Machine Learning Approaches in Crystal Plasticity

by

Olga Ibragimova

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2022

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Dr. Jeong Whan Yoon Professor, Dept. of Mechanincal Engineering, Korea Advanced Institute of Science and Technology
Supervisors:	Dr. Kaan Inal NSERC/General Motors Industrial Research Chair Professor, University of Waterloo
	Dr. Julie Lévesque R&D Project Manager, Québec Metallurgy Centre Adjunct Professor, University of Waterloo
Internal Member:	Dr. Cliff Butcher Assistant Professor, Dept. of Mechanical and Mechatronics Engineering, University of Waterloo
Internal Member:	Dr. Adrian Gerlich Assistant Professor, Dept. of Mechanical and Mechatronics Engineering, University of Waterloo
Internal-External Member:	Dr. Alexander Wong Professor, Dept. of System Design Engineering, University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

The following co-authors have contributed to the current work as outlined below:

Professor Kaan Inal supervised this PhD thesis and assisted with funding acquisition.

Dr. Julie Lévesque (R& D Project Manager, Québec Metallurgy Centre and Adjunct Professor at the University of Waterloo) co-supervised this PhD thesis and assisted with funding acquisition.

Dr. Abhijit P. Brahme (Research Associate, University of Waterloo) converted EBSD measurements to develop synthetic 2D microstructures for simulations, provided mentorship and guidance on the analysis of material data, and assisted with editing parts of the work.

Dr. Waqas Muhammad (Research Associate, University of Waterloo) provided the raw EBSD data, provided mentorship and guidance on the analysis of material data, and assisted with editing parts of the work.

Dr. Dan Connolly (Research Associate, University of Waterloo) provided valuable discussions during CNN model development and results analysis.

The balance of work is my own.

Abstract

The continued advancements in material development and design require understanding the relationships between microstructure and flow behaviour. Crystal plasticity (CP) is a high-fidelity computational method that helps unravel these relationships and assist in the development of high-performance materials. CP can capture the material behavior subjected to any applied loading as well as the local stress and strain partitioning and localisation in a given microstructure. The high-fidelity of the crystal plasticity models comes at the cost of computational demand. This research addresses the significant computational demand of CP simulations and uses machine learning methods to achieve rapid and accurate predictions of material plastic behaviour. This research project uses two different crystal plasticity models: the CP model under a fully constrained Taylor assumption and the crystal plasticity finite element model. This thesis covers two different machine learning applications to high-fidelity predictions of model behaviour.

The first application presents a machine learning- and crystal plasticity-based framework to predict stress-strain behaviour and texture evolution for a wide variety of materials within the face-centred cubic family (FCC). First, the process of the framework design is described in detail. The proposed framework incorporates an ensemble of artificial neural networks (ANN) and a crystal-plasticity based algorithm. Next, the dataset constituent of crystal plasticity simulations was collected. The dataset consisting of examples of monotonic deformation cases, was prepared for training using mathematical transformations, and finally used to train ANNs used in the framework. Then, the framework was demonstrated to predict full stress-strain and texture evolution of different FCC single crystals under uniaxial tension, compression, simple shear, equibiaxial tension, tension- compression-tension, compression-tension-compression, and, finally, for some arbitrary non-monotonic loading cases. The proposed framework predicts the stress-strain response and texture evolution with high accuracy. The results demonstrated in this research show that the proposed machine learning- and crystal plasticity-based framework exhibits a tremendous computational advantage over the conventional crystal plasticity model. Finally, one of the most important contributions of this work is to show the framework's feasibility. The work demonstrates that machine learning methods can help predict complex strain paths without training machine learning models on the extremely large set of possible non-monotonic loading scenarios. This part of the thesis presents a macro-level model that allows predictions for single and polycrystals.

The second part of this research project presents a micro-level type of model and utilizes convolutional neural networks (CNNs) in conjunction with the crystal plasticity finite element method (CPFEM). The inputs to the CNN model are material hardening param-

eters (initial hardness and initial hardening modulus), a global tensile strain value, and microstructure with varying number of grains, grain size, grain morphology and texture. This input selection allows performing simulations for a wide range of materials, as defined by microstructure and flow curves. The outputs of the CNN are the local stress and strain values. The proposed framework involves the following stages: feature engineering, generation of synthetic microstructures, CPFEM simulations, data extraction and preprocessing, CNN design and selection, CNN training, and validation of the trained network. The trained CNN was successfully demonstrated to predict local stress and strain evolution for the completely new dataset (test set) containing synthesised microstructures. The test set predictions were evaluated, and the median-, highest-, and lowest-error predictions were presented and discussed. Overall, the CNN demonstrated excellent agreement with CPFEM simulations, thus validating its accuracy. Then, the CNN was applied to predict the stress and strain evolution for AA5754 and AA6061 microstructures obtained using electron backscatter diffraction. These two microstructures were entirely new for the CNN and displayed size and grain morphology different from the synthesised microstructures. For both microstructures, the obtained stress and strain evolution predictions demonstrated excellent agreement with CPFEM simulations, thus confirming the flexibility of the trained CNN model. Then, the framework was extended to predict strain localisation and was evaluated on an AA6061 microstructure. The results demonstrate a clear computational advantage of CNN without losing accuracy.

Acknowledgements

Firstly, I would like to express my sincere gratitude to my supervisor, Professor Kaan Inal, who made this thesis and all this completed work possible. I sincerely thank him for his expertise, excellent supervision, and competent and encouraging leadership. Apart from his academic support, I sincerely thank him for his kindness, motivation, patience, understanding, generosity, and moral support. I would like to express my appreciation for the opportunity to take part as a visiting scientist at General Motors – Canadian Technical Centre, Oshawa, and for allowing me to contribute to projects unrelated to my thesis. The experience that I obtained during my program is priceless. Thank you with all my heart.

I would also express my gratitude to my co-supervisor, Professor Julie Lévesque, who made this thesis possible. I would like to express my sincere appreciation for all the support provided during the course of my PhD program. Thank you very much.

I am deeply thankful for the support of Dr. Abhijit Brahme for his mentorship, technical expertise, and helpful discussions. I would like to thank him for providing me with technical knowledge about material science and programming. I would not be able to do it without you. Thank you from the bottom of my heart.

Special thanks go out to Dr. Waqas Muhammad for his guidance and support. Apart from technical help, I would like to express my most profound appreciation for his moral support, friendship, and positive outlook on life. Thank you a lot.

I would also like to thank CMRG and all my office mates at the University of Waterloo for the good times, helpful discussions, and support throughout the entire process.

And of course, I am deeply grateful to my parents, family, and friends for all the support you provided me. I am grateful to have you on my side.

