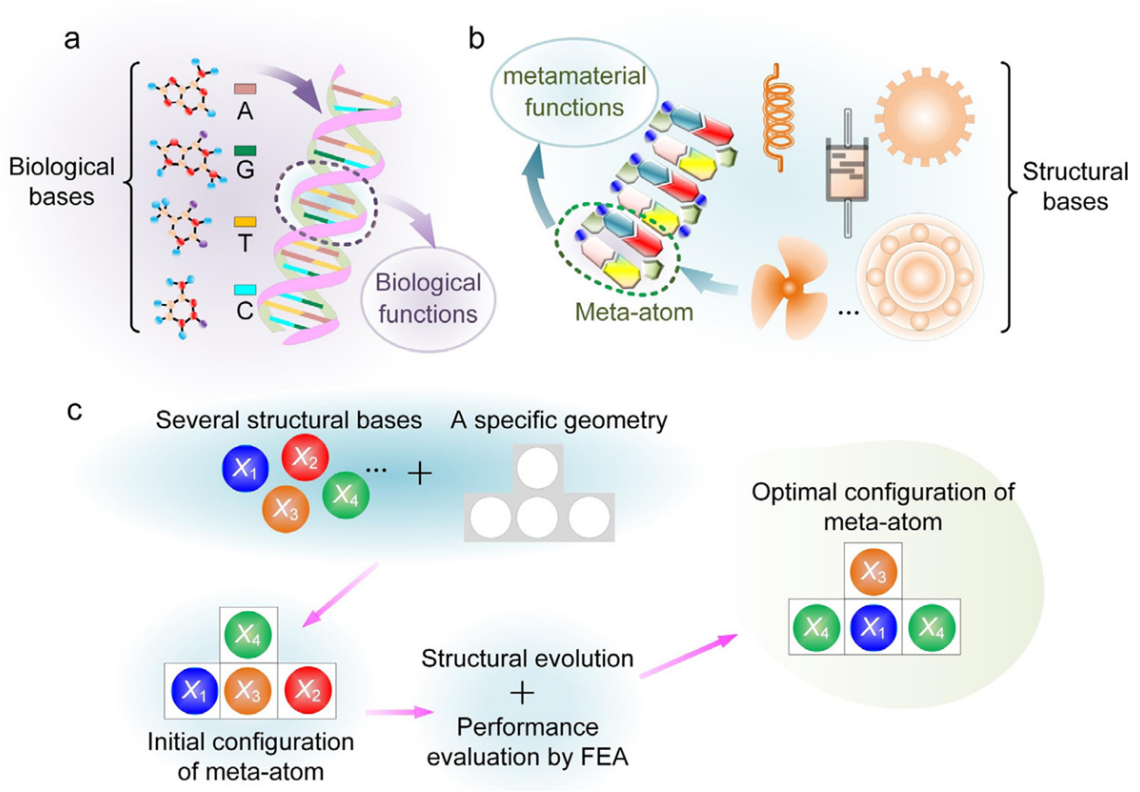

Figure 1.6: Schematic of (a)biological genome and (b)modular design of meta-atoms. (c)Schematic of machine learning-based method to design modular meta-materials. Reprinted with permission from Wu et al. (2020). Copyright 2020 by the Elsevier.

tures shown in fig. 1.7, which is report by Zapiain et al. (2021). Another limitation is that the phase reconstruction adopted by those authors is not very effective in reconstructing the diffuse interfaces which are a characteristic of phase-field simulations. Although, reconstruction of interface may be not be an issue for a majority of solid-state transformations where the thickness of the interfaces in quite small (below 1 nm), the above noted limitation is particularly concerning for the case of solidification microstructures where the solid-liquid interface is known to be comparatively thicker. As compared to PCA, tensor decomposition is a better alternative since it does not lead to any spatio-temporal losses, as demonstrated by Iquebal et al. (2023)

although accurate emulations of microstructures may incur large computational costs.

- Chapter 2 presented a detailed exploration of the phase field model utilized in generating the training, validation, and testing databases. Subsequently, two distinct data-driven approaches (DDEs) are introduced: the CRNN-based approach (Convolutional and Recurrent Neural Network) and the GPR-based approach (Gaussian Process Regression). The phase field model is employed to simulate spinodal decomposition, exhibiting diverse morphological evolution for various initial conditions. The derivation from the free energy functional to the governing concentration equation is outlined, along with the specification of initial parameters for generating the database.

  The CRNN-based approach is then introduced and expounded upon. It comprises three primary components: a convolutional neural layer, tensor decomposition, and a recurrent neural layer. The intricacies of each component are explained, and tensor dimensions for every layer are presented, utilizing an example input image size of $(32\Delta x \times 32\Delta y)$. Following this, the GPR-based approach is introduced, consisting of four main components: two-point correlation, tensor decomposition, Gaussian Process Regression (GPR), and phase reconstruction. Each component is detailed explained in this section, along with relevant appendixes.

- Chapter 3 provides comprehensive predictions and validations for the two distinct data-driven approaches (DDEs). Then potential future work has been discussed. Regarding the CRNN-based approach, predictions are presented for three microstructure evolution cases and are compared with corresponding phase field images. Through doubling the training database volume, signifi-

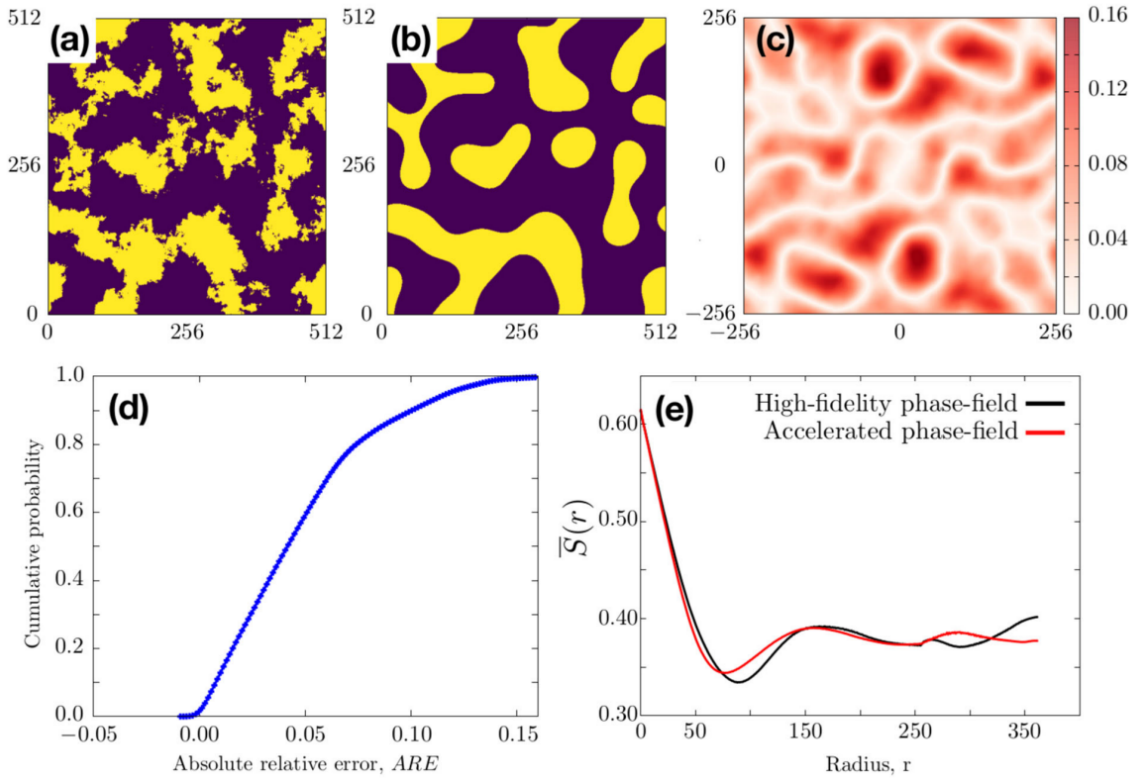

Figure 1.7: Accelerated phase-field predictions. (a) Reconstructed microstructure from the LSTM-trained surrogate model using a phase-recovery algorithm. (b) Phase-field predictions using LSTM-trained surrogate model as an input. (c) Point-wise error between predicted and true microstructure evolution. (d) Cumulative probability distribution of the absolute relative error on characteristic microstructural feature size. (e) Comparison of radial average of the microstructure auto-correlation between predicted (red) and true (black) microstructure evolution. Reprinted with permission from Zapiain et al. (2021). Copyright 2021 by the Nature.

cant improvements in emulating microstructural evolution are achieved across all cases. Average grain size plots from DDE predictions exhibit a consistent trend similar to that observed in phase field results. Comparative analyses are extended to DDE predictions for a larger domain size, showcasing reasonable agreement in temporal evolution trends of emulated free energy minimization with phase-field counterparts.

Minor discrepancies in emulating 1-D interface profiles of ϕ are noted with smaller training database volumes, but convergence improves when doubling the training volume. A table compares computation costs between phase-field calculations using finite difference and Fourier spectral solvers and DDE. Training efficiency gains through tensor decomposition (TD) per epoch are listed, and CPU runtime improvement is quantified, indicating that DDE is approximately 63 times faster than the finite difference solver. This section concludes by discussing the trade-off between accuracy and training efficiency.

In the context of the GPR-based approach (Gaussian Process Regression), comparisons are made between sequences of two-point correlation images and microstructure images from phase field and DDE. DDE predictions demonstrate a similar evolution trend to their corresponding phase field images. To highlight the advantages of GPR over other nonlinear regression methods, comparisons are made with support vector regression using radial basis and polynomial kernels. GPR accurately fits data points, while support vector regression struggles to capture nonlinear behavior. Additional comparisons involve average domain size, average normalized Mean Squared Error (MSE) for distinct Δt from both DDE and phase field. Finally, a table contrasts computational run times of the emulator and phase-field simulations across four different domain

sizes: $32\Delta x \times 32\Delta y$, $256\Delta x \times 256\Delta y$, $512\Delta x \times 512\Delta y$ and $1024\Delta x \times 1024\Delta y$.

- Chapter 4 concludes the dissertation document, with a summary of the many topics that are explored pertaining to phase separation in PVD alloy films, and the conclusions that are reached.

Chapter 2

PHASE FIELD AND DATA-DRIVEN EMULATORS

2.1 Phase Field Model

For the sake of completeness, the well-known Cahn-Hilliard model, which is used to generate microstructure training datasets, is briefly outlined in this section. Our diffuse interface approach for modeling the phase separation in binary immiscible alloy (A-B) films adopts a free energy functional consisting of distinct bulk and interfacial energy terms, as described by Cahn (1961), written as

$$F = \int_{\Omega} \left[f(\phi) + \frac{1}{2}\epsilon^2 |\nabla \phi|^2 \right] d\Omega. \tag{2.1}$$

Here, the order parameter, $\phi(x,t)$, denotes the scaled concentration assumed to be a continuous function of position and time, ranging from equilibrium concentration of 0 at the A-rich β phase to 1 at the B-rich α phase. The bulk free energy density, $f(\phi)$, is given by

$$f(\phi) = \frac{1}{4} W \phi^2 (1-\phi)^2 \tag{2.2}$$

where W is the well height that puts an energy penalty to all the states other than 0 and 1. ϵ is the gradient free energy coefficient, which penalizes large gradients in the order parameter, giving rise to the diffuse nature of phase boundaries. At equilibrium, the interfacial width, δ, and the interfacial energy, γ, is governed by an interplay of the two terms in the free energy functional and scales as $\delta \propto \sqrt{\epsilon^2/W}$ and $\gamma \propto \sqrt{\epsilon^2 W}$.

The kinetics of a phase-separating system can be simulated by solving the Cahn-Hilliard equation,

$$\frac{\partial \phi}{\partial t} = \nabla \cdot M \nabla \mu \tag{2.3}$$

where, μ is the chemical potential derived from the variational derivative of the free energy functional as $\mu = \frac{\delta F}{\delta \phi}$. M is the mobility of the diffusing species which can be related to the diffusion coefficient as $D = M\frac{\partial^2 f}{\partial \phi^2}$, assumed to be 1.0. In the present study, we assume the kinetic parameter to be independent of the order parameter, thus, the evolution equation may be rewritten as,

$$\frac{\partial \phi}{\partial t} = M\nabla^2 \frac{\delta F}{\delta \phi}. \tag{2.4}$$

Using Eqs. 2.1 and 2.2, we arrive at the final form of the Cahn-Hilliard equation,

$$\frac{\partial \phi}{\partial t} = M\nabla^2 \left[\frac{1}{2}W(2\phi^3 - 3\phi^2 + \phi) - \epsilon^2\nabla^2\phi\right]. \tag{2.5}$$

Eq. 2.5 is non-dimensionalized by using reduced variables which are defined as: $x^* = x/\Delta x$, $M^* = M/(M_0 k_B T)$, $\nabla^* = (\Delta x)^2\nabla$, $W^* = W/(k_B T)$, $\epsilon^* = \epsilon/(\Delta x\sqrt{k_B T})$, and $t^* = M_0 t/(\Delta x)^2$, where Δx is the grid spacing, M_0 is an arbitrarily defined constant bulk mobility that is dependent on the temperature, T, and k_B is the Boltzmann constant. The final dimensionless form is given by

$$\frac{\partial \phi(x^*,\, t^*)}{\partial t^*} = M^*(\nabla^*)^2\left\{\frac{1}{2}W^*(2\phi^3 - 3\phi^2 + \phi) - (\epsilon^*)^2(\nabla^*)^2\phi\right\}. \tag{2.6}$$

We solve Eq. 2.3 via an explicit finite difference method (FDM) on a regular square mesh. The method describe here is not the only way to solve those equations, one can also use Allen-Cahn equation with a Lagrange multiplier to achieve the same effect but with less computation time. One can also solve the We follow a forward difference Euler scheme for the temporal derivatives and a second-order central difference for the spatial derivatives. The Cahn-Hilliard Eq. 2.6 is solved in two steps. First, the chemical potential μ is calculated at each grid point, while the Laplacian is calculated using a 7-point stencil. The discretized equation required to compute μ is given by

$$\begin{aligned}\mu^t_{i,j,k} =& \frac{1}{2}W\phi^t_{i,j,k}(1-\phi^t_{i,j,k})(1-2\phi^t_{i,j,k}) \\ &- \epsilon^2\frac{\phi^t_{i+1,j,k} + \phi^t_{i-1,j,k} + \phi^t_{i,j+1,k} + \phi^t_{i,j-1,k} + \phi^t_{i,j,k+1} + \phi^t_{i,j,k-1} - 6\phi^t_{i,j,k}}{h^2}.\end{aligned} \tag{2.7}$$

The Laplacian of μ is determined by another central difference scheme such that the concentration evolution equation in the discretized takes the following form,

$$\frac{\phi_{i,j,k}^{t+\Delta t} - \phi_{i,j,k}^{t}}{\Delta t} = M\left\{\frac{\mu_{i+1,j,k}^{t} + \mu_{i-1,j,k}^{t} + \mu_{i,j+1,k}^{t} + \mu_{i,j-1,k}^{t} + \mu_{i,j,k+1}^{t} + \mu_{i,j,k-1}^{t} - 6\mu_{i,j,k}^{t}}{h^2}\right\}. \tag{2.8}$$

The square grid dimensions are given by $\Delta x = \Delta y = 0.3$, while the timestep is defined by $\Delta t = 0.0001$. Periodic boundary conditions are imposed along the x- and the y-directions. Non-dimensional $M^* = 1.0$ allows the phase separating microstructures to coarsen within a reasonable timeframe to limit the training time to below 1.5 hours, while $W* = 4.0$ and $\epsilon^* = 0.051$ ensure retention of equimolar composition as the diffuse phase interfaces of width, 6 grid points ($6\Delta x$), develops. The initial condition used for generating the phase-field training data comprises an equimolar alloy composition with fluctuations, the amplitude of which varies in the range of 0.01% - 0.50%. Since these composition fluctuations are seeded randomly, distinct microstructural evolution sequences are obtained from every simulation run.

2.2 Data-driven Emulator

In this section, two data-driven emulators will be presented. One is the convolutional and recurrent neural network (CRNN), which excels in extrapolation predictions, while the other approach employs Gaussian process regression, known for its proficiency in interpolation predictions. The first one is good at predicting into future, while the latter one is good at predicting middle timestep during a time-series.

2.2.1 Convolutional and Recurrent Neural Network

The data-driven emulator or the DDE proposed in this work is based on a CRNN framework that employs a tensor decomposed convolutional neural network (CNN) for image feature extraction, which is coupled to a recurrent neural network (RNN). The latter enables image sequence prediction based on the series of features extracted over time using CNN. Here, features refer to the line and curvature within the simulated micrographs generated from phase-field simulations. Detailed model structure is shown in Fig 2.1. TD entails decomposition of a 2D convolutional layer matrix into several smaller layers. Although the number of layers as a result of decomposition increases, the total number of floating-point operations and weights will be smaller than the parent layer. The input and output tensor dimensions of a tensor decomposed convolutional layer will be the same as a regular convolutional layer.