Dedication

I would like to dedicate this thesis to my beloved parents, Galina and Rustem, and my beloved grandparents, Ludmila, Vitaly, Ganza, and Makmun. I was blessed to have you bring me up, and I love you with all my heart.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Machine learning (ML) is a data-driven approach and is a powerful function approximation tool. The function approximation process is called training, and it is employed on prepared datasets. The advantage is that the ML algorithm does not need to be explicitly programmed for specific tasks since the training approach is similar regardless of the application. ML methods' extensive functionality and practicality are widely explored in various fields of business and research, and these methods have been accruing significant attention in the past few decades.

A growing body of literature describes the usability of ML for multitudes of tasks, including: imagery analysis, such as medical imaging analyses [13, 14, 15] or object detection [16, 17]; personalised medicine [18, 19]; sequence analysis, such as speech recognition [20]; fault diagnostics and anomaly detection [21, 22]; generative applications [23]; applications for self-driving cars [24, 25], and other applications. The ML models are highly applicable in the fields where the explicitly implemented algorithm application is nearly impossible but can also provide a flexible alternative to numerical methods. This work addresses the ML applications to high-fidelity material modelling methods, which are essential to the automotive industry for product development.

The automotive industry is one of the largest industries in the world. 74.9 million vehicles were sold globally in 2019, and 63.8 million were sold in 2020 [26]. To produce a vehicle, an automotive manufacturer should develop a sound design. Designing may require a physical prototype or can be done with computer-aided engineering tools. Safety

must be kept at the forefront of the design. Crash tests on the physical models can be employed to ensure passenger safety, but this method is costly. One crash dummy can cost up to 1,000,000 GBP (or approximately 1,700,000 Canadian dollars), according to research in Cranfield University, GB [27]. From a long-term perspective, it is much cheaper to use computer aids to create a modelling tool capable of designing a safe vehicle. However, advanced high-fidelity modelling tools are required to substitute the experimental testing. The crystal plasticity finite element method (CPFEM) is an advanced high-fidelity modelling tool that accurately predicts material performance, but its accuracy comes at a high computational cost. Even for a modern, powerful supercomputer, it would be not feasible to perform the computations required for component-scale CPFEM modelling.

Scalability is a significant problem in crystal plasticity (CP), as the description of the deformation behaviour occurs on the microscopic level. In the Taylor-type homogenous CP theory, the deformation behaviour is obtained for each grain (single crystal), and the averaged result describes the macro deformation behaviour. The CPFEM is a full-field model that requires even more computational resources, and for this model, the computational time increases exponentially with the increase of elements in the FE mesh. CPFEM models are able to output the stress and strain partitionings for the input microstructures and provide crucial information on localisation formation that is not accessible in the Taylor-type CP models. For instance, in modelling a car's front rail crash, the deformation of millions of individual crystals must be considered. Such simulation is not feasible due to computational limitations of both the Taylor-type and the CPFEM models. To make it easier to understand the scale of the problem, figure 1.1, (a) illustrates the dimensions of individual crystals and figure 1.1, (b), represents an example of what is used for a front rail of a car.

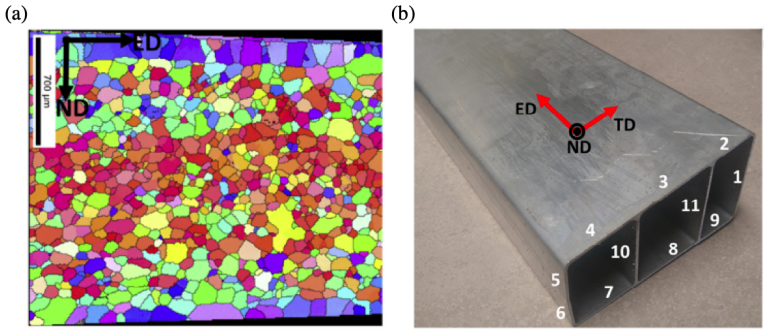

Figure 1.1: (a) Through thickness microstructure of the extruded AA6063-T6 material, shown on the next picture [1]; (b) Extrusion profile of AA6063-T6 material [1]

.

Crystal plasticity models can capture the material behaviour subjected to any applied loading and the local stress and strain partitioning and localisation in a given microstructure. The high fidelity of the crystal plasticity models comes at the cost of computational demand. The major challenge in implementing those models at the coupon- or component-level scales comes at the high computational cost. However, while it would be time-consuming to run specific simulations, it is feasible to collect extensive data on crystal plasticity simulations over a period of time. Therefore, it is viable for ML models to learn from this data, enabling rapid and high fidelity material behaviour predictions. ML models could supersede CP models when adequately trained and help obtain accurate predictions with lesser computational efforts. Moreover, ML methods are promising in terms of the scalability of such computationally expensive tools as CP. This research addresses gaps in ML applications to crystal plasticity and proposes new applications that help achieve rapid and high fidelity material modelling.

1.2 Scope and Objectives

The automotive and aerospace industries focus on adopting lightweight materials such as aluminium alloys to lower fuel expenditure and reduce carbon emissions. Addressing such optimisation can be achieved by advancing high-strength alloys or further developing existing designs and processes. An essential step in new alloy development and structural component design is to perform computer-aided engineering simulations, which generate optimised designs exhibiting superior crashworthiness and forming behaviour. In this regard, an accurate prediction of material deformation response is essential for optimising the impact performance and formability of designed components.

The use of CP models allows the calculation of accurate material deformation behaviour. However, it can only be used on the microscopic scale due to computational limitations. For that reason, these models are not implemented in commercial applications. This research hypothesises that machine learning algorithms can achieve rapid and high-fidelity microstructure-based predictions of material deformation behaviour. The possibility to scale up then emerges, and rapid high-fidelity microstructure-based material behaviour predictions on a macroscopic scale become feasible. This research addresses the significant computational demand of high fidelity simulations and uses machine learning methods to achieve rapid and accurate predictions of material plastic behaviour. The selected ML models are trained on datasets generated using crystal plasticity simulations to achieve that goal. Crystal plasticity models are advanced high fidelity microstructure based modelling tools applied for a wide range of mechanical problems. This research project uses

two different crystal plasticity models: the CP model under a fully constrained Taylor assumption (macro-level model) and the crystal plasticity finite element model (micro-level model). The main advantage of the Taylor-type homogenisation framework is the ability to solve polycrystal mechanical problems without using finite elements. Such models provide better accuracy than phenomenological models, and their computational time is much faster than crystal plasticity finite element applications. However, the Taylor-type assumption of homogeneous strain within grain aggregate disregards the factors of grain morphology and grain neighbour effects in polycrystals. Crystal plasticity finite element method models can account for these factors. Such models draw particular attention in research as they can capture fine deformation phenomena as orientation dependant strain localisation.

The main objectives for this work are the following:

1. Development of a machine learning-based framework to predict stress-strain behaviour and texture evolution for FCC single crystals and polycrystalline materials.
2. Development of a machine learning-based framework for predicting local stress and strain evolution for FCC polycrystalline materials.

1.3 Thesis Outline

This thesis consists of seven chapters. The current chapter is the first chapter, and it introduced the motivation for this research and set scope and objectives.

Chapter two presents the general background knowledge related to crystal plasticity and machine learning methods employed in this research. Crystal plasticity background includes material aspects (crystal structure, dislocation, crystallographic slip, texture representation and characterisation) and constitutive model. Machine learning background reviews artificial neural networks used in this research and the training process background: optimisation algorithms, activation functions, evaluation of neural networks, preparing and storing data for training and evaluation.

Chapter three presents a literature review. It includes a brief review of plasticity modelling, methods to accelerate computations in material science, database applications in computational material science, review of the significance of datasets and feature engineering in machine learning applications, machine learning applications in computational material science, and machine learning application to crystal plasticity. This section also identifies and summarises the gaps present in the literature.

Chapter four provides an overview of the research plan to achieve the thesis objectives.

Chapter five develops a machine learning-based framework to accurately predict stress-strain behaviour and texture evolution for complex strain paths for a wide range of face-centred cubic materials. The proposed framework comprises an ensemble of artificial neural networks and a crystal plasticity-update algorithm to allow accurate predictions of material behaviour for complex strain paths. The framework was successfully implemented and validated against the Taylor-type crystal plasticity model for monotonic and non-monotonic strain paths for single crystal and polycrystalline aggregates.

Chapter six develops a machine learning-based framework to achieve accurate local stress and strain evolution predictions for a wide range of face-centred cubic microstructures. The framework incorporates a single convolutional neural network trained to predict stress and strain partitionings of an aluminium microstructure under a proportional loading condition similar to uniaxial tension. Crystal plasticity finite element simulations for synthetically generated microstructures with defined material parameters were employed as a learning base for the convolutional neural network model. The framework was successfully implemented and validated against the crystal plasticity finite element model for a new set of synthetic microstructures. The flexibility of the convolutional neural network was demonstrated by its validation for two microstructures of AA5754 and AA6061 aluminium alloys.

Lastly, chapter seven presents the summary of the thesis, draws conclusions of the presented studies, and presents opportunities for future work.

Chapter 2

Background

2.1 Crystal Plasticity

Crystal plasticity models are a computational tool used to predict the loading behaviour of anisotropic crystalline materials, as well as texture evolution throughout the deformation process at finite plastic strains. This behaviour is anisotropic due to the crystallographic orientation dependence of deformational mechanisms. The CP methods were successful in modelling single-phase cubic metals where the dominant mechanism of plastic deformation was represented by crystallographic slip on specific slip systems of crystalline solids.

2.1.1 Microstructure

The CP models predict the elastic-plastic deformation and texture evolution of anisotropic heterogeneous crystalline solid matter. Crystalline materials consist of one or, usually, many crystals made of atoms. The arrangement of atoms into a regular, repeatable lattice characterises crystal structure. A lattice unit cell is the smallest possible subdivision of a crystal structure. The CPFEM often consider metals in its simulations. The crystal structure of most of the metals is either body centred cubic (BCC), face centred cubic (FCC), or hexagonal closest packed (HCP). For instance, aluminium has an FCC crystal structure. In the FCC crystal structure, atoms are located at all outer corners of a unit cell and in each centre of the cube faces.

Real crystalline solids very rarely have a perfect lattice. Usually, they contain lattice imperfections or defects. One can describe these imperfections geometrically. This description would depend on the type of imperfection: whether it is at the point, line, or surface

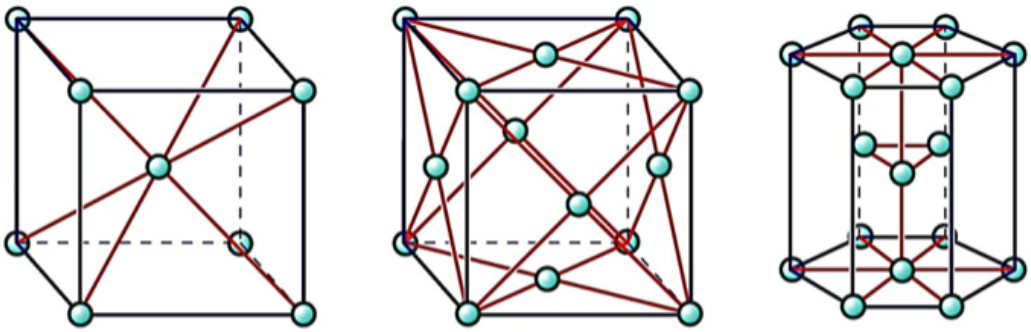

Figure 2.1: Crystal structures of some metals (left to right): Body Centred Cubic (e.g. Ferritic steels, Potassium, Molybden, etc.), Face Centred Cubic (e.g. Aluminum, Brass, Copper, Nickel, etc.), Hexagonal Closed-Packed Lattice (e.g. Magnesium, Titanium, Zirconium, etc.) [2]

imperfection. The most common point imperfections are vacancies, interstitial atoms, and substitutional impurity atoms. The most significant types of line and surface imperfections are a dislocation and grain boundaries, respectively.

Plastic deformation in polycrystalline materials is associated with the movement of dislocations within the crystal lattice. Taylor originally described dislocations as the shearing of different rows of atoms in the crystal lattice in the small regions and then yielding their growth though the crystal [28]. The shear stress along the slipping direction on the slip plane of dislocation is called the resolved shear stress (RSS) (figure 2.2). Critical resolved shear stress (CRSS) is required to initiate crystallographic slip. CRSS provides the force which causes dislocation motion to occur and that motion changes a material's geometry. There are other known mechanisms of plastic deformation in the material (twinning, grain boundary sliding, and diffusion); however, the slip is the prime one in the process of plastic deformation of FCC metals. Only this mechanism will be considered in the CP models related to this research.