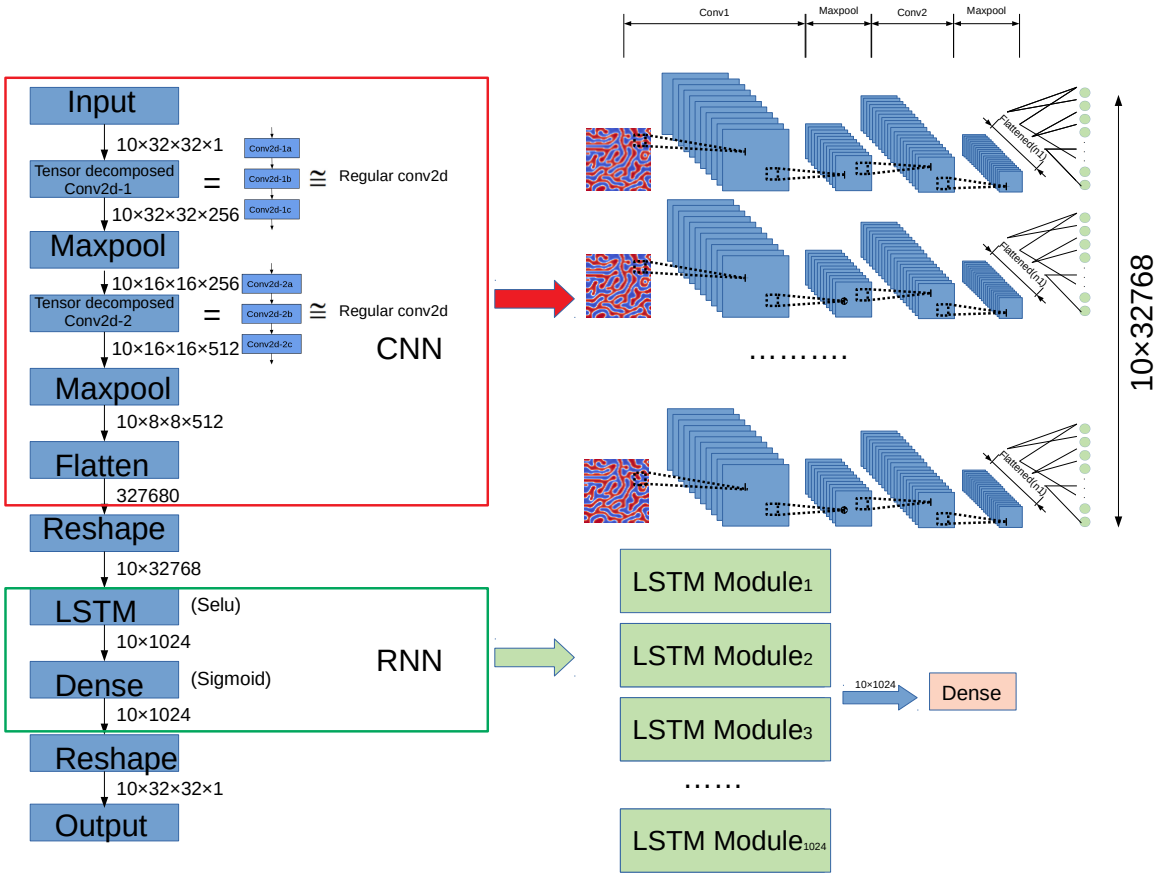

Figure 2.1: CRNN layer structure, as proposed by Peichen et al. (2023), and the corresponding tensor dimension. The activation function and padding type are also indicated. (32× 32 is taken as an example for input image size)

Tensor Decomposition

Tensor decompositions play a crucial role in various mathematical contexts, including the implementation of numerically efficient algorithms, solving systems of linear equations, and extracting essential information from matrices. Deep learning algorithms often needs to process huge amount of tensors, thus a efficient method is needed to speed up the training process. Our work has employed Tucker decomposition to decompose our regular convolutional neural layers into several smaller layers. Although the number of layers as a result of decomposition increases, the total num-

ber of floating-point operations and weights will be smaller than the parent layer. The process of tucker decomposition is shown in 2.2. This approach is originated from Kim et al. (2016)'s work, they successfully compressed the deep convolutional neural networks and deployed it in low computational power mobile devices. This method involves three steps: (1) selecting ranks through variational Bayesian matrix factorization, (2) applying Tucker decomposition to the kernel tensor, and (3) fine-tuning to regain any accumulated loss of accuracy. Each of these steps can be readily implemented using publicly accessible tools. The efficacy of this method has been evaluated on different compressed CNNs (AlexNet, VGGS, GoogLeNet, and VGG-16) using a smartphone. Substantial reductions in model size, runtime, and energy consumption have been achieved, albeit with a slight trade-off in accuracy. A detailed mathematical derivation for Tucker decomposition is shown in Appendix A.

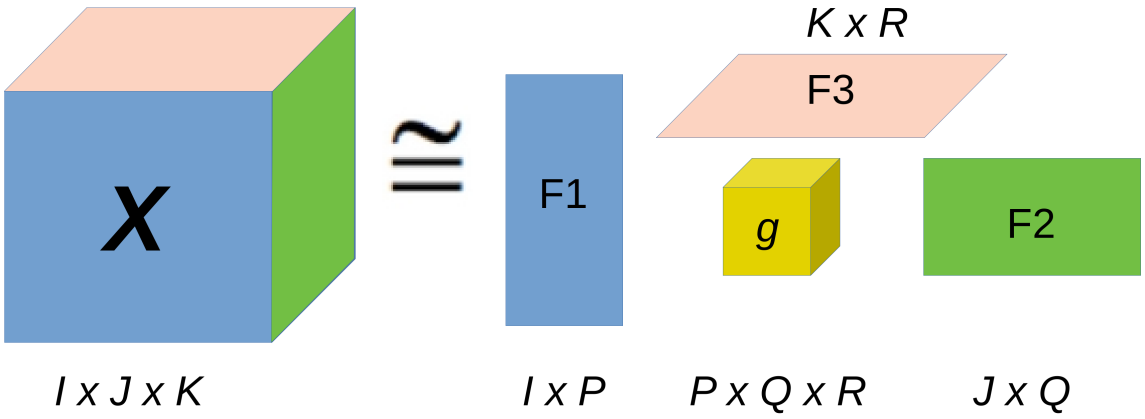

Figure 2.2: Illustration of a Tucker decomposition, as mentioned by Peichen et al. (2023), in which a tensor X can be decomposed as a core tensor g and factor matrices, $F1$, $F2$, and $F3$, i.e. one for every mode.

Convolutional Neural Layers

Artificial Intelligence has been undergoing significant expansion in narrowing the gap between human and machine capabilities. Researchers and enthusiasts are actively engaged in various aspects of the field, striving to achieve remarkable outcomes. Among the many areas of focus is Computer Vision. The objective within this field is to empower machines to perceive the world akin to humans, interpret it similarly, and apply this understanding to a myriad of tasks, including Image and Video recognition, Image Analysis and Classification, Media Recreation, Recommendation Systems, Natural Language Processing, and more. The progress in Computer Vision, particularly with Deep Learning, has evolved and refined over time, primarily revolving around a specific algorithm, the Convolutional Neural Network. O'Shea and Nash (2015)

A Convolutional Neural Network (CNN) is a Deep Learning algorithm designed to process input images, assigning significance through learnable weights and biases to different elements or objects within the image, enabling it to distinguish and differentiate between them. The structure of a CNN is similar to the connectivity pattern of neurons in the human brain, and draws inspiration from the organization of the visual cortex. Each neuron responds to stimuli within a limited region of the visual field, referred to as the receptive field. A combination of these fields overlaps to encompass the entire visual area.

CNN is a non-parametric approach, and the pre-processing demand in CNN is significantly reduced compared to other classification algorithms. In contrast to primitive methods where filters are manually engineered, CNN possess the capability to learn these filters or characteristics through sufficient training. An image is represented as a matrix of pixel values, and in contrast to traditional Feed-Forward Neural

Networks, CNNs demonstrate exceptional accuracy, particularly when dealing with complex images. By employing appropriate filters, a CNN can effectively capture spatial and temporal dependencies in an image. This architecture achieves improved fitting to the image dataset by reducing the number of parameters and facilitating the reuse of weights. In essence, the network can be trained to comprehend the intricacies of the image more effectively. Thus, CNN is an ideal choice for extracting our microstructure image features.

The succeeding paragraph will showcase key parameters within CNN and some of them that require adjustment to achieve improved training performance.

1. Input image size: The standard input image has an RGB format and an example for input image has been shown in Fig. 2.3.

2. Kernel size, stride and padding: The dashed square in fig. 2.1 represent a kernel which will scan through the whole image to extract image features. Then the stride length define how long the kernel will move for each scan. For example, if the input image I has a size as $5 \times 5 \times 1$, the kernel size K is $3 \times 3 \times 1$ and the stride length equal to 1, the kernel will only move left/right or up/down one pixel length and the kernel will totally shift 9 times for the input image size $5 \times 5 \times 1$.

The operation yields two types of outcomes — one where the convolved feature undergoes a reduction in dimensionality compared to the input, and the other where the dimensionality either increases or remains unchanged. This is achieved by employing Valid Padding in the former case and Same Padding in the latter.

By expanding the 5x5x1 image to a 6x6x1 image and subsequently applying the 3x3x1 kernel, the resulting convolved matrix maintains dimensions of 5x5x1. This aligns with the concept of Same Padding. Conversely, when conducting the same operation without padding, the resulting matrix assumes the dimensions of the kernel itself (3x3x1), leading to what is known as Valid Padding.

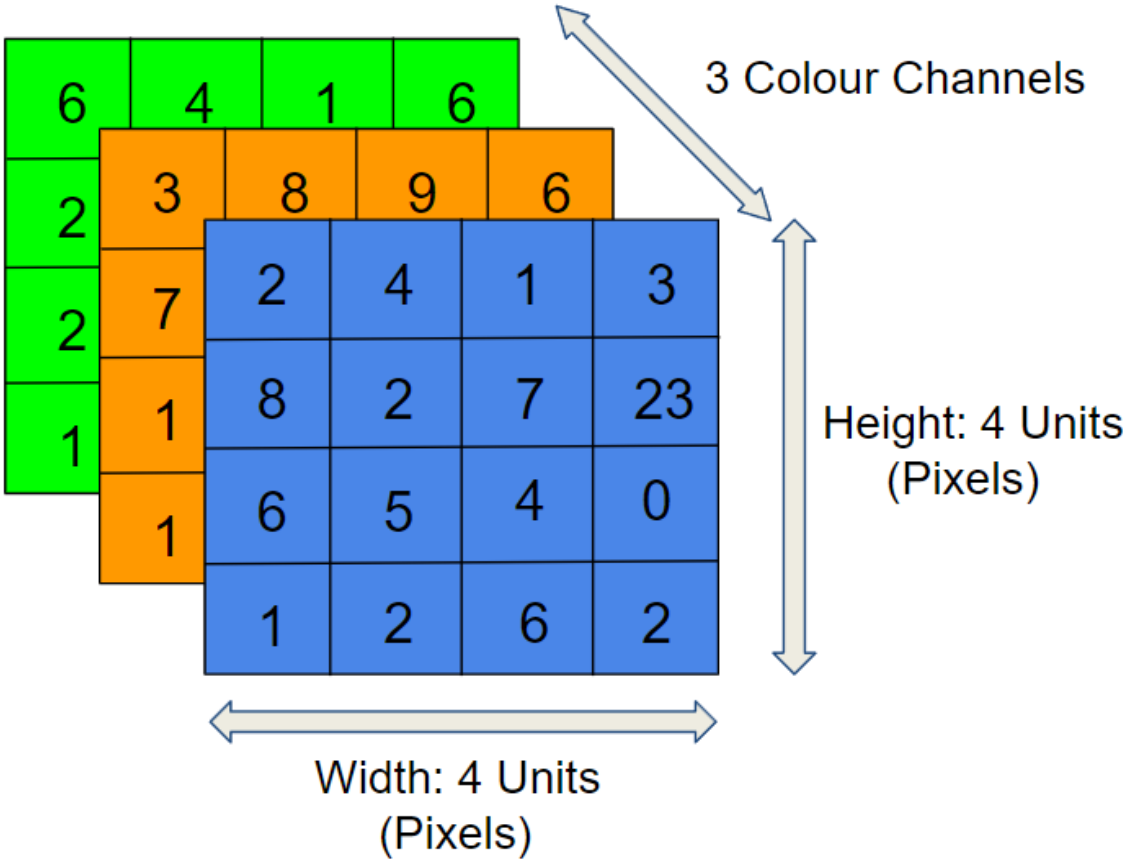

Figure 2.3: An example for CNN's input image

The Convolution Operation aims to extract high-level features, such as edges, from the input image. ConvNets can comprise multiple Convolutional Layers. Typically, the initial ConvLayer is designed to capture low-level features like edges, color, and gradient orientation. As additional layers are introduced, the architecture evolves to comprehend high-level features, providing a network with a comprehensive understanding of images in the dataset, akin to human perception.

Pooling Layer

Much like the Convolutional Layer, the Pooling layer plays a role in diminishing the spatial size of the Convolved Feature. This reduction is intended to lessen the

computational demands for processing data through dimensionality reduction. Additionally, it proves beneficial for extracting dominant features that exhibit rotational and positional invariance, thereby contributing to the effective training of the model.

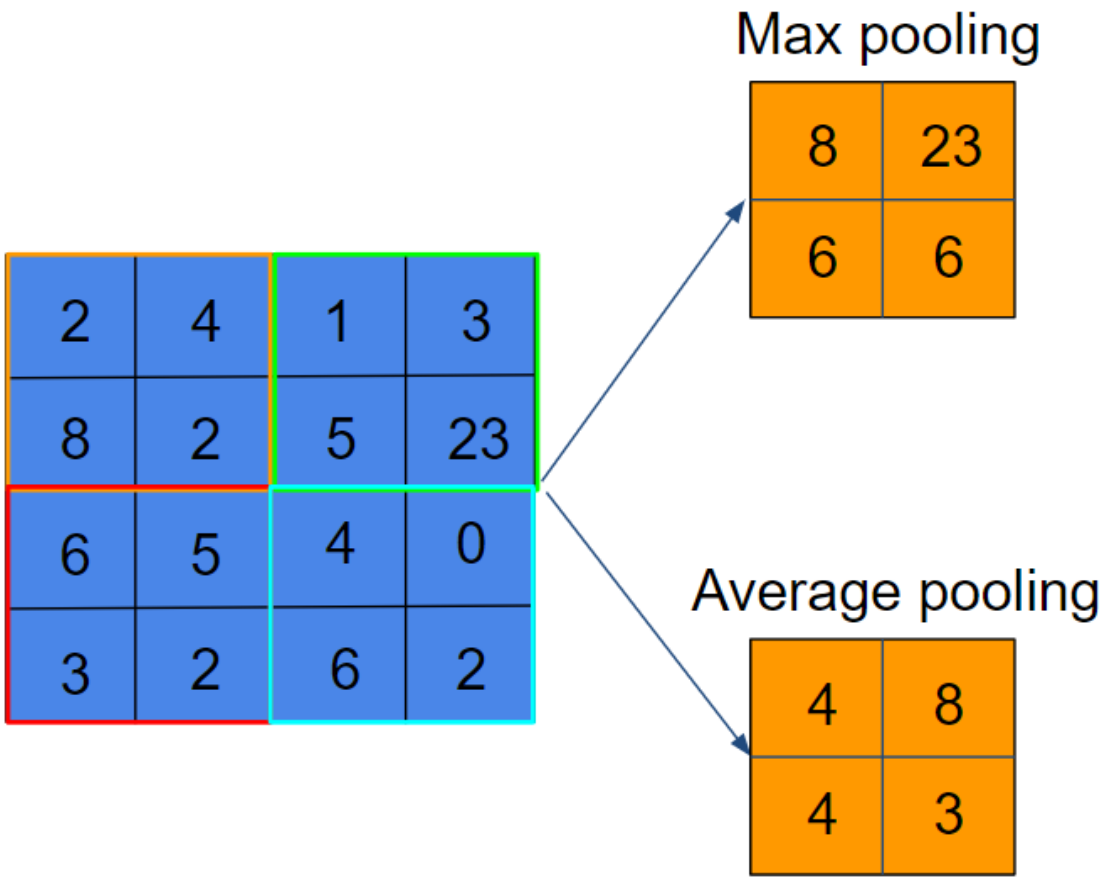

Figure 2.4: An example for pooling operation

Pooling comes in two varieties: Max-Pooling and Average-Pooling. In Max-Pooling, the maximum value from the kernel-covered portion of the image is returned, whereas Average-Pooling provides the average of all values within the kernel-covered portion.

Max-Pooling not only acts as a noise suppressant by discarding noisy activation

but also serves as a denoising mechanism alongside dimensionality reduction. Max Pooling preserves the most salient features of the feature map, resulting in a sharper image compared to the original. Conversely, the Average-Pooling layer operates by calculating the average of the pool and preserves the average values of features within the feature map, contributing to image smoothing while maintaining the essential features.

The Convolutional Layer and the Pooling Layer, together form the i-th layer of a Convolutional Neural Network. Depending on the complexities in the images, the number of such layers may be increased for capturing low-level details even further, but at the cost of more computational power.

After going through the above process, the model has been enabled to understand the features. Regular CNN will flatten the final output and feed it to a regular Neural Network for classification purposes. However, our case also need the model to recognize the time evolution between each microstructure image. After flatten, CNN will be connected with a recurrent neural layer(LSTM).