Mechanism of crystallographic slip is anisotropic. That indicates that the movement of dislocations occurs along particular crystal planes in specific directions; these are called slip planes and slip directions respectfully. By definition, the slip system describes the set of symmetrically identical slip planes and associated family of slip directions. These slip planes and directions are almost always of maximum atomic density and conform to those slip systems in which dislocations are most likely to move. For an FCC metal, e.g. Aluminum, slip occurs along $< 110 >$ slip directions (there are three unique directions within each plane) and in the [111] slip plane (there are four of them). Consequently, given

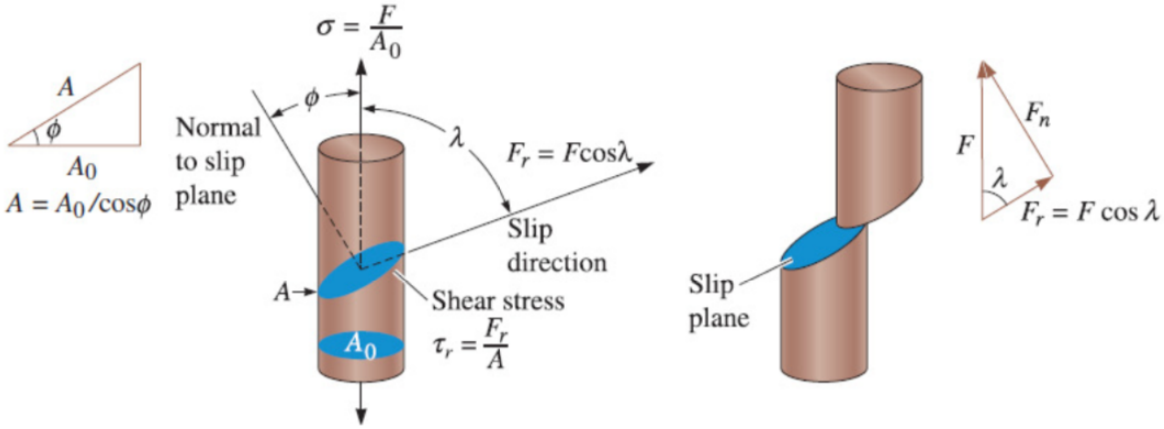

Figure 2.2: Visual representation of resolved shear stress [3]

the permutations of possible the slip planes and directions, there are twelve unique slip systems for materials with FCC crystal structure.

Assume that a unidirectional force F is applied to single crystal cylindrical morphology (figure 2.2). Then, for the dislocation to move in its slip system, a resolved shear stress produced by the applied force must reach its critical value. This value is called critical resolved shear stress and is an initial microscopic yield criterion for single crystals [29]. It is a parameter that depends on the material.

Schmid's law is used to find the resolved shear stress τ_r:

$$\tau_r = \frac{F}{A_0} \cos\lambda \cos\phi \tag{2.1}$$

In equation 2.1, F is the applied force, A_0 is the initial cross-sectional area, λ is the angle between the direction of slip and the applied force, and ϕ is the angle between the normal to the slip plane and the applied force.

Deformation textures

Commonly, when metals undergo medium to large deformations, the grains tend to orient themselves into preferred configuration [30, 4]. The resulting texture is the direct outcome of the manufacturing process.

Textures significantly affect mechanical properties of materials, i.e. their performance during fabrication and after, as the final products. The post deformation texture could unveil the production history of a polycrystal. It is very desirable to predict and simulate texture developments: many forming operations are conducted on rolled materials, and the performing capability of these materials is very closely associated with their textures. Being able to predict texture evolution for any applied strain path is significant for control purposes in industrial practices.

Texture representation

Polycrystalline aggregate constituent grains are characterised by their orientations. Orientation of a crystal describes the relationship between the coordinate frame of a crystal lattice and the coordinate frame of a sample. Crystallographic texture can be represented in multiple forms, including Euler angles, pole figures, inverse pole figures, and others [30].

Euler angles in the $Z-X-Z$ convention can be used to present crystal orientation. In this notation, an orientation is represented by a set of three angles $(\varphi_1, \Phi, \varphi_2)$, where $\varphi_1 \in [-180°, 180°]$ is the first rotation and is performed about $z-$axis , $\Phi \in [0°, 180°]$ is the second rotation and is performed about $x-$axis, and $\varphi_2 \in [-180°, 180°]$ is the third rotation and is performed about $z-$axis, where (x, y, z) is a reference base coordinate system. These angles can also be used to describe the orientation with rotation matrices. The rotation matrix representation depends on the Euler angles and is expressed as:

$$\boldsymbol{a}(\varphi_1, \Phi, \varphi_2) = \begin{bmatrix} \cos\varphi_1 \cos\varphi_2 - \sin\varphi_1 \sin\varphi_2 \cos\Phi & \sin\varphi_1 \cos\varphi_2 + \cos\varphi_1 \sin\varphi_2 \cos\Phi & \sin\varphi_2 \sin\Phi \\ -\cos\varphi_1 \sin\varphi_2 - \sin\varphi_1 \cos\varphi_2 \cos\Phi & -\sin\varphi_1 \sin\varphi_2 + \cos\varphi_1 \cos\varphi_2 \cos\Phi & \cos\varphi_2 \sin\Phi \\ \sin\varphi_1 \sin\Phi & -\cos\varphi_1 \sin\Phi & \cos\Phi \end{bmatrix} \quad (2.2)$$

In addition to a numerical representation of texture, the angular distribution of orientations in a polycrystalline solid can be graphically represented with pole figures [31]. The schematic representation of the pole figure projection sphere is depicted in figure 2.3. A pole figure represents a two-dimensional stereographic projection of crystallographic directions of all crystals.

Experimental Characterisation Methods

This thesis uses AA5754 and AA6061 microstructures to validate one of the proposed frameworks. These microstructures were obtained using the electron backscatter diffraction

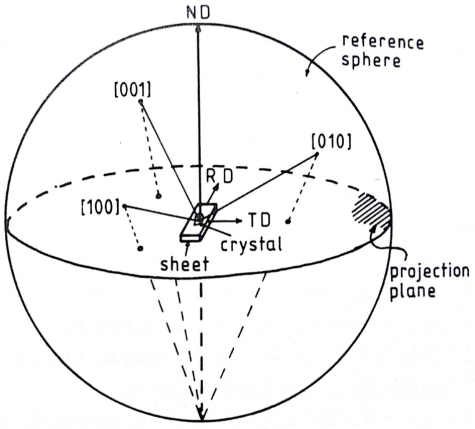

Figure 2.3: The schematic representation of pole figure projection sphere [4]

(EBSD) method. The EBSD method is a modern approach to extract crystallographic information about the microstructure with the use of a scanning electron microscope [32]. To perform an EBSD analysis of a sample, a stationary electron beam is directed at a tilted (at approximately 70°) sample surface, and the diffracted electrons arrange in a pattern that is detected by a fluorescent screen. The resultant pattern characterises the examined crystal orientation and structure. The EBSD analyses provide important microstructure characteristics such as texture orientation, grain size and morphology, grain boundary analysis, and phases of the analysed sample.