Recurrent Neural Layers

A recurrent neural network (RNN) is a category of artificial neural networks designed for processing sequential or time series data. These deep learning algorithms are commonly employed for tasks involving temporal or ordinal aspects, such as language translation, natural language processing (NLP), speech recognition, and image captioning. They find applications in popular services like Siri, voice search, and Google Translate. Similar to feedforward and convolutional neural networks (CNNs), recurrent neural networks undergo training using data to acquire knowledge. What sets them apart is their "memory", as they incorporate information from previous inputs to influence the current input and output. Unlike traditional deep neural networks

that assume independence between inputs and outputs, recurrent neural networks base their output on prior elements within the sequence. While considering future events would enhance predictions, unidirectional recurrent neural networks lack the ability to incorporate such events into their forecasts.

However, traditional RNNs are known to be inefficient in capturing the temporal information for long sequences, particularly due to the vanishing gradient problem where information from pastime steps decreases exponentially. Long Short-Term Memory (LSTM) networks represent a modified iteration of recurrent neural networks, facilitating improved retention of past data in memory and addressing the vanishing gradient problem encountered in RNNs. LSTM proves effective in tasks involving the classification, processing, and prediction of time series data with unknown time lags. Training is accomplished through back-propagation. Within an LSTM network, three gates are shown in fig 2.5:

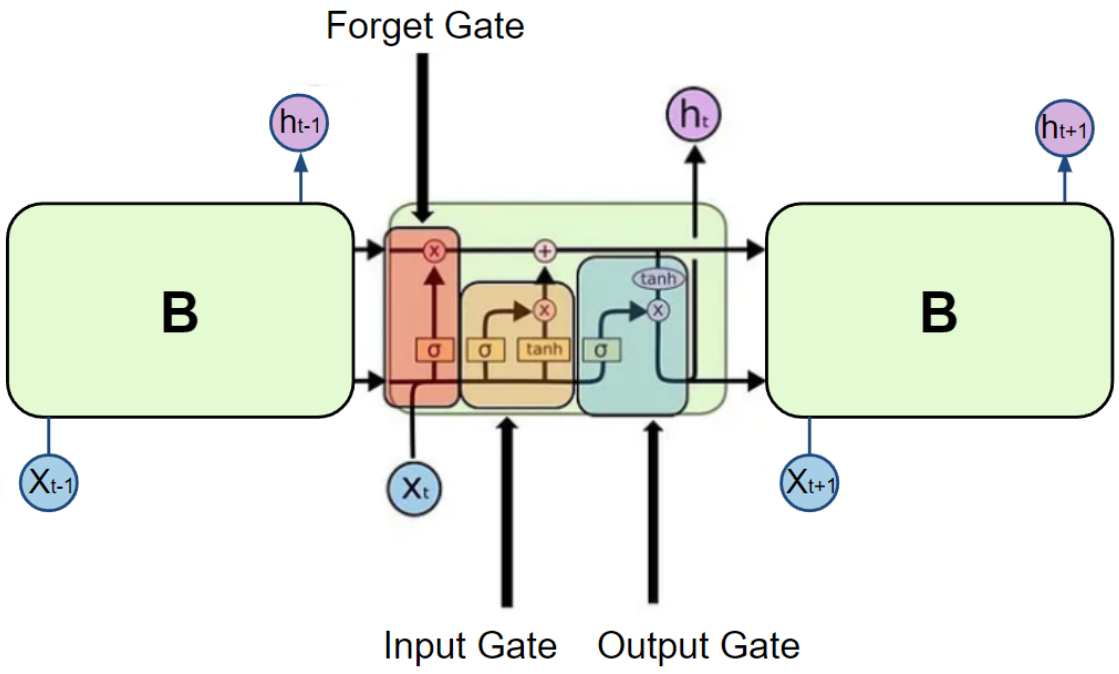

Figure 2.5: Long shot term memory

1. Input gate: determines the value from the input that should be utilized to modify the memory. The *sigmoid* function governs which values to allow through (ranging from 0 to 1), while the *tanh* function assigns weights to the passed values, determining their level of importance within a range from -1 to 1.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.9}$$

$$\widetilde{C}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{2.10}$$

2. Forget gate: determines which information to discard from the block. This is determined by the *sigmoid* function, which assesses the previous state ($ht - 1$) and the content input (Xt), producing a value between 0 (indicating omission) and 1 (indicating retention) for each element in the cell state $Ct - 1$.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.11}$$

3. Output gate: the input and the memory of the block is used to decide the output. *Sigmoid* function decides which values to let through $0, 1.$ and *tanh* function gives weightage to the values which are passed deciding their level of importance ranging from -1 to 1 and multiplied with output of *Sigmoid*.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{2.12}$$

$$h_t = O_t * tanh(C_t) \tag{2.13}$$

Tensor Dimension

For an input image of dimension $32\Delta x \times 32\Delta y$, the input to the convolutional layer is a tensor of $10 \times 32 \times 32$, while the output has a dimension of $10 \times 32 \times 32 \times 256$. This is represented as "Tensor decomposed Conv2d-1" in the CRNN architecture

(see Fig. 2.1). In this algorithmic sequence, the convolutional layer is followed by a max-pooling layer which is responsible for downsampling the output produced from the previous layer. A pooling operation is performed to reduce the CNN framework's computational cost, which causes dimensional reduction from $10 \times 32 \times 32 \times 256$ to $10 \times 16 \times 16 \times 256$. A combination of convolutional and max-pooling layers is useful in extracting the low-level features, such as the presence of phase boundaries. We additionally include convolutional "Tensor decomposed Conv2d-2" and max-pooling layers to extract higher-level features, such as phase boundary curvature, that enable complete feature extraction and training. Also, the padding for all the convolutional and max-pooling layers ensures that the convolved features retain their dimensions for an entire sequence.

Within the proposed workflow shown in Fig. 2.1, the LSTM layer will output a tensor of size 10 times the total number of LSTM's repeat modules. Next, the dense layer will output 10×1024 values for every pixel point. Then, the 10×1024 output matrix is converted to the original input size of $10 \times 32 \times 32 \times 1$ before comparing with the input tensor using a mean square error (MSE) function given by:

$$\text{MSE} = 1/n \sum_{i}^{n} (Y_i - \bar{Y_i})^2, \tag{2.14}$$

where $\bar{Y_i}$ is the prediction value, Y_i is the actual value, and n is the total number of predictions.

For microstructure emulation using CRNN, there are several parameters and hyperparameters that need to be tuned, which requires prior experience and rounds of trial. For example, in the present work, we incorporated 32, 64, 128, 256, or 512 filters without altering other parameters and hyperparameters to optimize the losses in the validation database.

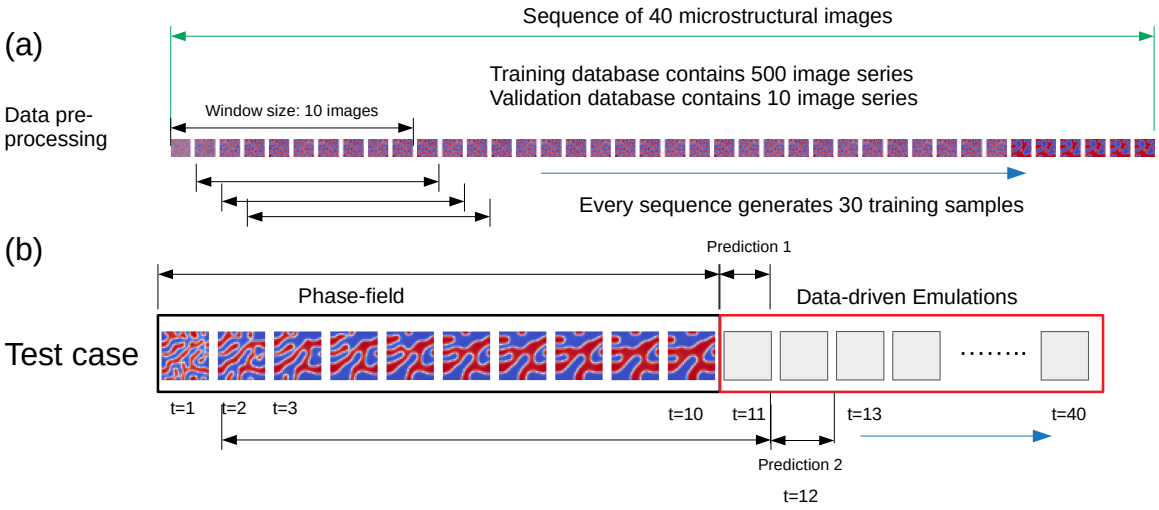

Figure 2.6: (a) Structure of the training and validation databases. (b) Schematic diagram showing our data-driven emulation methodology, as noted by Peichen et al. (2023).

Dataset Pre-processing (Training, Validation and Test)

Using the phase-field constitutive equations outlined in section 2.1, we first generate the training, validation, and test datasets that are required to train and validate the proposed CRNN-based algorithm. The validation dataset is used to tune the neural network while maintaining the associated losses to a minimum during this process. After achieving the lowest validation loss, CRNN outputs an image corresponding to every new microstructural data in the test set, which does not contain any dataset that was used for training or validation. Fig. 2.6(a) shows an exemplary training dataset which comprises a series of 40 snapshots of phase separating microstructures obtained every 1000 timesteps using the phase-field method. As shown, the first ten images represent the first training sample, the next 10, the second sample, and so on. Therefore, the output corresponding to the first training sample is overall the eleventh image in sequence. Thus, we have a total of 30 training samples and 30

responses from every image sequence. We generate 500 such image sequences using different initialization of the phase-field i.e. seed number, with other material-specific parameters consistent, such that we have a total of 15000 training samples. Similarly, we generate a validation dataset with 300 validation samples.

In total, we obtain 10 testing datasets, one of which is shown in Fig. 2.6(b). The first ten phase-field images are used as input to the DDE for predicting the microstructure corresponding to $t = 11$. Then the phase-field images from $t \in \{2, 10\}$ in addition to the predicted image are rendered as the second input in sequence for predicting the microstructure corresponding to the $t = 12$. In this manner, we continue to predict the microstructures sequentially until $t = 40$.

The phase-field model that was used to generate microstructures for the training dataset are natively coded in C++. The DDE model is constructed using Keras library in Anaconda Python 3. In the spirit of establishing a fair comparison between the DDE and phase-field method, both the codes are serially-coded or executed on a single CPU running Linux OS.

2.2.2 Interpolation Prediction via Gaussian Process Regression

This approach is mainly developed by Ashif et al. (2023) and it is based on representing the microstructure evolution as a tensor and obtaining a low-dimensional tensor decomposition of the two-point correlation functions representing the evolving microstructures. The tensor decomposition do not require any unfolding and simultaneously preserves the spatio-temporal relationships, therefore overcoming the limitations of PCA. With the low-dimensional representation at hand, we then employ GPR—a non-parametric nonlinear regression approach to predict the microstructure evolution in the low-dimensional space at arbitrary time steps that cannot be accomplished without running multiple phase-field simulations with distinct timestep widths. For training, we generate spinodal decomposition via high-fidelity phase-field simulations. After we obtain the predictions from GPR, we use a hybrid input-output phase retrieval algorithm to reconstruct the microstructures from the predicted two-point correlation functions. An outline of the proposed methodology is presented in Fig. 2.7.

Two-point Correlation

We first begin with a mathematical description of the microstructures. For a discrete microstructure as described in fig. 2.2.2, we denote m_s^n as the probability of finding the local state $n = 1, 2, ..., N$ at a spatial location $s = 1, 2, ..., S$ such that $\sum_n m_s^n = 1$ and $m_s^n > 0$. Note that for a two-phase microstructure, $N = 2$. Such a mathematical representation allows one to characterize the microstructure using various statistical measures such as n-point statistics, chord-length, and nearest neighbor functions, as demonstrated by Torquato (2002). One of the most commonly employed statistics, as introduced by Kalidindi (2015), is the two-point spatial correlation that captures

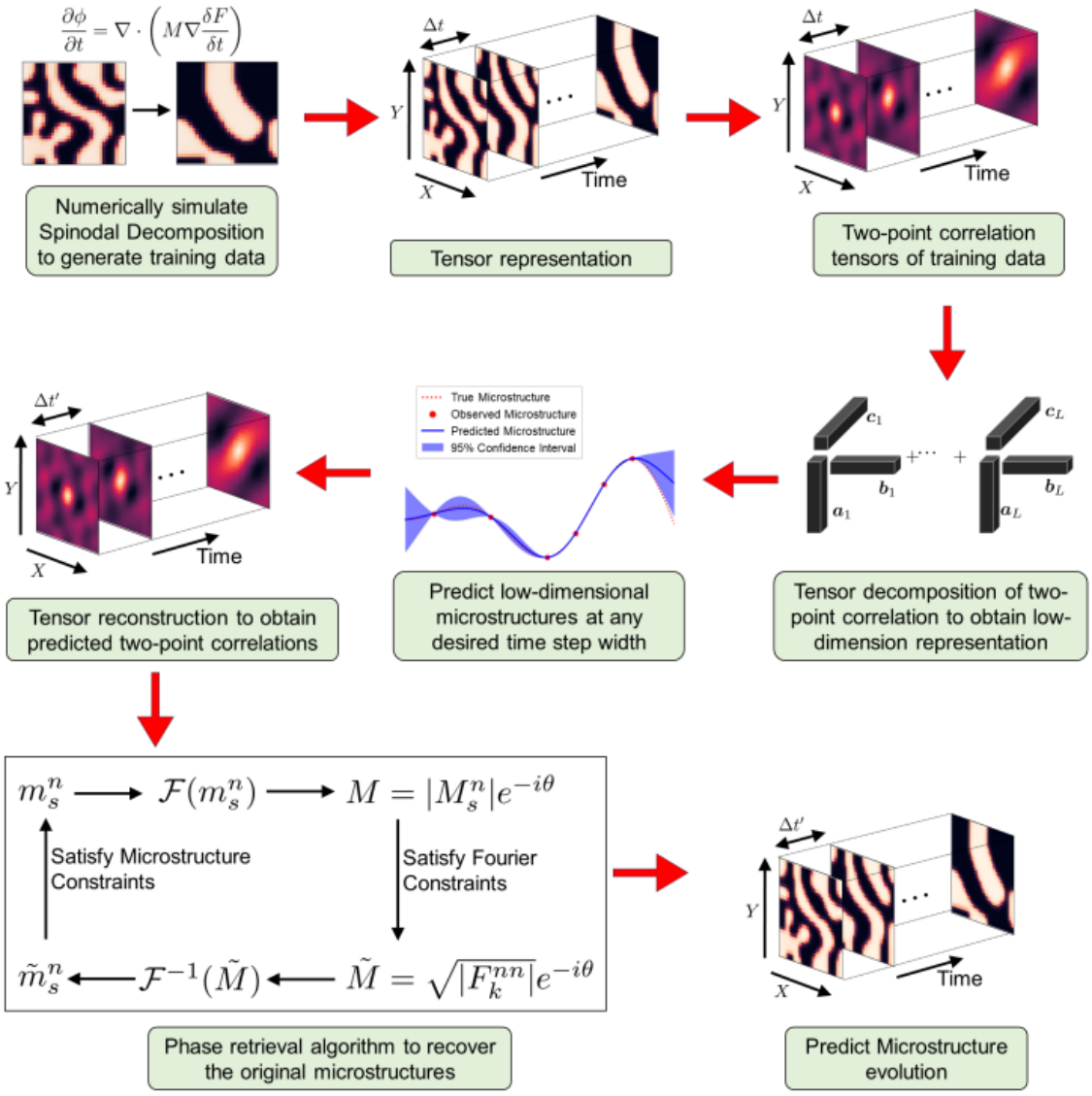

Figure 2.7: Flow sequence outlining our microstructure emulation approach via Gaussian process regression, as proposed by Ashif et al. (2023).

the conditional probability of finding a local state p at location $s + r$ given that a local state n is present at location s and is given as:

$$F_r^{np} = \frac{1}{S_r}\frac{1}{J}\sum_{s}^{Sr}\sum_{j=1}^{J}{}^{(j)}m_s^{n(j)}m_{s+r}^{p} \tag{2.15}$$

where S_r represents the number of admissible values of r for a given s and is upper bounded by S. Note that superscript $j = 1, 2, ...J$ represents different realizations of the microstructure.