2.1.2 Crystal Plasticity Formulation

Schmid's law serves as the basis for activation of plastic deformation within the current crystal plasticity formulation. The law displays a relationship between the applied stress and the RSS on a slip system. It serves as a yield criterion for a for crystals within a polycrystalline solid. When RSS reaches its critical value, a crystal undergoes plastic deformation.

The total deformation is expressed as a function of two physics-based phenomena. The phenomena are the crystallographic slip due to motion of dislocations on the active slip systems, and the elastic distortion of the lattice. The deformation gradient F is:

$$\mathrm{F} = \mathrm{F}^{*}\mathrm{F}^{\mathrm{p}} \tag{2.3}$$

In equation 2.3, F^* denotes elastic stretching and rigid body rotations of crystal lattice. F^p denotes the deformation due to plastic shearing on crystallographic slip system. Slip systems are denoted by the pair (s, m) where s is a unit vector co-directed with the direction of slip, and m is a unit vector co-directed with a normal direction to a slip plane.

The rate of change of F^p, $\dot{F}^p$, is associated with the slip rate $\dot{\gamma}^{(\alpha)}$ of the α^{th} slip system according to:

$$\dot{F}^p.F^{p-1} = \sum_{\alpha} \dot{\gamma}^{(\alpha)} s^{(\alpha)} m^{(\alpha)} \tag{2.4}$$

In equation 2.4, the summation is over all activated slip systems. Accordingly, the vectors $s^{(\alpha)}$ and $m^{(\alpha)}$ are the (s, m) vectors related to the α^{th} slip system. The formulas of these vectors are defined as follows:

$$s^{*(\alpha)} = F^* \cdot s^{(\alpha)}, \quad m^{*(\alpha)} = m^{(\alpha)}.F^{*-1} \tag{2.5}$$

For each deformation stage, the velocity gradient is defined by:

$$L = F \cdot F^{-1} = D + W \tag{2.6}$$

In equation 2.6, the tensor D is symmetric and defines the strain-rate. The tensor W is skew-symmetric and defines the spin.

In its turn, the strain-rate tensor, D, is decomposed into elastic, D^*, and plastic, D^p strain-rates. Then, equivalent decomposition is performed for spin W:

$$D = D^* + D^p, \quad W = W^* + W^p \tag{2.7}$$

Then, plastic parts of strain-rate and spin could be defined as:

$$D^p = \sum_{\alpha} D^{(\alpha)} \dot{\gamma}^{(\alpha)}, \quad W^p = \sum_{\alpha} W^{(\alpha)} \dot{\gamma}^{(\alpha)} \tag{2.8}$$

For each slip system α, strain rate is defined as:

$$D^{(\alpha)} = \frac{1}{2} \left(s^{*(\alpha)} \otimes m^{*(\alpha)} + m^{*(\alpha)} \otimes s^{*(\alpha)} \right) \tag{2.9}$$

For each slip system α, spin is defined as:

$$\mathrm{W}^{(\alpha)} = \frac{1}{2}\left(s^{*(\alpha)} \otimes \mathrm{m}^{*(\alpha)} - \mathrm{m}^{*(\alpha)} \otimes s^{*(\alpha)}\right) \tag{2.10}$$

The elastic strain rate of the lattice and the Jaumann rate of Kirchoff stress are related as:

$$\overset{\nabla}{\tau}{}^{*} = \dot{\tau} - \mathrm{W}^{*}\tau + \tau\mathrm{W}^{*} = \mathcal{L}\mathrm{D}^{*} \tag{2.11}$$

In equation 2.11, $\overset{\nabla}{\tau}{}^{*}$ is the Jaumann rate of Kirchoff stress. $\mathcal{L}$ is a tensor of the elastic moduli.

Equation 2.11 can be also defined in terms of the Jaumann rate of Cauchy stress $\overset{\nabla}{\sigma}$ by defining a tensor $\mathrm{R}^{(\alpha)}$:

$$\mathrm{R}^{(\alpha)} = \mathcal{L}\mathrm{D}^{(\alpha)} + \mathrm{W}^{(\alpha)}\sigma - \sigma\mathrm{W}^{(\alpha)} \tag{2.12}$$

Using equations 2.7 to 2.12, the constitutive equation can be presented in the form of the Jaumann rate of Cauchy stress, $\overset{\nabla}{\sigma}$:

$$\overset{\nabla}{\sigma} = \mathcal{L}D - \dot{\sigma}^{0} - \sigma \operatorname{tr} D \tag{2.13}$$

In equation 2.13, $\dot{\sigma}^{0}$ is a term of a visco-plastic stress. It is defined as:

$$\dot{\sigma}^{0} = \sum_{\alpha} \mathrm{R}^{(\alpha)}\dot{\gamma}^{(\alpha)} \tag{2.14}$$

The term $\tau^{(\alpha)}$ is the RSS on the slip system α. It is related to the Cauchy stress, σ, as follows:

$$\tau^{(\alpha)} = \mathrm{D}^{(\alpha)} \cdot \sigma \tag{2.15}$$

In the rate-dependent formulation, the slip rate of the slip system α, $\dot{\gamma}^{(\alpha)}$, is dependant on the corresponding RSS, $\tau^{(\alpha)}$:

$$\dot{\gamma}^{(\alpha)} = \dot{\gamma}_0 f^{(\alpha)} \left(\frac{\tau^{(\alpha)}}{g^{(\alpha)}} \right) \tag{2.16}$$

In equation 2.16, the constant $\dot{\gamma}_0^{(\alpha)}$ is the reference strain rate on the slip system α. $f^{(\alpha)}$ is a general function that describes the relation of the slip strain-rate to RSS. Defining $f^{(\alpha)}$ as a power-law expression, equation 2.16 becomes:

$$\dot{\gamma}^{(\alpha)} = \dot{\gamma}_0 \operatorname{sgn} \tau^{(\alpha)} \left| \frac{\tau^{(\alpha)}}{g^{(\alpha)}} \right|^{1/m} \tag{2.17}$$

In equation 2.17, m is the parameter of strain-rate sensitivity. The function $g^{(\alpha)}$ defines the strain-hardening of the slip system α. A single crystal work hardening is characterised by the evolution of strain hardening according to the incremental relationship:

$$\dot{g}^{(\alpha)} = \sum_{\beta} h_{\alpha\beta} \dot{\gamma}^{(\beta)} \tag{2.18}$$

In equation 2.18, $h_{\alpha\beta}$ is a tensor that determines the hardening rate on slip system α due to shearing on slip system β. Every element of $h_{\alpha\beta}$ is dependant on the history of deformation. The hardening moduli are chosen to incorporate the effects of slips on all the systems on the active hardening of each system, $h_{\alpha\alpha}$, and in the latent hardening of each system, $h_{\alpha\beta}$ ($\alpha \neq \beta$). The tensor $h_{\alpha\beta}$ can be represented as:

$$h_{\alpha\beta} = q_{\alpha\beta} h_{\beta} \quad \text{(no sum on } \beta) \tag{2.19}$$

In equation 2.19, h_{β} is a single crystal hardening, and the tensor $q_{\alpha\beta}$ represents the latent hardening behaviour of a crystalline solid. The form of this tensor is the following:

$$q_{\alpha\beta} = \begin{bmatrix} A & qA & qA & qA \\ qA & A & qA & qA \\ qA & qA & A & qA \\ qA & qA & qA & A \end{bmatrix} \tag{2.20}$$

In equation 2.20, the term q is the ratio of latent hardening rate, and As are the 3×3 matrices fully populated with ones.