Two-point correlation begin by considering a statistical representation of the microstructures. Mathematically, a microstructure function m_s^n is represented as the probability of finding some local state n at a spatial location $s \in \mathbb{R}^2$ for two-dimensional microstructures. Given the stochastic nature of microstructures in space, we first obtained their statistical representation using two-point correlations. Intuitively, the two-point correlations, denoted as f_{np}^r, capture the likelihood of observing some combination of microstructure states (n and p) at two randomly selected locations separated by a distance of r. For a two state microstructure (say 0 and 1), as we are concerned in this work, there are a total of four combinations of states, i.e., 00, 01, 11, and 10 and therefore four set of two-point correlations. However, two-point correlation functions have inherent redundancies in them, making them interdependent. For instance, the probability of observing states 1 and 0 in spatial bins separated by a distance of r, i.e., f_{10}^r is the same as observing the two states in the reverse order separated by a distance of $-r$, i.e., f_{01}^{-r}. Taking into account all redundancies, in the case of a two-state microstructure, it is sufficient to compute either an autocorrelation or a cross-correlation function, as the remaining correlations can be reconstructed, as suggested by Frisch and Stillinger (1963). Since the two-point spatial correlation summarizes the pairwise distribution of local states within the microstructure, they are particularly useful in tracking microstructure evolution. Details of the two-point

correlation functions are provided in the Methods section. To efficiently calculate the two-point correlations, we subscribed to Discrete Fourier Transform (DFT) of the microstructure function m_s^n. We first note that the DFT of the two-point statistics is given as:

$$F_k^{np} = \mathcal{F}(F_k^{np}) = \frac{1}{S}|M_k^n|e^{i\theta_k^n}|M_k^p|e^{i\theta_k^p} \tag{2.16}$$

where F is the DFT operator, $|M_k^n|$ denotes the magnitude component, and θ_k^n

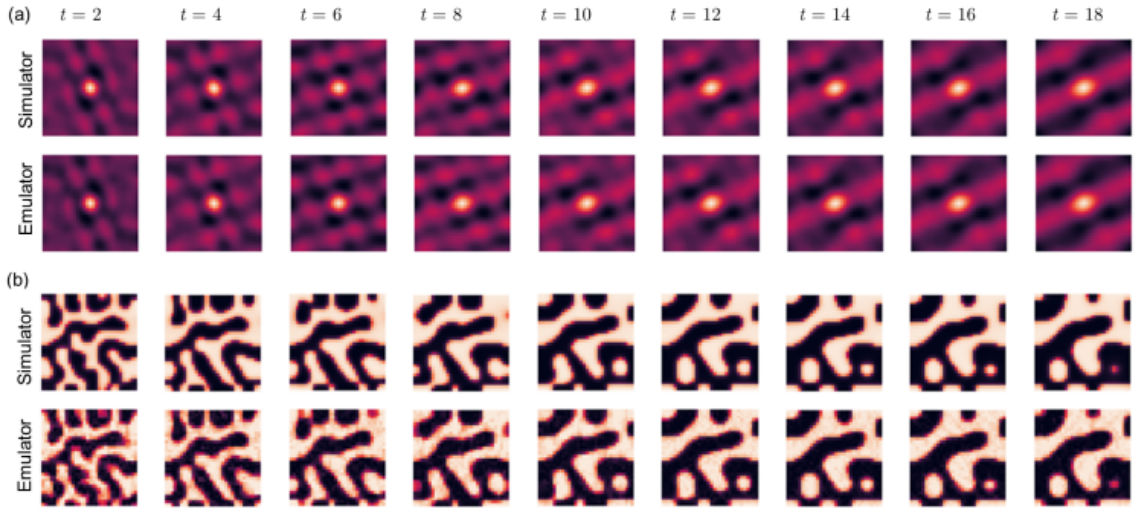

Figure 2.8: Frame-by-frame comparison of microstructure evolution simulated using phase-field method with the corresponding emulation. The top row shows the original sequence of the (a) two-point correlation and (b) auto-correlation functions of the phase-field generated dataset while the emulations are plotted in the bottom row, as demonstrated by Ashif et al. (2023).

represents the phase component of the microstructure function m_s^n for state n. For $n = p$, the DFT of the two-point statistic gives us the auto-correlation function and for $n \neq p$, the DFT gives the cross-correlation function. As mentioned earlier, we only need either an auto-correlation or cross-correlation function for a two-phase

microstructure and the rest could be recovered. In the remainder of the discussion, we work with the auto-correlation function of one of the states. Figures 2.8(b) and 2.8(a) show representative examples of the microstructure and its corresponding auto-correlation function, respectively. We note that auto-correlation functions follows a similar evolution trend as their corresponding microstructures, i.e., they evolve rapidly during the initial stages and slowly in the later stages of coarsening.

Canonical Polyadic (CP) Tensor Decomposition

We next look at the tensor representation and decomposition of the auto-correlation functions. Microstructure evolution lies in a high-dimensional space, e.g., the evolution of a $32 \times 32 \times 32$ microstructure, which is the smallest dataset we are working with, for just 10 time-steps (or samples) lies in a 327,680 dimensional space. To handle such high-dimensional data structures, microstructures and their relevant statistics such as two-point correlations have traditionally been represented via matrix or vector-based methods such as principal component analysis that require flattening them into vectors, as presented by Zapiain et al. (2021); Herman et al. (2020). However, upon vectorization, microstructure sequences lose their topological structures and dependencies across spatial and temporal modes. The absence of an appropriate mathematical representation has precluded the development of models that can efficiently capture the higher order nonlinear patterns in microstructure evolution. In this regard, we provide a tensor representation of microstructure evolution. More specifically, we consider microstructure evolution as a three-way tensor with dimensions of Parameter $\times$ Space $\times$ Time. Mathematically, microstructure evolution will be represented as $X : P \times S \times T$ where P, S, and T represents the dimension of parameter, spatial, and temporal modes, respectively. For the emulation results reported below, the parameter mode is kept constant. To obtain a low-dimensional representation of the microstructure evolution, we employ a tensor decomposition approach known as Canonical Polyadic (CP) decomposition. Tensor decomposition is an extension of conventional PCA for multi-way datasets that do not require any unfolding. Several methods for tensor decomposition exists such as CP as demonstrated by Richard (1970), Tucker as noted by Tucker (1966), tensor train as presented by Oseledets (2011) etc.

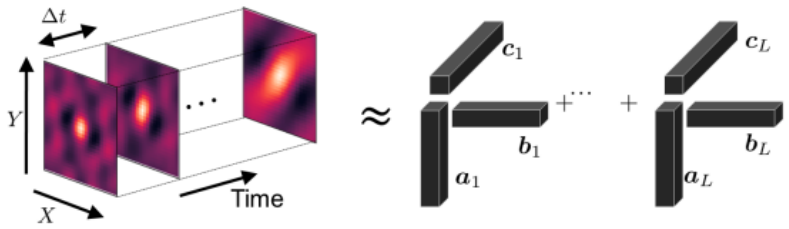

Figure 2.9: Schematic of tensor decomposition as presented by Ashif et al. (2023).

CP decomposition also known as $PARAFAC$ decomposition is a generalization of singular value decomposition to multi-way tensor datasets. Originally proposed by Carroll and Chang (1970), CP decomposition aims to decompose a tensor χ of dimension $n \times n \times T$ into an outer product of rank one tensor given as:

$$\chi \approx \sum_{i=1}^{R} \lambda_i a_i \circ b_i \circ c_i = [[\lambda, \mathbf{A}, \mathbf{B}, \mathbf{C}]] \tag{2.17}$$

where R is a positive number representing the number of rank one tensors needed to represent χ, λ_i represents the weights, $a_i \in \mathbb{R}^n$, $b_i \in \mathbb{R}^n$ and $c_i \in \mathbb{R}^T$ represents the rank one tensors, $\lambda = [\lambda_1, ..., \lambda_R]$ and lastly, A, B, and C are the factor matrices containing the rank one tensors as columns, i.e., $A = [a_1, ..., a_R]$ and likewise for B, C. Ideally, the number of rank one tensors required to recover the tensor χ is equal to the rank of χ. However, there is no finite algorithm for computing the rank of a tensor, leading to the approximation in Eq. 2.17. To identify the minimum number of rank one tensors needed for CP decomposition, we start with an arbitrary rank and gradually increase until a good fit (say 95%) is obtained. With the rank R specified, we use the alternating least squares (ALS) to identify the rank one tensors by minimizing the Frobenius norm between the original χ and the reconstructed tensor $\hat{\chi}$ i.e., $\hat{\chi} = argmin_{\hat{\chi}} ||\hat{\chi} - \chi||_F$. For a three-way tensor as considered in our case, the ALS approach fixes one of the factor matrices to solve for the other two and continues iteratively until some convergence criterion is reached. Detailed procedures

for ALS can be found in other sources, as outlined by Kolda and Bader (2009). A schematic of the CP decomposition approach is presented in Fig. 2.9. n this study, we utilize the parafac function from the Tensorly package in Python, as described by Kossaifi et al. (2019).

In this work, we implement CP decomposition as it factorizes the microstructure tensor into an outer-product of rank one tensors that are simple to handle using GPR. The number of such rank one tensors represents the rank of the original microstructure tensor. Intuitively, the rank of the microstructure tensor increases as the complexity in the spatio-temporal makeup of the microstructures increases. However, computing the rank parameter is NP hard and therefore, we perform a sensitivity analysis to determine the optimal rank. Essentially, we check the reconstruction error for varying ranks and select the rank for which the reconstruction error is within 5% of the lowest reconstruction error. We measure the reconstruction error using normalized mean square error:

$$NMSE = \frac{||X - \hat{X}||^2}{\sqrt{||X||}} \tag{2.18}$$

Figure 2.10 shows the reconstruction error for different ranks. We obtain the lowest average normalized mean squared error (NMSE) of 86.17 for a rank 400. However, for a lower rank of 300, we obtain the average NMSE of 86.32 which is within 5% of the lowest normalized MSE. Therefore, to reduce computational complexity, we fix the tensor rank to 300 for the rest of the discussion.

Before we present the results on GPR fitting, we provide an overview of the microstructure data from a low-dimensional perspective. Figure 2.11(a) shows the first rank-one tensor across the temporal domain after performing CP decomposition and its corresponding trend is shown in red. Clearly, the rank one tensor component shows a gradual increase as the microstructure evolves. In Fig. 2.11(b) we notice some seasonality effect that is obtained after removing trend from the original data. Since

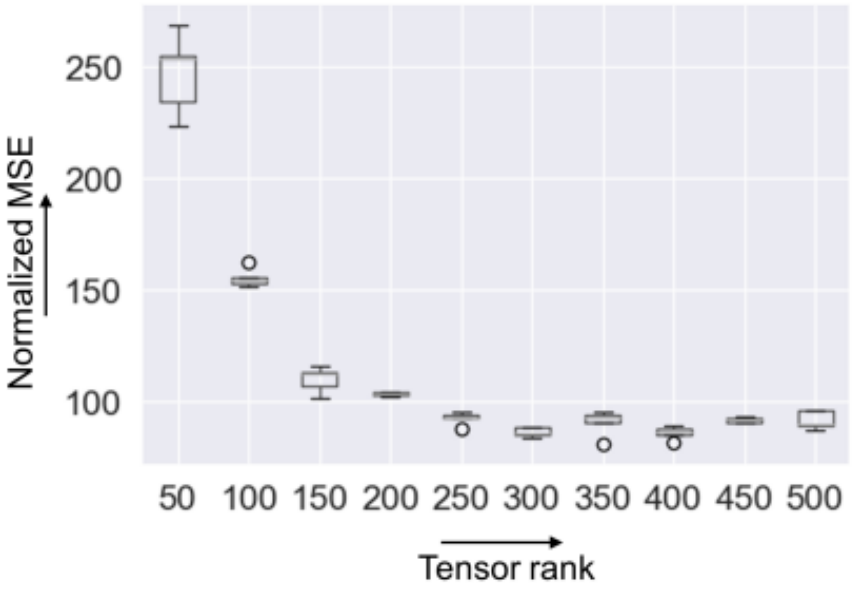

Figure 2.10: Variation in the normalized MSE for microstructure reconstruction with different tensor ranksAshif et al. (2023).

the magnitude of seasonality is almost one-tenth of the magnitude of the original data, we can assume these variations to be noise. Furthermore, in Figs. 2.11(c) and (d), we present the auto-correlation (i.e., correlation between the data and its lagged version) and the partial auto-correlation function which is the same as auto-correlation but without the correlation effect from data in between. From the auto-correlation plot, we can infer that successive data points are highly correlated, especially for smaller lag values. From a microstructure standpoint, this indicates that the microstructures sequences at successive time instants are highly correlated which is also evident from Fig. 2.8. The partial auto-correlation plot in Fig. 2.11(d) shows a strong correlation if lag is 1, i.e., for successive data points, but it drops rapidly afterwards. This again reaffirms our observation that the microstructure images are strongly correlated.

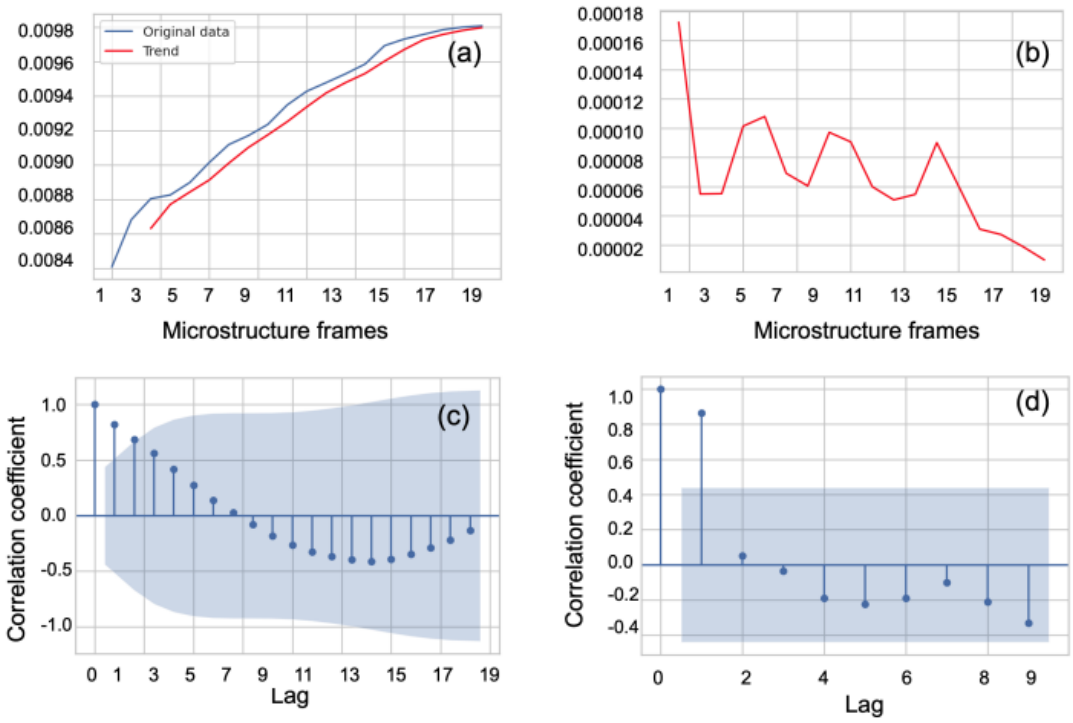

Figure 2.11: Exploratory analysis of the microstructure data in the low dimensional space. (a) The first rank one tensor across the temporal dimension (referred as original data) and the corresponding trend, (b) seasonality in the original data, (c) auto-correlation function and (d) partial auto-correlation function, as demonstrated by Ashif et al. (2023).

Gaussian Process Regression (GPR)

After we obtain the low-dimensional representation of the microstructures, we use GPR to emulate microstructural evolution. Once the microstructure evolution is projected in the rank one tensor space, we can learn the microstructural variations using statistical and machine learning techniques. In this work, we choose Gaussian process regression, a non-parametric regression model that is well suited to capture the nonlinear function variability. The flexibility of Gaussian process regression

arises from the assumption that the underlying data–in this case rank one tensors, $a_i, b_i, c_i, i = 1, 2, ..., R-$ are drawn from a Gaussian Process with mean function $\theta(x)$ and covariance function $k(x, x^{'})$,i.e., $f(x) \sim \mathcal{GP}(\theta(x), k(x, x^{'}))$ where x is the domain for each of the rank one tensors. For simplicity, we assume a zero mean Gaussian process prior in this work.

Let us represent the time step and the target variable for the i^{th} rank one tensor be $(T, c_i) = (t_1, c_{i1}), (t_2, c_{i2}), ..., (t_o, c_{io})$. For any $t_* \notin t_1, t_2, ..., t_o$, the posterior predictive distribution is given as $c_{i*}|T, c_i, t_* \sim \mathcal{N}(\bar{c_{i*}}, cov(c_{i*}))$ where

$$\bar{c_{i*}} = K(t_*, T)[K(T, T) + \sigma^2 I]^{-1} c_i \tag{2.19}$$

$$cov(c_{i*}) = K(t_*, t_*) - K(t_*, t_*)^{'}[K(T, T) + \sigma^2 I]^{-1} K(T, t_*) \tag{2.20}$$

To model the covariance structure, we use a Matérn kernel function given as,

$$K(t, t^{'}) = (1 + \frac{\sqrt{3}(t - t^{'})^2}{\sigma^2}) exp(-\frac{sqrt3(t - t^{'})^2}{\sigma^2}) \tag{2.21}$$

where σ^2 is the length scale parameter. We train one GPR for each of the rank one-tensors across the temporal dimension obtained from training frames 0, 1, 3, 5, . . . , 19 and subsequently, predict the rank-one tensors for frames 0, 1, 2, . . . 18, 19. For every training microstructure sequence, we perform hyperparameter (σ^2) optimization by maximizing the log-marginal likelihood function. We tune the hyperparameters individually for every rank one tensor. After obtaining the predictions, we reconstruct the auto-correlations by using the outer product of the predicted one-dimensional tensors across every mode.

Microstructures Reconstruction

Once we have predicted the auto-correlation functions, the last step involves recovery of microstructures from the auto-correlation functions. This implies extracting phases from the amplitude information since the phase information is typically lost during the convolutions performed for determining auto-correlation functions. Thus, we employ an iterative hybrid input-output phase retrieval algorithm as proposed by Fienup (1982), based on the widely known Gerchberg-Saxton algorithm, as presented by Gerchberg and Saxton (1972b).

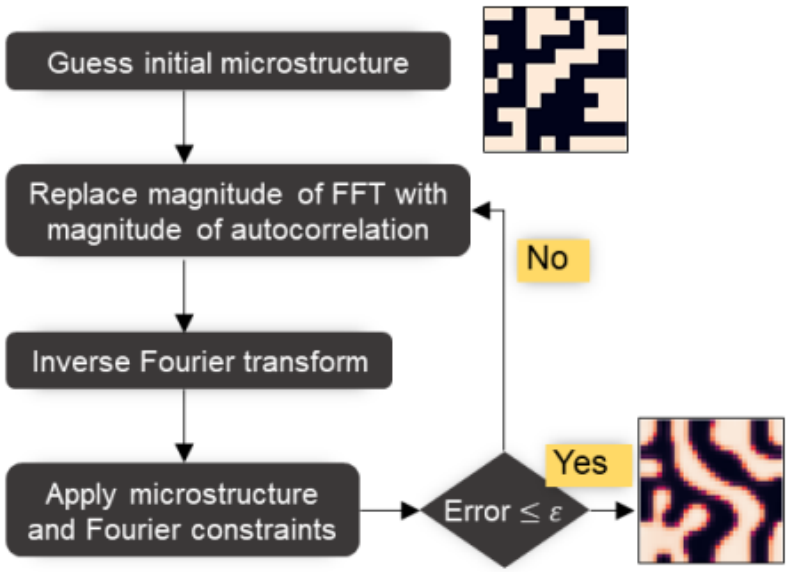

Figure 2.12: Flow chart of the hybrid input-output algorithm for microstructure reconstruction from the predicted auto-correlation functions, as outlined by Ashif et al. (2023).

After the microstructure predictions are obtained from GPR, we reconstruct the microstructure using a two-step procedure. First, we reconstruct the two-point correlations corresponding to the predicted values of the rank of tensors. With the

corresponding rank one tensors, the reconstructed two-point correlation is given as:

$$\hat{\chi} = \sum_{l=1}^{R} \lambda_l a_l \circ b_l \circ c_l \tag{2.22}$$

To recover the microstructure from reconstructed two-point correlation $\hat{\chi}$, we use a hybrid input output (hIO) phase retrieval algorithm also known as the Gerchberg-Saxton algorithm, which is proposed by Gerchberg and Saxton (1972a). Note that traditionally, phase-retrieval methods have been reported to suffer from poor reconstruction accuracy and are subject to the initial conditions, as noted in Zapiain et al. (2021); Herman et al. (2020). To overcome these challenges, we embed a padding with zero pixel value around the microstructures which has been shown to significantly improve the reconstruction accuracy as indicated by Liu (2012). For a microstructure of dimension $n \times n$, we set a padding of n on each of the four sides. We now present the algorithm that consists of the following steps:

(a) Initialize the microstructure with random 0-1 values, $m_s^n(0)$.

(b) For a given microstructure $m_s^n(\tau)$ at iteration τ , obtain the corresponding Fourier transform, i.e., $\mathcal{F}(m_s^n) = |M_k^n|e^{i\theta_k^n}$ where $|M_k^n|$ and θ_k^n are respectively the amplitude and phase of the Fourier transform. Here, we removed the iteration number τ to simplify the notations.

(c)Replace the amplitude of the Fourier transform in step (b) with the amplitude of the auto-correlation function ($\hat{\chi}$) predicted in the previous step, such that $\widetilde{\mathcal{F}(m_s^n)} = \sqrt{|\mathcal{F}(\hat{\chi})|e^{i\theta_k^n}}$. Obtain the resulting microstructure using inverse Fourier transform, i.e.,$\widetilde{m_n^s} = \mathcal{F}^{-1}(\widetilde{F(m_s^n)})$.

(d)Finally, we apply the constraints in the real space. Let us consider that Γ contains all the spatial locations where the local state violates the allowable values, e.g., the local state is negative. We update the microstructure in the next iteration as:

$$m_s^n(\tau+1) = \begin{cases} m_s^n(\tau) - \theta\widetilde{m_n^s(\tau)}, & s \in \Gamma \\ \widetilde{m_n^s(\tau)}, & s \notin \Gamma \end{cases} \tag{2.23}$$

where $0 < \theta < 1$ controls the rate of convergence. Additionally, if any of the local states exceed 1, then we reassign the state to 1.

(e)Repeat steps (b)-(d) until the reconstruction error defined as the Frobenius norm, $||m_s^n(\tau) - m_n^s(\tau - 1)||_F$ between successive steps falls below a prespecified threshold.

Note that the amplitude substitution in step (b) is based on the property of auto-correlation and is a crucial step in ensuring that the guess microstructure gradually converges to the true microstructure. In particular, it says that the Fourier Transform of auto-correlation is equal to the square of the magnitude of the corresponding microstructure function. The final step enforces the non-negativity constraint before updating the microstructure function, i.e.,$m_s^n(\tau) \geq 0$ at any iteration τ . A schematic of the hIO algorithm is presented in Fig. 2.12.

Traditionally, phase-retrieval methods are known to suffer from poor reconstruction inaccuracies that depend on the initial conditions, as noted in Zapiain et al. (2021); Herman et al. (2020). To circumvent this issue, we add a padding around the microstructures before extracting the auto-correlation functions during the training phase. The padding around each of the microstructures, although increases the computational complexity, significantly improves the reconstruction accuracy. In this work, we find the optimal padding on every side to be equal to the image dimensions such that a computational microstructure of size 32×32 upon padding scales up to 96×96. The value of pixels in the padding is set to zero. Since the phase retrieval algorithm is iterative, if the normalized MSE of the reconstructed microstructure does not change beyond 5% for five consecutive iterations, the flow sequence exits. The

top row in Fig. 2.8(b) shows the microstructure frames that were not included in the training dataset while the corresponding predictions obtained from the phase-field emulator are shown at the bottom. We note that the predicted microstructures closely resemble the true microstructures.

Dataset Format (Training and Testing)

we first train the emulator using the microstructure evolution obtained at a large time step and predict the microstructure evolution at smaller time steps. In particular, we construct the training data by considering the microstructures at timesteps $2\Delta t$ including the first and the last microstructure frames and predict the microstructures at timesteps Δt. Every microstructure sequence contains $N = 20$ frames and therefore, the training data is composed of 11 frames such that the dimension of the microstructure tensor is $32 \times 32 \times 11$. The input space after CP decomposition is 11×300 in the temporal dimension and 32×300 across each of the two spatial modes. To predict the microstructure evolution at a smaller time step, we now merely have to perform training and testing over the temporal mode.

Chapter 3

RESULTS AND DISCUSSION

3.1 Extrapolation Prediction via CRNN

In this section, we discuss the DDE predictions corresponding to domains of size, $32\Delta x \times 32\Delta y$ and $128\Delta x \times 128\Delta y$, while highlighting the trade-off between accuracy and training efficiency of the novel CRNN approach for predicting microstructural evolution.

3.1.1 Comparison for Domain Size $32\Delta x \times 32\Delta y$ *and* $128\Delta x \times 128\Delta y$

Fig. 3.1 compares the simulated temporal evolution of microstructures of size $32\Delta x \times 32\Delta y$, with the emulated ones at the representative timesteps, starting from distinct seed numbers. The adjacent plots compare the simulated with emulated average domain sizes computed based on the scaling approach reported by Toral et al. (1988); Bray and Emmott (1995). At steady state, the average domain size of microstructures undergoing spinodal decomposition scales as

$$S = At^n \tag{3.1}$$

where, S is the domain size, n is the power exponent, and A is the prefactor. The same approach has been traditionally used to obtain the scaling behavior of phase separating microstructures, one of the more recent examples being that of vapor co-deposited alloy films as demonstrated by Ankit et al. (2019); Raghavan et al. (2020). It is found that when the volume of the training database equals 500 image series, the emulated microstructure although visually comparable shows minor discrepancies

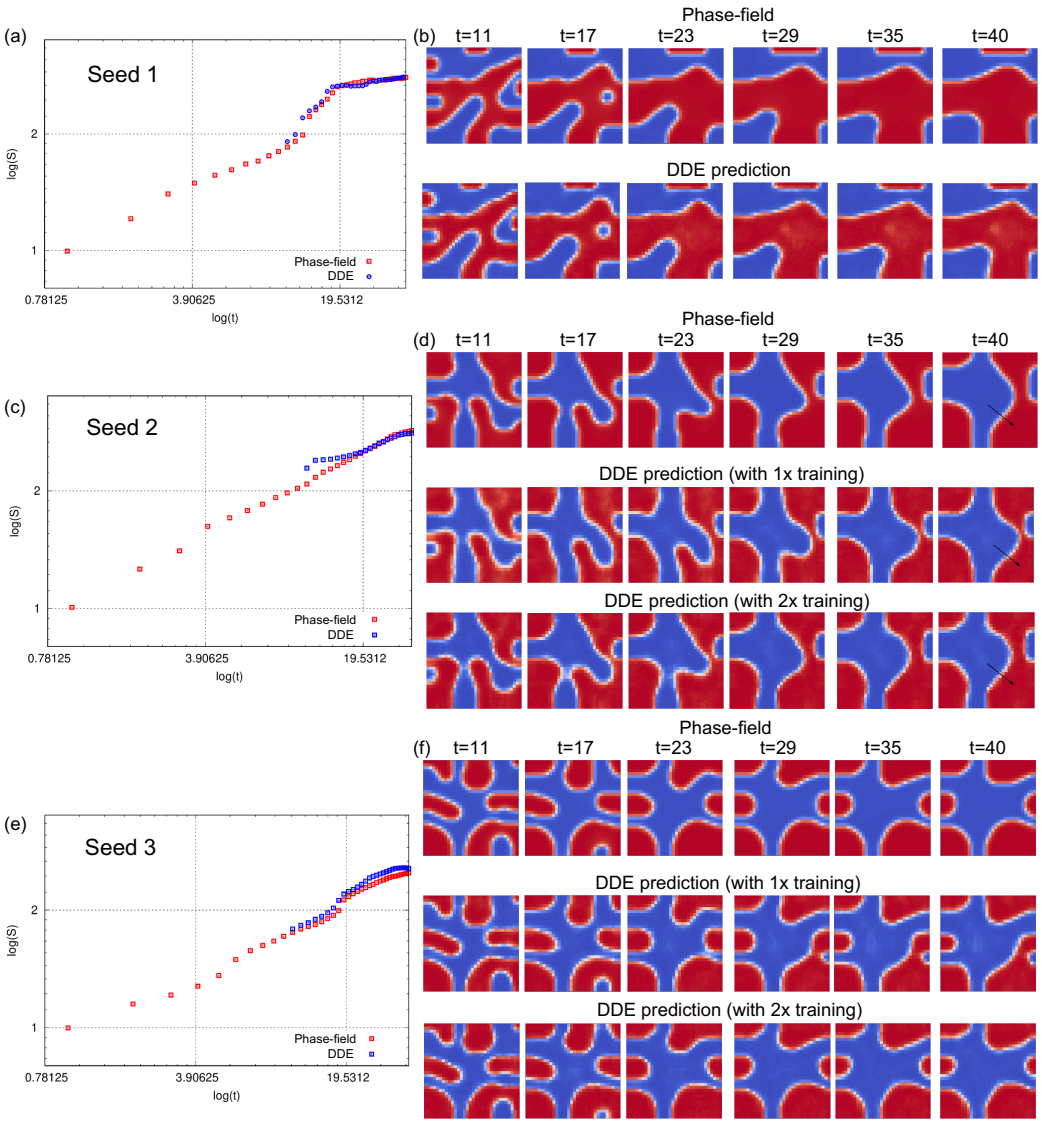

Figure 3.1: Comparing the simulated and the emulated evolution of microstructural evolution at representative timesteps for distinct initial conditions generated by random seeds, as discussed by Peichen et al. (2023). The scaling of the domain size which is obtained by both these techniques is compared in the adjacent plots. Doubling the training volume increases the accuracy of emulations as evident from (d) and (f). '1x training' refers to training dataset comprising of 500 phase-field simulations whereas '2x training' implies twice as much i.e. 1000. The black arrows in (d) indicate the regions where the 1D diffuse interface profiles shown in Fig. 3.4 were plotted.

with respect to phase-field results, the most notable being the thinning of the vertical ligament in Fig. 3.1d. Similar discrepancies can also be noted in Fig. 3.1f, where the globule spanning the right edge does not pinch off unlike in the corresponding phase-field simulated evolution. However, in either case, we were able to correctly emulate the microstructural evolution by doubling the training database volume. Based on a favorable comparison of scaling dynamics as well as the morphological evolution in Fig. 3.1, we conclude that the accuracy of the CRNN-based emulation approach is independent of the initial condition. At this juncture, we highlight relevant differences between our approach with respect to recently reported deep learning approaches which are reported by Zapiain et al. (2021); Yang et al. (2021); Herman et al. (2020). For instance, Yang et al. (2021) implemented an RNN with eidetic 3D LSTM cells to predict the evolution pattern in four different evolution phenomena including plane-wave propagation, grain coarsening, spinodal decomposition, and dendrite growth. Their methodology involved training the RNN with microstructure sequences followed by predicting the microstructure evolution for a new initial microstructure. In Zapiain et al. (2021)'s work, they implemented a LSTM network to predict the microstructure evolution by obtaining a low-dimensional representation using two-point statistics and principal component analysis. While both these previous approaches have demonstrated the viability of using deep learning for microstructure prediction, their application is limited by the following factors: first, deep learning approaches are known to be data-hungry and therefore their performance is highly contingent on the availability of large datasets. For instance, the LSTM network trained by Zapiain et al. (2021) warranted 5000 high-fidelity phase-field simulations, each of which comprised 60 sequential microstructure datasets. Another limitation of the above approaches is the use of principal component analysis (PCA) for dimensionality reduction that has limited applicability in emulating microstructure sequences,

since it entails unfolding images into one-dimensional arrays. In this operation, the spatio-temporal information related to phase boundaries, particularly the width and the curvature, is lost which ultimately manifests as poor reconstruction of emulated microstructures , as demonstrated by Zapiain et al. (2021); Herman et al. (2020). We emphasize that unlike the work of Zapiain et al. (2021), where the microstructure prediction is limited to the final timestep, the CRNN approach reported here has predicted the complete microstructure evolution sequence with a reasonable accuracy. Nonetheless, one of the limitations of our approach is that minor errors are accumulated as the microstructure prediction is extrapolated for future time steps. This is an expected outcome since we have utilized the phase-field generated microstructures from the first 10 time steps to emulate microstructural evolution for the subsequent 30 time steps.

The microstructural evolution emulated using the DDE in a domain of size, $128\Delta x \times 128\Delta y$ is found to be visually comparable to the corresponding phase-field simulations, as observed in Fig. 3.2, and exemplified by the disappearance of the globular phase separating domains within the encircled region.

3.1.2 Further Validations for DDE Predictions

To further validate the DDE in accordance with respect to the benchmark problems reported by Jokisaari et al. (2017), we estimated the total free energy from the emulated micrographs. Depending on the value of the order parameter specific to every grid point in the emulated microstructure, the corresponding bulk and the interfacial energy contributions are added to obtain the total free energy for the entire microstructure. The temporal evolution trend of the emulated free energy minimization shows reasonable agreement with the phase-field free energy minimization as shown in Fig. 3.3. Minor discrepancies in emulations of the 1-D interface profiles of

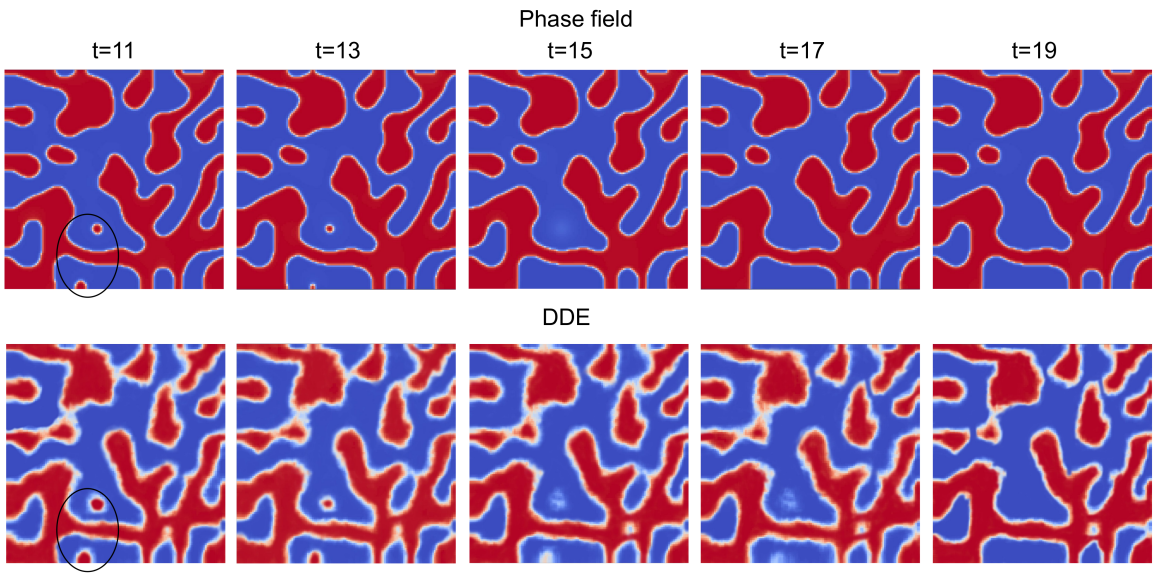

Figure 3.2: The comparison between the phase-field simulation and the data-driven emulation of microstructural evolution at representative timesteps for a simulation box size of size 128×128 grid points, as discussed by Peichen et al. (2023). Encircled regions compare an instance when the disappearance of globular features seen in phase-field simulations are satisfactorily emulated.