2.1.3 General Introduction to Polycrystal Deformation Models

A polycrystal deformation model should exhibit beneficial advantages in addition to the general capabilities of phenomenological models. It should be able to model phenomena that are out-of-reach of macroscale phenomenological theories. Crystallographic textures prediction is an example of such phenomena. Generally, a single crystal deformation model could be used to derive a model capable of that. Then, the following question arises: how can one define the relationships between microstructural mechanisms of deformation operating on the single crystal level and overall polycrystal behaviour?

To relate the deformation of a polycrystalline aggregate to that of constituent individual grain, something that must be known or assumed about the stresses and strains of individual crystals. Usually, a polycrystal's stress and strain distribution assumptions are made. Polycrystal response is identified with average (weighted or not) of the response of its constituent grains. Multiple models have been proposed, and they provided significant insight into texture development and hardening response. Some of these models are described below. All of them are based on the concept of plastic deformation due to slip in FCC metals' single crystals. Other deformation mechanisms are not considered in the scope of this work.

Sach's Model

Sach's model is one of the earliest in the history of modelling polycrystal deformation. An assumption of this model is that each constituent crystal is subjected to the same stress, which is the same as macroscopic stress. Sach considers the grains as an array of free single crystals that deform independently. Texture and deformation are deducted from a stress state. In 1941 Kochendorfer [33] updated the model with the assumption of each grain being subjected to the same stretch.

Due to the assumption that each constituent crystal is subjected to the same stress as a macroscopic, the stresses arising from constraints necessary to satisfy an imposed strain are neglected. This results in a violation of strain continuity across a grain boundary [34, 35]. Numerical deviation from an experiment is existent in this model [36]. Notably, this theory was not accurate in predicting deformation textures.

Taylor's Model

Taylor's assumption was different from Sach's: he proposed that all the grains are subjected to the same strain as a polycrystal [37]. He deduced that from his experimental

observations. An examination of a cross-section micrograph of a drawn wire showed that all the grains were elongated in the direction of extension. Additionally, they were contracted in the two perpendicular directions. Consequently, the conclusion of a strain field being homogeneous was drawn. This assumption linked the behaviour of a deformation of a polycrystalline solid and its single constituent crystals. Moreover, this assumption has a benefit of satisfying the condition of continuity of the strain rate across the grain boundaries, which implies that no voids are created. Since the strain on all the constituent crystals is the same as the macro strain of the polycrystal, it was evident that stress state would not be continuous, and therefore it varies from grain to grain abruptly. As it was pointed out by Bishop and Hill [34, 35] each grain satisfies the relation:

$$\frac{\sigma_g}{\tau} = \frac{d\gamma}{d\varepsilon} = M \tag{2.21}$$

Where σ_g and $d\varepsilon$ are the axial stress in the grain and the macroscopic aggregate strain increment, respectively. τ and $d\gamma$ are the shear strength and slip-system shear strain increment, respectively. M is an orientation factor, depending only on the lattice geometry and the relationship between the loading axis and the slip system of a crystal. The uniaxial tension of aluminium polycrystals was studied by Taylor in 1923 [38]. Taylor assumed that each grain is at the same stage of strain hardening; he predicted that the yield stress of a random aggregate would be 3.06 τ_y, where τ_y is the yield strength in shear of a single grain. A very similar result was obtained when Taylor tested this theory by comparing the tensile stress-strain relation ($\sigma_{aggr}{}^{\smile}\varepsilon$) measured on the aggregate with that deducted from the shear stress-strain ($\tau - \gamma$) curve of a single crystal where

$$\sigma_{aggre} = \bar{M}\tau, \tag{2.22}$$

$$\varepsilon = \frac{\gamma}{\bar{M}}\tau, \tag{2.23}$$

where $\bar{M}$ represents the Taylor factor, and its value is dependent on the texture of polycrystalline material. For isotropic polycrystalline aggregates, its value is approximately 3.06. Two main points of the Taylor theory with regards to the deformation relation of single crystals and polycrystals can be summarized as follows:

1. A single crystal's and macroscopic deformations are the same; however, there is no morphological consideration made.

2. The macroscopic stress of a polycrystal is the average of the stress states of its constituent single crystals.

These concepts are adopted for analysis and modelling in the thesis.

2.1.4 Finite Element implementation

For the CNN-based framework, the crystal plasticity model was incorporated into a two-dimensional FE framework [39, 40, 41]. The Lagrangian formulation of the field equations was used in the basis of the FE algorithm with the usage of convected coordinates [40]. Let θ_i be the base vectors in the undeformed configuration of the material corresponding to the convected coordinates χ_i. Let the initial volume of the undeformed configuration of body be V, and the surface area be S. Next, the deformed base vectors will be $\Theta_i = \boldsymbol{F}\chi_i$, where $\boldsymbol{F}$ is a deformation gradient. Then, the equilibrium equations for quasistatic deformations can be expressed using the virtual work condition for arbitrary variations of the displacement components u_i and corresponding variations of the components η_{ij} of the Lagrangian strain tensor:

$$\int_V \tau^{ij}\delta\eta_{ij}dV = \int_S T^i\delta u_i dS \tag{2.24}$$

where τ^{ij} are the component of the undeformed basis of the second Piola-Kirchhoff, and and T^i are the corresponding traction vectors. Then, equation 2.24 can be expanded with Taylor series to determine equations for the field quantity rates:

$$\Delta t\int_V \left(\dot{\tau}^{ij}\delta\eta_{ij} + \tau^{ij}\dot{u}_{k,i}\delta u_{k,j}\right)dV = \Delta t\int_S \dot{T}^i\delta u_i dS - \left[\int_V \tau^{ij}\delta\eta_{ij}dV - \int_S T^i\delta u_i dS\right] \tag{2.25}$$

This step is required as the problem is considered to be linear-incremental, and the current state of equilibrium must be known for any time t. The last term of the right hand side of equation 2.25 allows correction of equilibrium correction to prevent any drift away from the true equilibrium path during the incremental procedure [40].