ϕ are also observed when the training database volume is smaller. However, the emulated interface profile is found to better converge with the phase-field diffuse interface when the training volume is doubled, as shown in Fig. 3.4. Both these findings indicate the importance of selecting a large training database that ultimately facilitates accuracy in microstructure emulations.

3.1.3 Trade-off Between Accuracy and Training Efficiency

An another advantage of DDE is the minuscule runtimes as compared to phase-field which are tabulated in Table 3.1.

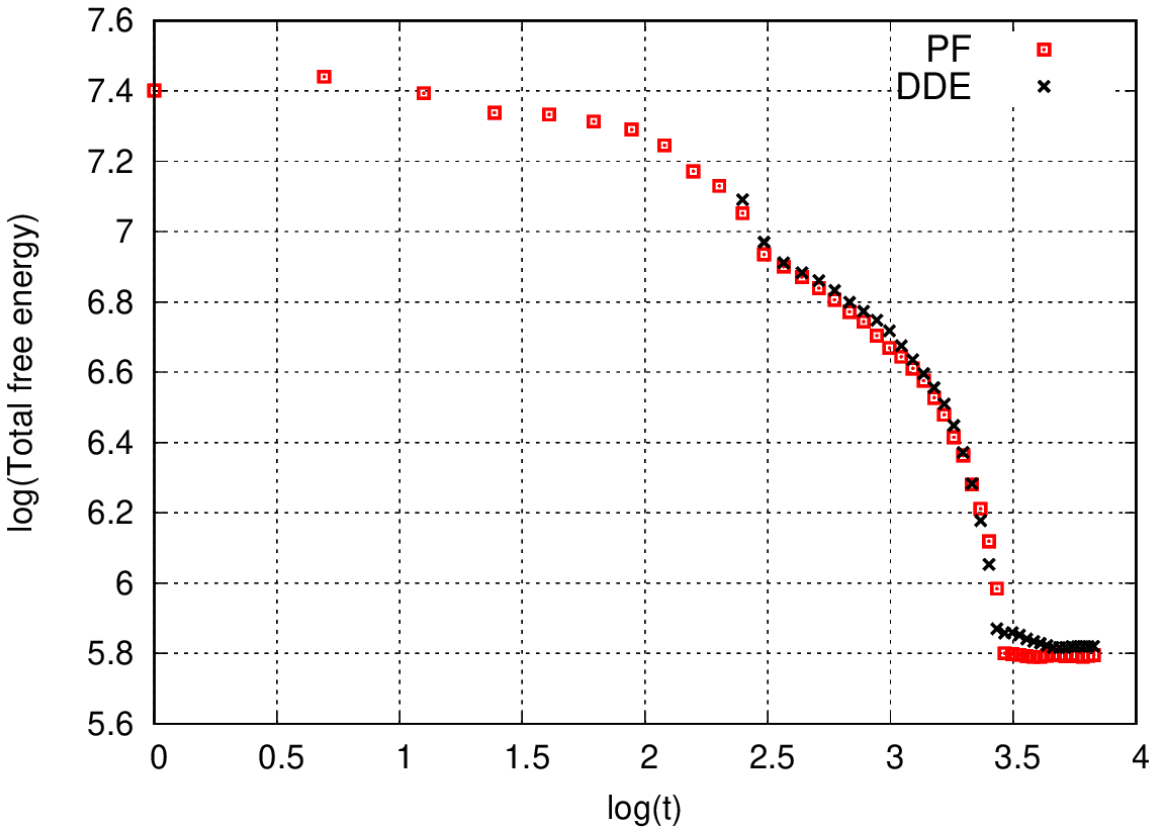

Figure 3.3: Comparison of the temporal minimization of the total free energy obtained from phase-field simulations and DDE, as noted by Peichen et al. (2023). Both trends conform with the benchmark laid by Jokisaari et al. (2017). The total free energy for DDE is estimated from the emulated microstructures by accounting for the bulk and interfacial energy contributions specific to every grid point depending on the value of the order parameter.

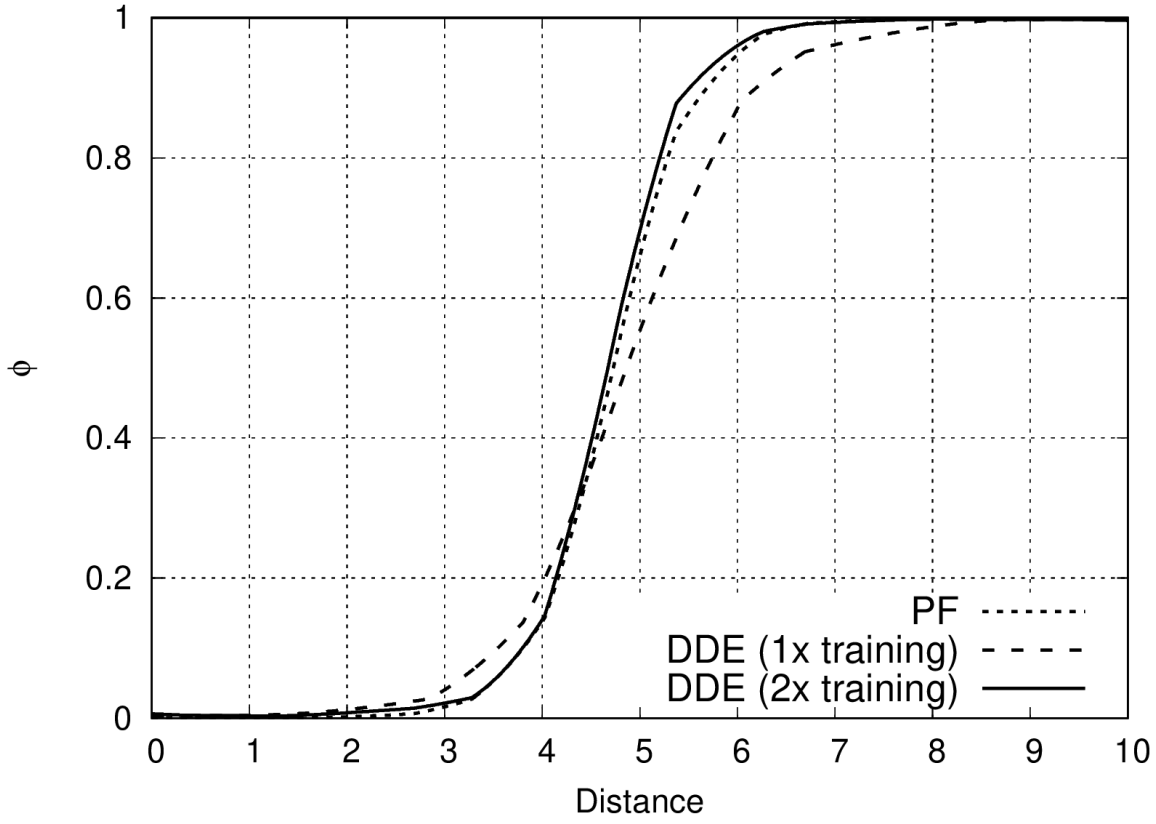

Figure 3.4: Comparing the simulated and the emulated 1-D interface profiles of the conserved order parameter, ϕ, plotted along the black arrow (at t=40) in Fig. 3.1d, as demonstrated by Peichen et al. (2023).

Simulation domain size	Numerical scheme	Total number of image series	Total number of training samples	Time used for generating training data	Training time without TD	Average gain for each epoch with TD	Average loss of validation set (MSE)	DDE prediction time	CPU runtime improvement
32×32	Finite difference	500	15000	0.417 h	13.42 h	44.78%	0.0056	24 s	62.55
32×32	2nd order fourier spectral	500	15000	0.549 h	13.13 h	43.67%	0.0034	22 s	89.84
128×128	Finite difference	400	3000	1.321 h	288.8 h	26.98%	0.0725	79 s	60.20

Table 3.1: Comparison of the computation costs associated with phase-field calculations employing finite difference and Fourier spectral solvers (for numerical scheme, refer to Chen and Shen (1998)) with DDE, as presented by Peichen et al. (2023). Training efficiency improved by tensor decomposition (TD) per epoch is also listed. CPU runtime improvement is calculated as the time used for generating the training data$\times$3600/24. An improvement of 62.55 implies that DDE is around 63 times faster than the finite difference solver.

This acceleration in run time comes at the cost of accuracy which is quantified by a metric that measures the average gain per epoch

$$\text{Average gain} = \frac{t - t_{TD}}{t} \times 100, \tag{3.2}$$

where t refers to the time needed for running one epoch without applying tensor decomposition while t_{TD} corresponds to the runtimes when tensor decomposition is incorporated. In the present context, the epoch refers to the number of times that datasets pass in the forward or in the backward directions per the workflow shown in Fig. 2.1. Corresponding gains in runtime, the MSE, and the overall emulation runtimes are listed in Table 3.1 for computational domains of size, 32×32 and 128×128 grid points. We note that the MSE and the prediction times increase with the computational domain size, since the latter entails handling a larger input variable matrix that increases the processing time. Clearly, a trade-off between prediction accuracy and overall runtime exists in order to predict the microstructural evolution within a reasonable timeframe. Our current technique of accelerating the CNN by deploying tensor decomposition, although novel in the context of emulating microstructural evolution, is essentially motivated from previous findings where more than eight times improvement in the training efficiency could be achieved without any significant decrease in accuracy, as demonstrated by Lebedev et al. (2015). One of the ways by which the performance of the data-driven emulator can be further improved is by lowering the rank of tensor decomposition based on the prediction accuracy threshold. Image downsampling, which is known to be compatible with CRNN approaches, can also be employed for this purpose.

Before concluding, we would like to specify that in our above estimation of average gains and runtime improvements, we have completely ignored the training/learning time that is required for phase-field modeling. Typically, learning phase-field theory

can take over months even when an individual has an appropriate background in materials science, computer programming, and related discipline. Apparently, even a phase-field expert would invest significant time ranging from weeks to months (or even a year) to write and debug the phase-field code. In the current work, we have accounted for DDE training time but excluded the phase-field learning times, thereby, providing a conservative estimate of the acceleration that the former provides. Finally, our study does not discount the importance of phase-field methodology which still remains the best technique for simulating microstructure evolution. Machine learning based surrogate models of microstructure evolution would more often than not completely rely on phase-field models for the generation of training datasets, therefore, mastering the latter which entails acquiring well-rounded knowledge of materials thermodynamics and kinetics, computer programming, and numerical techniques, remains crucial and irreplaceable.

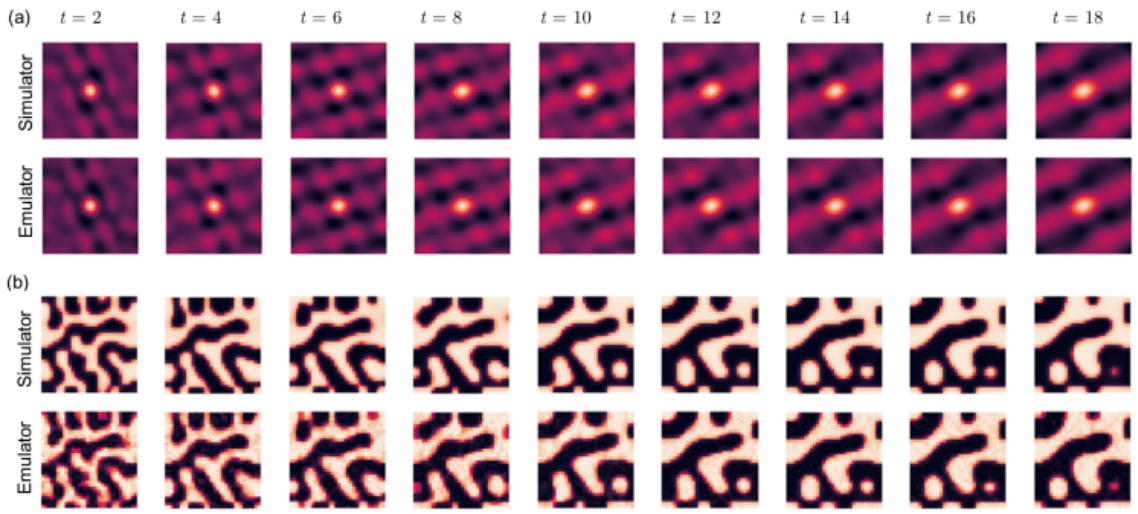

Figure 2.8: Frame-by-frame comparison of microstructure evolution simulated using phase-field method with the corresponding emulation. The top row shows the original sequence of the (a) two-point correlation and (b) auto-correlation functions of the phase-field generated dataset while the emulations are plotted in the bottom row, as demonstrated by Ashif et al. (2023). (repeated from page 33)

Fig. 2.8(a) shows the two-point correlation functions obtained from the phase-field simulator while the corresponding predictions obtained from the emulator are plotted in the second and fourth row. The top row in Fig. 2.8(b) shows the microstructure frames that were not included in the training dataset while the corresponding predictions obtained from the phase-field emulator are shown at the bottom. We note that the predicted microstructures closely resemble the true microstructures.

To demonstrate the advantage of using GPR over other nonlinear regression methods, we compare the corresponding predictions in a representative low-dimensional representation with that of support vector regression with two different kernel functions (a) radial basis kernel and (b) polynomial kernel. The corresponding predictions

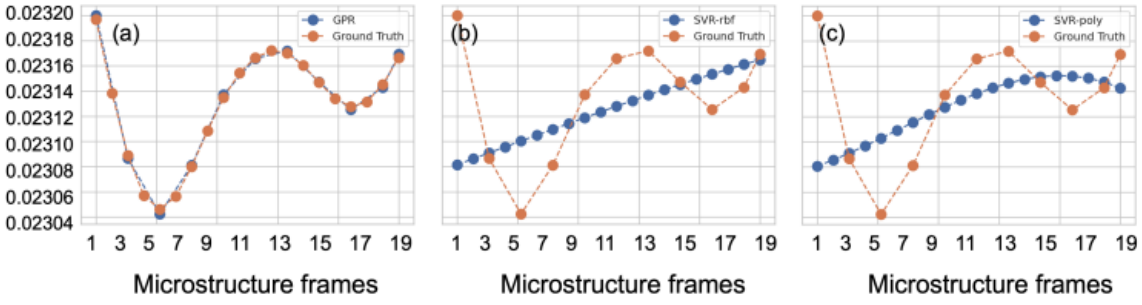

Figure 3.5: Comparison of prediction in a representative low-dimensional space obtained from (a) Gaussian process regression with support vector regression using (b) radial basis function kernel and (c) polynomial kernel, as discussed by Ashif et al. (2023). The y-axis is unlabeled as it may not have physical significance in the low-dimensional space.

are shown in Figure 3.5. From the figure, it is evident that GPR is able to accurately fit the data points whereas support vector regression fails to capture the nonlinear behavior.

The emulated domain size is plotted as a function of time for distinct Δt, as shown in Fig. 3.6(a). The average feature size, as presented by Ankit et al. (2019), was measured by calculating the inverse of the first moment, $k_1(t)$ as noted by Glotzer et al. (1994), which in turn is obtained from the structure factor, $s(k,t)$ given as:

$$k_1(t) \equiv \frac{\sum_k k(t)s(k,t)}{\sum_k s(k,t)} \tag{3.3}$$

See Bray and Emmott (1995) for details on the structure factor and the average feature size. As indicated by these plots, the scaling dynamics are found to nearly overlap with the phase-field results. To further quantify these emulations, we calculate the normalized MSE (see Eq. 2.18) of the predicted microstructures for different values of Δt as shown in Fig. 3.6(b). The average normalized MSE for distinct Δt is equal

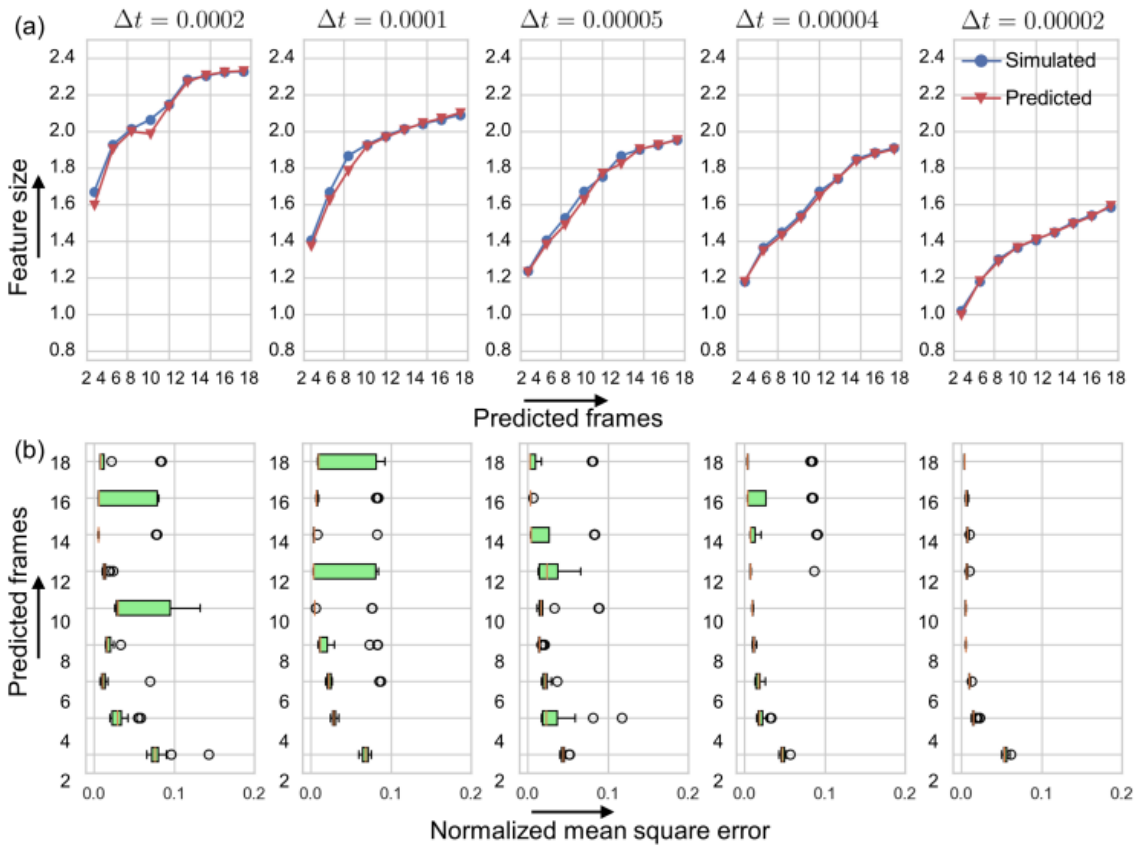

Figure 3.6: (a) Comparison of the microstructure feature size of the predicted microstructure and the simulated microstructure for different values of the time step, Δt. (b) Normalized mean square error for predicted microstructures for different values of time step, Δt, as presented by Ashif et al. (2023).

to 0.0314, 0.0280, 0.0236, 0.0207, and 0.0128.

At this point, some important observations are in order; first, the reconstruction of the microstructures obtained from our methodology is statistically accurate (with average normalized $MSE = 6.78 \times 10^{-7}$) as evident from Fig. 2.8(a). Secondly, we notice that the proposed method is able to predict the microstructure evolution both in the transient stages initially where the phase separating domain size increases rapidly as well as later during the slow coarsening stage with a sufficiently high accu-

racy (see Fig. 2.8(b)). It is noteworthy to emphasize that predicting microstructural evolution during the transient stages has proven to be challenging for recently published algorithms, as highlighted by Yang et al. (2021); Zapiain et al. (2021). This difficulty arises from the rapid evolution of microstructures in the initial stages, making it challenging to capture the evolving microstructural features. In contrast, the tensor-based phase field emulator, developed in this research, preserves the spatio-temporal relationships even in the low-dimensions, therefore, we are able to accurately predict the microstructural evolution and recover the corresponding scaling dynamics. These results show that the proposed approach is not only statistically accurate, but is also able to capture the complex microstructural features. Thirdly, we notice that the normalized MSE of the predicted microstructures decreases as the spinodal decomposition proceeds towards the late-stage coarsening regime given that the feature complexity is higher when the average phase separated domain size is small. We also observe that the average normalized MSEs decrease as the timestep width, Δt, for the training data decreases, which is owing to the larger interpolations that are performed by the emulator at larger Δt, causing a relative loss in accuracy.

The phase-field emulator proposed in this work involves four steps as outlined in Fig. 2.7. The first three steps comprise the training phase of the emulator which includes extraction of the auto-correlation functions, tensor decomposition, and GPR training. Testing phase involves prediction using GPR and hybrid input-output algorithm for phase retrieval. For a microstructure with S grids evolving for T time periods and a tensor rank of R, the computation cost during the training phase for each of the steps are $\mathcal{O}(S)$, $\mathcal{O}(STR)$, and $\mathcal{O}(RT^3)$ respectively. Note that the computational complexity for tensor decomposition is determined for CP decomposition with alternating least squares Ma and Solomonik (2021). For the testing phase, the computational complexity for a single microstructure image is $\mathcal{O}(R)$ for GPR predic-

Domain	Training time	Testing time	PFS time	PFS to Emulator
32×32	0.24	0.12	0.19	1.58
256×256	0.78	12.98	68	5.24
512×512	3.18	64.5	588.97	9.13
1024×1024	198.88	289.58	5057.3	17.46

Table 3.2: Computational run times of emulator and phase-field simulations (in seconds), as presented by Ashif et al. (2023).

tion and $\mathcal{O}(MS \log S)$ for the hybrid input-output algorithm considering M iterations. If the rank R of CP decomposition is fixed, then the computational complexity during the testing phase scales as $\mathcal{O}(MS \log S)$, making phase retrieval as the most resource intensive step. Table 3.2 shows the breakdown of computational cost for each of the steps. All computations presented in this work were performed on an Intel(R) Core(T) i9-10900K CPU with 32 GB of RAM without any GPU acceleration. The run time for obtaining the microstructure sequence from phase-field simulation required approximately 0.19 sec for microstructure of size 32×32. For every microstructure sequence generated from the phase-field simulation, we used $n/2 + 1$ of the frames to construct the training dataset. The distribution for each of the steps involved in the phase-field emulator are as follows: In the training phase, the calculation of auto-correlation function consumed 44.4%, CP decomposition consumed 41.8%, while training the GPR required 13.8% of the total training time. In the prediction phase, GPR predictions accounted for merely 0.4% and the microstructure reconstruction

from GPR predictions consumed more than 99.6% of the total run time. We also compare the computational costs of microstructure prediction with an increase in domain size. Table 3.2 shows the time (in seconds) for generating one microstructure in the sequence for 32×32, 256×256, 512×512, and 1024×1024. The last column shows the ratio of time taken by the phase-field simulator for simulating one microstructure to the time taken by phase-field emulator for predicting one microstructure. We note that the proposed emulator is at least 5 times faster as compared to the phase-field simulator. Interestingly, the phase-field emulator performs faster as the domain size increases. In fact, for the largest domain size of 1024×1024, the phase-field emulator is more than 17 times faster as compared to the phase-field simulations.

We also note some limitations of the proposed approach. In the current implementation, we first obtain a tensor decomposition followed by a separate GPR for each rank one tensors. This two-step approach may be computationally intensive when complex microstructural features are involved that require higher rank CP decomposition. In our future works, we aim to develop nonlinear tensor regression methods that would not require any decomposition. A second limitation of the approach originates from the CP decomposition itself. CP decomposition results in rank one tensor across each of the temporal and spatial modes. Since we only predict across the temporal mode, the information across the spatial mode is smeared across the temporal mode. As a result, we notice that the reconstruction errors are higher in the initial stages of spinodal decomposition when the microstructures are rapidly evolving (see Fig. 2.8(b)). Finally, we also notice some limitations with the two point correlations. For two phase microstructures (as considered in this work), it was possible to handle two-point correlations since we can summarize all the information with just either auto-correlation or cross-correlation. However, for microstructure with multiple phases, two-point correlations may be cumbersome and may require higher-order

correlation functions.

3.2.1 Limitations and Future Work

Data-driven approach is a promising method, however there are still some limitations:

1. Demanding substantial human resources for the verification of training images: In our case, we employ the established phase-field method to automatically generate a series of microstructure evolution images. This approach is advantageous as it eliminates the need for manual checks on the training database. However, for future work, the inclusion of real-world images in the training database will necessitate significant human resources to ensure the quality and validity of each image data.

2. Intensive computational resource is needed for 3D emulations: As the dimension increase from 2D to 3D, the computational power required increase substantially. Take training time without TD for 32×32 in Table. 3.1, the training time for $32 \times 32 \times 32$ is at least equal to $13.42 \times 32 = 429.44$ hours. One of the future work plans to combine the two DDE methods introduced here to solve the second issue. The objective is to employ these two methods for the efficient emulation of similar 3D microstructure evolution shown in fig. 3.7 and fig. 3.8.

In this study, we have introduced two data-driven approaches. The CRNN method excels in predicting future microstructure images. On the other hand, GPR-based (Gaussian process regression) approach demonstrates high efficiency in predicting arbitrary interpolated time steps during the evolution of microstructures and this high efficiency come from the implementation of the low representation of image data. While CRNN can achieve impressive extrapolation predictions, it comes at the expense of intensive computational resources on training. Figure 3.8 illustrates a 3D

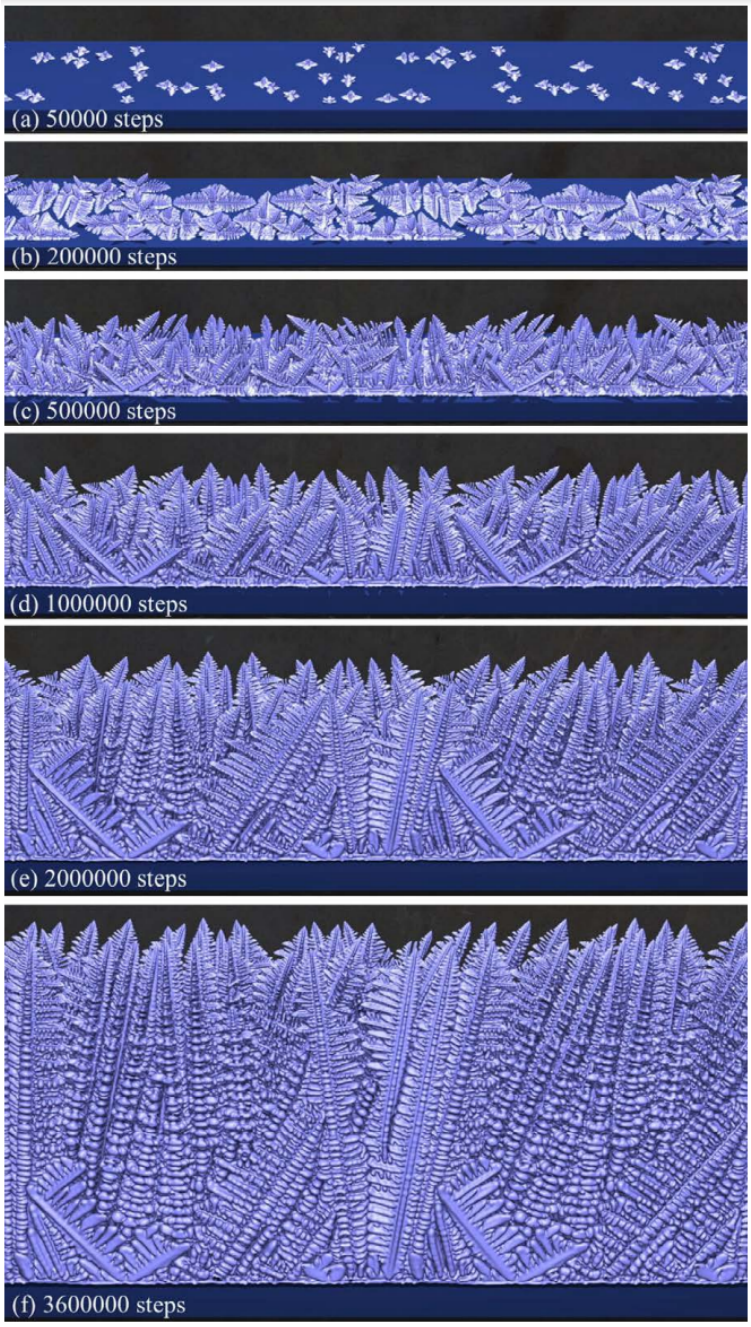

Figure 3.7: Time slices showing competitive dendritic growth during directional solidification of Al–Si alloy. This phase-field simulation was performed in a computational domain of $3.072mm \times 0.768mm \times 3.072mm$ ($4096 \times 1024 \times 4096$ lattices) using GPU supercomputer TSUBAME2.0 at Tokyo Institute of Technology. Reprinted with permission from Takaki. (2014). Copyright 2014 by J-Stage.

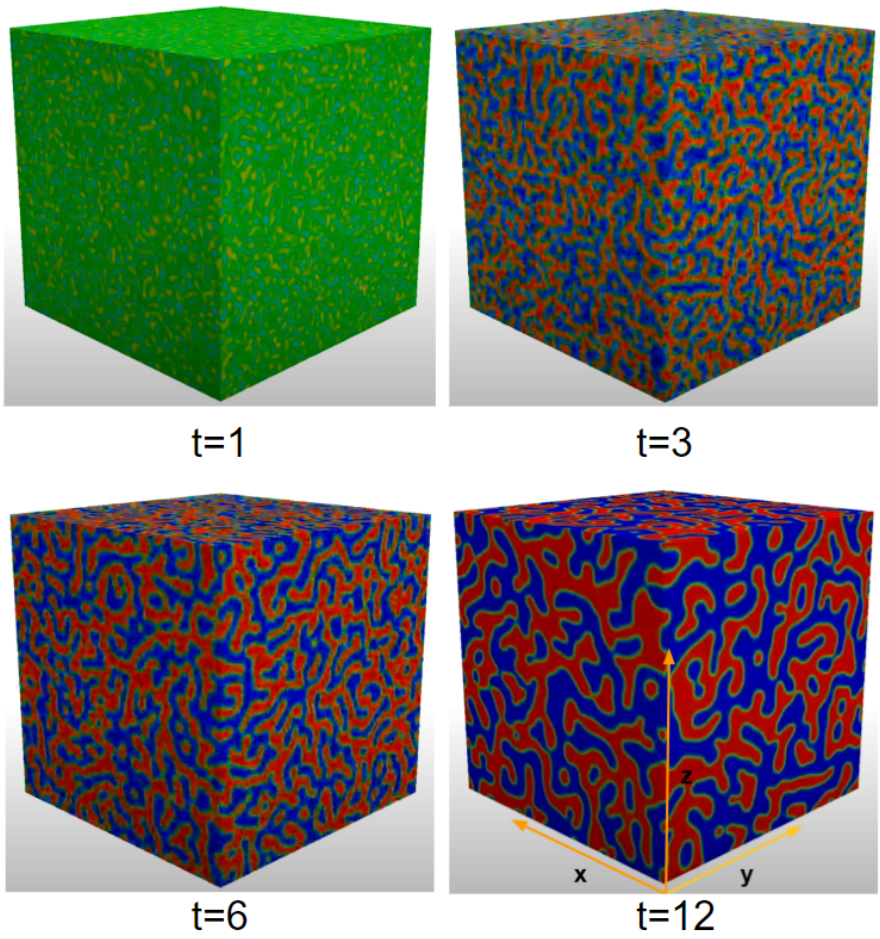

Figure 3.8: 3D spinodal decomposition microstructure.

spinodal decomposition microstructure, where the 3D image data can be conceptualized as a series of 2D images in the x-y plane arranged vertically in the z direction.

The computational challenges associated with using CRNN directly for 3D extrapolation predictions have prevented anyone from achieving 3D microstructure predictions thus far. To address this challenge, we adopt a strategy where CRNN is used for 2D extrapolation predictions on a subset of layers within a 3D image. Simultaneously, the method (GPR) described in Subsec 2.2.2 is employed for interpolation predictions along the vertical direction (z) to complement the remaining 2D layers in the same 3D image.

While the foregoing discussions have employed phase-field simulations of spinodal decomposition, the proposed approach is generalizable to other microstructural evolution scenarios, such as coarsening of precipitates and dendritic growth. The GPR approach may be attractive for another two applications. The first application involves obtaining the microstructure evolution at arbitrarily small-time scales via phase-field methods. Traditionally, the computational complexity limits the temporal resolution of phase-field simulations. With the proposed phase-field emulator, it is possible to obtain the microstructure evolution at a smaller time step, i.e., higher temporal resolution using just the microstructure evolution obtained at a larger timescale. Therefore, the emulator would preclude the need for running multiple simulations whenever there is a change in the time step parameter. The second application will be in physical experiments, such as those that were performed under microgravity by NASA, where physical restrictions and resource constraints limit the number of experiments that could be performed, as noted by Snyder et al. (2001). Similar limitations are encountered when observing in-situ microstructure evolution using costly high-fidelity measurement techniques, such as transmission electron microscopy, as noted by Le Ferrand et al. (2019). These techniques often impose constraints on the frequency of recording microstructures over time. Under such circumstances, the proposed emulator could provide an opportunity to observe the microstructure evolution at intermediate time steps that could not recorded by the measurement system.

Chapter 4

CONCLUSIONS

In conclusion, we have first introduced a novel CRNN-based approach for emulating microstructural evolution that accompanies spinodal decomposition. This data-driven approach, which leverages phase-field generated microstructure datasets as input, combines tensor decomposition and deep learning to predict the morphological evolution efficiently. Based on a systematic study, we found that a fully trained DDE emulator can predict the microstructural evolution 60 times faster when compared to the phase-field method.

The convolution and recurrent neural network (CRNN) follows an image-based, nonparametric algorithm, which is capable of extracting microstructural features and their evolution pattern from phase-field simulated sequences. The proposed CRNN is material as well as process agnostic, and therefore could potentially be used for predicting microstructural evolution in distinct scenarios as long as the training data is available. Unlike previous studies by Zapiain et al. (2021); Herman et al. (2020); Yang et al. (2021), the present DDE model, in addition to reproducing the correct scaling dynamics and the morphological evolution, conserves the characteristic diffuse nature of the interfaces. However, in order to reproduce the smooth hyperbolic tangent-like interface profiles which the phase-field predicts, the training volume needs to be increased. The time required for a trained DDE to predict microstructural evolution typically spans a minute, which is minuscule when compared to phase-field simulation run times.