The elements of the FE mesh used in this research were akin to those presented by [42]. These elements have quadrilateral shape. Each element contains four linear velocity triangular sub-element in order to employ a higher order integration scheme. The CPFEM model is incorporated using a parallel computing [39, 40] algorithm to accelerate computational procedure for data collection.

2.2 Machine Learning Background

AI is a fast-developing and vast field. Its history finds its beginning in myths and stories of an ancient world, e.g. giant automation of bronze, Talos, who protected Crete from its enemies, stories of which dated to around 400 BCE. However, the field of AI as a research field was established long after, in summer of 1956 at a campus' workshop of Dartmouth College [43].

Machine learning is a sub-field of AI which aims to perform a task of interest without using explicit instruction and using a set of given data prepared by a human. The end product of training a model is a computational scheme (or a complex multivariable function), but not an explicitly programmed algorithm. In other words, ML is an application that provides systems with the ability to learn and improve automatically from finding patterns and inference from data, without being explicitly programmed. Deep learning is a sub-filed of ML, where the word "deep" refers to a number of successive layers in the model. The essential parts of a successful application include a proper set of data accompanied by a appropriate feature design, selection of a machine learning method, optimal parameters and hyperparameters tuning, choice of the right training algorithm and evaluation scheme.

2.2.1 Machine Learning: Supervised and Unsupervised Learning

Machine Learning methods can be generally split into two classes: supervised learning and unsupervised learning. Supervised learning is a task of fitting (or "learning") a function that maps an "input" (features, or some parameters of interest) to an "output" (a target of interest) based on example pairs (input-output) that were predefined by a human [44]. The example of supervised learning task could be a cat vs dog classifier: a model which, given a picture, could "classify" it as a "cat" or "dog" or "neither" of those. Supervised learning models are trained on a "labelled" data. Labelling data means creating input-output pairs, for example, marking a picture of a cat with a "cat" label. Unsupervised learning is a task of automatically finding previously unknown patterns of given data [44]. In this scenario, the data was not previously labelled by a human. It is fed to a model after it was correspondingly processed. An excellent example of unsupervised learning application would be splitting the market into several categories ("clusters"). Figure 2.4 represents an example of K-means clustering for $K = 3$. Another known application of unsupervised learning is a dimensionality reduction method [45]. Unsupervised learning models are also used to model probability densities of the given input. Nowadays, supervised learning applications are more common than unsupervised learning applications.

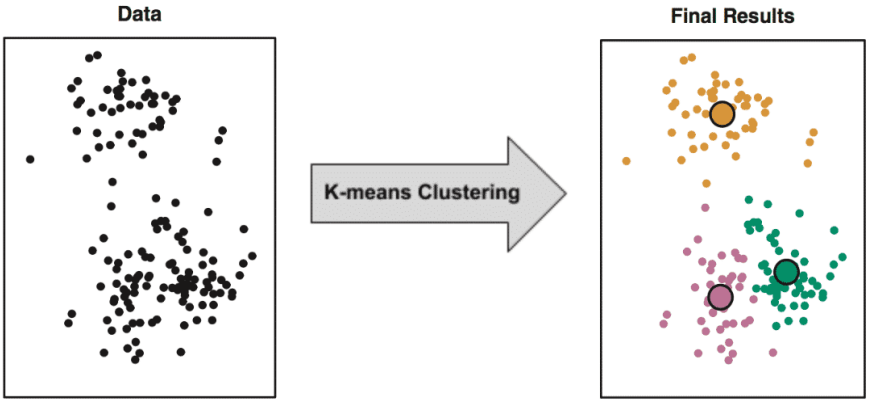

Figure 2.4: An example from K-means clustering [5]

2.2.2 Machine Learning: Regression and Classification

There are two basic types of ML problems: classification and regression. As for classification, it is a process of determining to which class given input corresponds. The classes could either be predefined (e.g. supervised learning) or not (e.g. splitting the given data to K different classes: the case of unsupervised learning). Let us get back to a previously discussed example. An ML model trained on the proper data can classify pictures into "a cat pictures" and "not a cat picture"; in this case, there will only be two classes. ML model that was trained to distinguish for two classes is called a binary classifier.

The ML framework developed within this research will focus on a regression-type approach. Regression is the type of ML method that predicts continuous data, meaning that there is no limited amount of classes, but the method can predict the continuous (not discrete) number or a tensor consisting of continuous quantities. Some of the examples of regression models applications are the prediction of prices of a stock market, quantitive prediction of parameters of interest, image generation.

2.2.3 End-to-end Learning

End-to-end (E2E) learning refers to training a possibly complex machine learning model for a system as a whole [46]. In the E2E learning, a training set would consist of just input audio files and outputs of transcripts. In the case of a non-E2E approach, there would be one or many of intermediate steps within the model such as, for example, Mel frequency

Cepstral coefficients (MFCC) algorithm [47, 48] to extract specific voice low-level features from the audio clip. In the proposed research, a non-E2E learning approach is considered.

2.2.4 Artificial Neural Networks: Review

Numerous types of artificial neural networks exist, and researchers proceed developing new types and advances to existing types regularly. Training a NN can also be referred to as "deep learning". Generally, deep learning is a powerful tool for obtaining a non-linear real-valued function. This function is an approximation mapping of input to the desired output. Training occurs under the smoothness assumption: the sought-for mapping can be represented as a Lipschitz-continuous function.

Artificial Neural Networks: Feed Forward Neural Network

Feed Forward Neural Network (FFNN) is a most basic type a neural network. One can refer to them as "perceptrons". Connections within the FFNNs do not form cycles. FFNNs are described as having successive layers. This description is common for many network types but originates from FFNNs. The first layer is a input layer, then a series of hidden layers and an output layer (see figure 2.5). Each layer consists of the nodes or unit cells. One can think of a layer as a data-processing module that filters data according to a defined mathematical law [45]. The hidden layer has a name of "hidden" because the values for hidden unit cells ("weights") are not observed, unlike the values of input and output cells. After training a desirable network, its architecture, hyperparameters, and the weights (and biases) values are saved for using the model, and training and testing sets are nor longer needed. A basic neural network cell is shown in the figure 2.6. An explanation of what are parameters and hyperparameters will be provided later in this chapter.

One can call neural networks shallow or deep. An example of a shallow NN is a 2-layer network: it consists of an input layer, one hidden layer, and an output layer. The example of a deep neural network could be a 100-layer network (with 99 hidden layers). A shallow neural network can refer to a neural network with one hidden layer and a small amount of nodes [49]. Nevertheless, there is no strict line drawn between how many hidden layers shallow and deep networks would have. In general, training a network with more than two hidden layers can refer to deep learning.