While our work is not the first one to develop a data-driven simulator for phase field methods, our novel implementation of tensor decomposition into CNN instead

of using low-dimensional projections of two-point correlation functions as used by Zapiain et al. (2021); Herman et al. (2020), accelerates the neural network without significant accuracy losses. In closing, we would like to emphasize that the development of a microstructure emulation method such as the one reported here is not aimed at replacing the phase field models. Rather it is meant to aid in scenarios where obtaining the solution of partial differential equations is non-trivial, or in those cases where the complexities related to multi-physics make the formulation of phase field models, difficult. In numerous scenarios, phase-field models could be the only means for generating the training data for DDE. Apparently, the present emulation technique heavily relies on the development of phase-field models without which the former cannot be applied.

CRNN can achieve impressive extrapolation predictions, it comes at the expense of intensive computational resources on training. Unlike the CRNN-based approach, we secondly introduced the GPR-based (Gaussian process regression) approach to efficiently predict arbitrary interpolated time steps during the evolution of microstructures and this high efficiency come from the implementation of the low representation of image data. Traditionally, the computational complexity of phase-field simulations limit the smallest time-scales that are tractable. In light of this, the proposed approach provides an alternative to obtain phase-field simulations at smaller time scales without the need to running costly simulations. By casting microstructure evolution as a tensor, the proposed approach preserves the spatio-temporal relationships in the low-dimensional representations obtained via tensor decomposition. Overall, the proposed-phase field emulator is able to predict the microstructures with average NMSE of 6.78×10^{-7}. We also highlight that the phase-field emulator is material agnostic and may be applied to a wide array of evolving microstructures.

It is also equally important to note that both of our data-driven approaches

(DDEs), unlike numerical schemes, does not involve solving any partial differential equations. It is agnostic of the complex multi-physics associated with phase transformations as the morphological evolution that is emulated is entirely based on image processing and machine learning. Therefore, the full potential of DDE lies in scenarios where phase-field models that could possibly encapsulate all the relevant physics are yet to be developed, for e.g., additive manufacturing. In such cases, training datasets for DDE can be generated using in-situ experiments.

Finally, we have also discussed the potential future work. Thus far, no one has successfully undertaken 3D microstructure predictions, primarily because the direct application of CRNN for 3D extrapolation predictions is computationally demanding. Nevertheless, through the amalgamation of our two data-driven approaches (DDEs), there is a substantial potential to overcome existing challenges and attain effective and efficient predictions for the evolution of 3D microstructures.

REFERENCES

Ankit, K., B. Derby, R. Raghavan, A. Misra, and M. J. Demkowicz (2019). 3-D phase-field simulations of self-organized composite morphologies in physical vapor deposited phase-separating binary alloys. *J. Appl. Phys. 126*(7).

Ashif, I., W. Peichen, S. Ali, and A. Kumar (2023). Emulating the evolution of phase separating microstructures using low-dimensional tensor decomposition and nonlinear regression. *MRS Bulletin 48*, 602–613.

Ashif, I., W. Peichen, and A. Kumar (2023). mulating the evolution of phase separating microstructures using low-dimensional tensor decomposition and nonlinear regression. *MRS Bulletin 48*, 602–613.

Behler, J. and M. Parrinello (2007). Generalized neural-network representation of high-dimensional potential-energy surfaces. *Physical Review Letters 98*(146401).

Boettinger, W. J., J. A. Warren, C. Beckermann, and A. Karma (2002). Phase-field simulation of solidification. *Annual Review of Materials Research 32*(1), 163–194.

Bray, A. J. and C. L. Emmott (1995). Lifshitz-Slyozov scaling for late-stage coarsening with an order-parameter-dependent mobility. *Phys. Rev. B 52*(2), R685.

Bruder, F. and R. Brenn (1992). Spinodal decomposition in thin films of a polymer blend. *Physical Review Letters 69*(4), 624.

Cahn, J. W. (1961). On spinodal decomposition. *Acta Metallurgica et Materialia 9*(9), 795 – 801.

Carroll, J. and J. Chang (1970). Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika 35*, 283–319.

Chen, L.-Q. (2002). Phase-field models for microstructure evolution. *Annual Review of Materials Research 32*(1), 113–140.

Chen, L.-Q. and J. Shen (1998). Applications of semi-implicit fourier-spectral method to phase field equations. *Computer Physics Communications 108*(2-3), 147–158.

Cheng, K., W. Feng, C. Wang, and S. Wise (2019). An energy stable fourth order finite difference scheme for the Cahn–Hilliard equation. *Journal of Computational and Applied Mathematics 362*, 574–595.

Desai, S. and R. Dingreville (2022). Learning time-dependent deposition protocols to design thin films via genetic algorithms. *Materials & Design 219*, 110815.

DeWitt, S., S. Rudraraju, D. Montiel, W. B. Andrews, and K. Thornton (2020). PRISMS-PF: A general framework for phase-field modeling with a matrix-free finite element method. *NPJ Computational Materials 6*(1), 1–12.

Elhaik, E. (2022). Principal component analyses (pca)-based findings in population genetic studies are highly biased and must be reevaluated. *Scientific Reports 12*, 14683.

Fienup, J. (1982). Phase retrieval algorithms: a comparison. *Applied Optics 21*, 2758–2769.

Frisch, H. and F. Stillinger (1963). Contribution to the statistical geometric basis of radiation scattering. *The Journal of Chemical Physics 38*, 2200–2207.

George, W. L. and J. A. Warren (2002). A parallel 3d dendritic growth simulator using the phase-field method. *Journal of Computational Physics 177*(2), 264–283.

Gerchberg, R. and W. Saxton (1972a). A practical algorithm for the determination of the phase from image and diffraction plane pictures. *Optik 35*, 283–319.

Gerchberg, R. and W. Saxton (1972b). A practical algorithm for the determination ofthe phase from image and diffraction plane pictures. *Optik 35*, 237–246.

Glotzer, S. (1995). Computer simulations of spinodal decomposition in polymer blends. *Annual Reviews of Computational Physics II*, 1–46.

Glotzer, S. C., M. F. Gyure, F. Sciortino, A. Coniglio, and H. E. Stanley (1994, jan). Pinning in phase-separating systems. *Phys. Rev. E 49*(1), 247–258.

Greenwood, M., K. Shampur, N. Ofori-Opoku, T. Pinomaa, L. Wang, S. Gurevich, and N. Provatas (2018). Quantitative 3D phase field modelling of solidification using next-generation adaptive mesh refinement. *Computational Materials Science 142*, 153–171.

Herman, E., J. Stewart, and R. Dingreville (2020). A data-driven surrogate model to rapidly predict microstructure morphology during physical vapor deposition. *Applied Mathematical Modelling 88*.

Hu, C., S. Martin, and R. Dingreville (2022). Accelerating phase-field predictions via recurrent neural networks learning the microstructure evolution in latent space. *Computer Methods in Applied Mechanics and Engineering 397*, 115128.

Iquebal, A. S., P. Wu, A. Sarfraz, and K. Ankit (2023). Emulating the evolution of phase separating microstructures using low-dimensional tensor decomposition and nonlinear regression. *MRS Bulletin*, 1–12.

Jokisaari, A., P. Voorhees, J. Guyer, J. Warren, and O. Heinonen (2017). Benchmark problems for numerical implementations of phase field models. *Computational Materials Science 126*, 139–151.

Kalidindi, S. (2015). Statistical quantification of material structure. *Hierarchical Materials Informatics 16*, 75–110.

Kautz, E. J. (2021). Predicting material microstructure evolution via data-driven machine learning. *Patterns*, 100285.

Kim, Y., Y. Jernite, D. Sontag, and A. M. Rush. (2016). Character-aware neural language models. *Thirtieth AAAI Conference on Artificial Intelligence*.

Kolda, T. and B. Bader (2009). Tensor decompositions and applications. *SIAM review 51*, 455–500.

Kossaifi, J., Y. Panagakis, A. Anandkumar, and M. Pantic (2019). Tensorly: Tensor learning in python. *Journal of Machine Learning Research 20*, 1–6.

Langer, J. (1971). Theory of spinodal decomposition in alloys. *Annals of Physics 65*(1), 53–86.

Le Ferrand, H., M. Duchamp, B. Gabryelczyk, H. Cai, and A. Miserez (2019). Time-resolved observations of liquid–liquid phase separation at the nanoscale using in situ liquid transmission electron microscopy. *Journal of the American Chemical Society 141*, 7202–7210.

Lebedev, V., Y. Ganin, M. Rakhuba, I. Oseledets, and V. Lempitsky. (2015). Speeding-up convolutional neural networks using fine-tuned cp-decomposition. *The International Conference on Learning Representations*.

Liu, J. (2012). Controlling the aliasing by zero-padding in the digital calculation of the scalar diffraction. *JOSA A29*, 1956–1964.

Ma, L. and E. Solomonik (2021). Fast and accurate randomized algorithms for low-rank tensor decompositions. *NeurIPS 2021 Conference*.

Merriman, B., J. Bence, and S. Osher (1992). *Diffusion generated motion by mean curvature*. Department of Mathematics, University of California, Los Angeles.

Merriman, B., J. K. Bence, and S. J. Osher (1994). Motion of multiple junctions: A level set approach. *Journal of Computational Physics 112*(2), 334–363.

Millán, E. N., C. A. Ruestes, N. Wolovick, and E. M. Bringa (2017). Boosting materials science simulations by high performance computing. *Mecánica Computacional 35*(10), 467–482.

Miller, M., J. Hyde, M. Hetherington, A. Cerezo, G. Smith, and C. Elliott (1995). Spinodal decomposition in Fe–Cr alloys: Experimental study at the atomic level and comparison with computer models–I. Introduction and Methodology. *Acta Metallurgica et Materialia 43*(9), 3385–3401.

Moelans, N., B. Blanpain, and P. Wollants (2008). An introduction to phase-field modeling of microstructure evolution. *Calphad 32*(2), 268–294.

Mukherjee, A., R. Mukherjee, K. Ankit, A. Bhattacharya, and B. Nestler (2016, Mar). Influence of substrate interaction and confinement on electric-field-induced transition in symmetric block-copolymer thin films. *Physical Review E 93*, 032504.

Oommen, V., K. Shukla, S. Goswami, R. Dingreville, and G. Karniadakis (2022). Learning two-phase microstructure evolution using neural operators and autoencoder architectures. *NPJ Computational Materials 8*(1), 190.

Oseledets, I. (2011). Tensor-train decomposition. *SIAM Journal on Scientific Computing 33*, 5.

O'Shea, K. and R. Nash (2015). An introduction to convolutional neural networks.

Peichen, W., I. Ashif, and A. Kumar (2023). Emulating microstructural evolution during spinodal decomposition using a tensor decomposed convolutional and recurrent neural network. *Computational Materials Science 224*, 112187.

Plapp, M. and A. Karma (2000). Multiscale random-walk algorithm for simulating interfacial pattern formation. *Physical Review Letters 84*(8), 1740.

Provatas, N. and K. Elder (2010, Oct). *Phase-Field Methods in Materials Science and Engineering*. Wiley.

Provatas, N., N. Goldenfeld, and J. Dantzig (1998). Efficient computation of dendritic microstructures using adaptive mesh refinement. *Phys. Rev. Lett. 80*, 3308–3311.

Raghavan, R., A. Mukherjee, and K. Ankit (2020). Nanostructural evolution in vapor deposited phase-separating binary alloy films of non-equimolar compositions: Insights from a 3d phase-field approach. *Journal of Applied Physics 128*(17), 175303.

Richard, H. (1970). Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. *Computer Science 38*, 2200–2207.

Rundman, K. and J. Hilliard (1967). Early stages of spinodal decomposition in an aluminum-zinc alloy. *Acta Metallurgica 15*(6), 1025–1033.

Shimokawabe, T., T. Aoki, T. Takaki, T. Endo, A. Yamanaka, N. Maruyama, A. Nukada, and S. Matsuoka (2011). Peta-scale phase-field simulation for dendritic solidification on the tsubame 2.0 supercomputer. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–11.

Singer-Loginova, I. and H. Singer (2008). The phase field technique for modeling multiphase materials. *Reports on Progress in Physics 71*(10), 106501.

Snyder, V., J. Alkemper, and P. Voorhees (2001). Transient ostwald ripening and the disagreement between steady-state coarsening theory and experiment. *Acta materialia 49*, 699–709.

Takaki., T. (2014). Phase field modeling and simulations of dendrite growth. *ISIJ International 54*(2).

Toral, R., A. Chakrabarti, and J. D. Gunton (1988, May). Numerical study of the cahn-hilliard equation in three dimensions. *Physical Review Letters 60*, 2311–2314.

Torquato, S. (2002). Random heterogeneous materials: Microstructure and macroscopic properties. *Interdisciplinary Applied Mathematics 16*, 237–246.

Tucker, L. R. (1966). Some mathematical notes on three-mode factor analysis. *Psychometrika 31*, 279–311.

Wang, H., L. Zhang, J. Han, and W. E. (2018). Deepmd-kit: A deep learning package for many-body potential energy representation and molecular dynamics. *Computer Physics Communications 228*.

Wu, L., L. Liu, Y. Wang, Z. Zhai, H. Zhuang, D. Krishnaraju, Q. Wang, and H. Jiang. (2020). A machine learning-based method to design modular metamaterials. *Extreme Mechanics Letters 36*(100657).

Yang, K., Y. Cao, Y. Zhang, S. Fan, M. Tang, D. Aberg, B. Sadigh, and F. Zhou. (2021). Self-supervised learning and prediction of microstructure evolution with convolutional recurrent neural networks. *Patterns Cell press*.

Zaitzeff, A., S. Esedoḡlu, and K. Garikipati (2020). Second order threshold dynamics schemes for two phase motion by mean curvature. *Journal of Computational Physics 410*, 109404.

Zapiain, D., J. Stewart, and R. Dingreville (2021). Accelerating phase-field-based microstructure evolution predictions via surrogate models trained by machine learning methods. *NPJ Computational Materials 7*(1), 1–11.

APPENDIX A

DERIVATIONS FOR TUCKER DECOMPOSITION

Kernel tensor K can be approximated by using the Tucker decomposition:

$$K(i,j,s,t) = \sum_{r_1=1}^{R_1=1} \sum_{r_2=1}^{R_2=2} \sum_{r_3=1}^{R_3=3} \sum_{r_4=1}^{R_4=1} \Sigma_{r_1 r_2 r_3 r_4} K^x_{r_1}(i) K^y_{r_2}(j) K^s_{r_3}(s) K^t_{r_4}(t) \tag{A.1}$$

The Tucker decomposition possesses a beneficial characteristic in that it is not obligatory to decompose along all the axes or modes. Given that the spatial dimensions are already relatively small (3×3), decomposing those dimensions may not be particularly meaningful. Instead, it can be used for decomposition along the input and output channels (a mode-2 decomposition):

$$K(i,j,s,t) = \sum_{r_3=1}^{R_3=3} \sum_{r_4=1}^{R_4=1} \Sigma_{ij r_3 r_4}(j) K^s_{r_3}(s) K^t_{r_4}(t) \tag{A.2}$$

Plugging this into the formula for the convolutional layer output form above:

$$\begin{aligned} V(x,y,t) &= \sum_i \sum_j \sum_s \sum_{r_3=1}^{R_3=3} \sum_{r_4=1}^{R_4=1} \sigma_{(i)(j) r_3 r_4} K^s_{r_3}(s) K^t_{r_4}(t) X(x-i, y-i, s) \\ &= \sum_i \sum_j \sum_{r_3=1}^{R_3=3} \sum_{r_4=1}^{R_4=1} K^t_{r_4}(t) \sigma_{(i)(j) r_3 r_4} \sum_s K^s_{r_3}(s) X(x-i, y-i, s) \end{aligned} \tag{A.3}$$

By doing this, it gives us the following result:

1. Point-wise convolution $K^s_{r_3}(s)$ with for reducing the number of channels from S to $K^s_{r_3}(s)$.

2. Regular (not separable) convolution with $\sigma_{(i)(j) r_3 r_4}$.

Contrary to the original layer with S input channels and T output channels, this convolution now has R_3 input channels and R_3 output channels. The speed gain is achieved when these ranks are smaller than S and T.

APPENDIX B

PERMISSION STATEMENT ONE

I have permission from all my co-authors to use the work listed below in my dissertation:

Wu, P*, Iquebal, A, & Ankit, K (2023). Emulating microstructural evolution during spinodal decomposition using a tensor decomposed convolutional and recurrent neural network. *Computational Materials Science* 224: 112187

Iquebal, A, Wu, P*, & Ankit, K (2023). Emulating the evolution of phase separating microstructures using low-dimensional tensor decomposition and nonlinear regression. *MRS Bulletin* 48: 602-613

Wu, P*, Farmer, W, & Ankit, K (2023). A Novel Data-Driven Emulator for Predicting Electromigration-Mediated Damage in Poly-crystalline Interconnects. *Journal of Electronic Materials* 52: 2746-2761

Raghavan, R., Wu, P*., & Ankit, K. (2022). Phase-field modeling of nanostructural evolution in physical vapor deposited phase-separating ternary alloy films. *Modelling and Simulation in Materials Science and Engineering* 30(8), 084004

Glicksman, M. E., Ankit, K., & Wu, P*. (2022). Capillary effects on curved solid–liquid interfaces: An overview. *Journal of Crystal Growth* 126871

Glicksman, M. E., Wu, P*., & Ankit, K. (2022). Periodic Grain Boundary Grooves: Analytic Model, Formation Energies, and Phase-Field Comparison. *Journal of Phase Equilibria and Diffusion* 1-20

Glicksman, M. E., Wu, P*., & Ankit, K. (2021). Surface Laplacian of interfacial thermochemical potential: its role in solid-liquid pattern formation. *Nature Microgravity* 7(1), 41