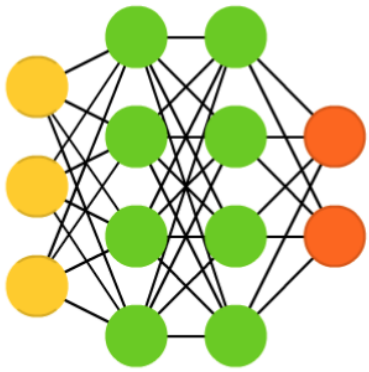

Figure 2.5: Deep feed forward neural network, originally called as "perceptron" [6]

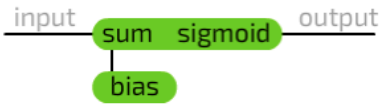

Figure 2.6: A basic neural network cell [7]

Artificial Neural Networks: Convolutional Neural Network

Convolutional neural networks (CNNs) is another class of artificial neural networks. The advantage of CNNs is their ability to take two- and three-dimensional tensors as an input and also output tensors of such dimensions. In this research, a CNN constituent of convolutional and deconvolutional layers is employed.

Convolutional and deconvolutional layers are formed by implementing a direct convolution and a transposed convolution operations on square neuron layers of CNNs. The operation of direct convolution can be described with the following equation:

$$x_{ij}^{[L]} = \sum_{\xi=0}^{m-1} \sum_{\zeta=0}^{m-1} \theta_{\xi\zeta} x_{(i+\xi)(j+\zeta)}^{[L-1]}, \tag{2.26}$$

where x is the pre-nonlinearity square layer of a filter, L is a positive integer number and denotes the number of the layer ($L \in [0, \ell]$, where $L = 0$ is the index of an input

and $L = \ell$ is the index of an output layer), and θ is the filter matrix of the dimension (kernel size) $m \times m$ and it is populated with trainable parameters. Direct convolution results in downsampling of an input layer. The transposed convolution operation is similar to direct convolution, and is implemented by swapping the forward and backward passes of a convolution and serves as a trainable upsampling.

Padding parameter can be applied to control dimensions of output of convolution operations. The term "padding" refers to adding zeros to the borders of filters prior to applying a convolution operation. Two type of paddings are used in this research: "same" and "valid". "Valid" padding refers to no padding, and "same" results in dimensions of filters of input and output of convolutions being equal.

2.2.5 Optimisation

Training is an optimisation procedure. It is a process of minimising the parameters of NN so that the loss function value on these parameters is minimised. Using optimisation algorithms, parameters are updated in the opposite direction of the gradient of the cost function. Gradient descend (GD) lies in the foundation of the most optimisation algorithms and originates from a stochastic approximation method by Robbins and Monro [50]. GD-based algorithms often referred as a "black-box" optimisers since it is hard to mathematically explain their advantages and drawbacks [8].

Batch Gradient Descend

Batch gradient descend (BGD) is the most basic GD-based algorithm. It computes the gradient of the cost function $\nabla_\theta J(\theta)$ with respect to the parameters for the entire dataset:

$$\theta = \theta - \alpha \cdot \nabla_\theta J(\theta) \tag{2.27}$$

The drawback of this methods is that it will take too long per iteration in the case of large dataset.

In equation 2.27 and in all subsequent equations, α refers to a hyperparameter of learning rate and it will be discussed further in this chapter.

Stochastic Gradient Descend

Some dataset are so large so it becomes impractical and sometimes not even possible to optimise on the entire set. In contrast to BGD, one can update parameters for each training example using stochastic gradient descend (SGD):

$$\theta = \theta - \alpha \cdot \nabla_\theta J\left(\theta; x^{(i)}; y^{(i)}\right) \tag{2.28}$$

But it happens that if the dataset is very large, SGD algorithm will perform too many redundant computations and will be very slow.

Mini-Batch Gradient Descend

The middle ground between BGD and SGD is a mini-batch gradient descend (MBGD). Mini-batch is a portion of the entire dataset; it typically consist of $m_{batch} = 2^n$ training examples where n is a natural number from 2 to 16. Nevertheless, this number could vary for different applications. MBGD calculates an update for each mini-batch of m_{batch} training examples

$$\theta = \theta - \alpha \cdot \nabla_\theta J\left(\theta; x^{(i:i+m_{batch})}; y^{(i:i+m_{batch})}\right) \tag{2.29}$$

Despite the computational benefit with regards to the processed amount of data, MBGD does not guarantee good convergence [8]. Few improvements have been proposed for this algorithm as discussed next.

Momentum in Gradient Descent

One of the biggest challenges for MBGD is navigating thru "hills" and "valleys" of the hyperplane of a cost function [51]. This could yield optimising towards a local minimum instead of a preferred global minimum. In some cases, the algorithm will oscillate back and forth on a surface of hyperplane without making noticeable optimisation progress. The concept of momentum [52] addresses this issue.

The momentum stimulates GD-based algorithm to accelerate optimisation in the relevant direction. Visual interpretation of the previous statement is shown in figure 2.7. Momentum concept is relevant to a concept of an exponentially weighted moving average

[53]. Momentum accelerates GB by adding a ratio β of the new update vector of the previous iteration to the current update vector:

$$\begin{aligned} v_t &= \beta v_{t-1} + \alpha \nabla_\theta J(\theta) \\ \theta &= \theta - v_t \end{aligned} \tag{2.30}$$

Figure 2.7: SGD without momentum (on the left) and SGD with momentum (on the right) [8]

Nesterov accelerated gradient

Nesterov accelerated gradient (NAG) [54] gives a better "sence" of direction towards a performed update. It will better approximate future values of the parameters:

$$\begin{aligned} v_t &= \beta v_{t-1} + \alpha \nabla_\theta J\left(\theta - \beta v_{t-1}\right) \\ \theta &= \theta - v_t \end{aligned} \tag{2.31}$$

Adam

Adaptive Moment Estimation (Adam) [55] is one of the recently developed (2014) optimisation algorithms. It computes adaptive learning rates for each parameter. It stores an exponentially decaying average of past squared gradients v_t like other algorithms. Yet, what is does differently, is that it keeps an exponentially decaying average of past gradients m_t, similar to momentum:

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + \left(1 - \beta_1\right) g_t \\ v_t &= \beta_2 v_{t-1} + \left(1 - \beta_2\right) g_t^2 \end{aligned} \tag{2.32}$$

m_t and v_t are approximations of the first moment (the mean) and the second moment (the non-centered variance) of the gradients. Bias-corrected first and second moment estimates are computed to counter moments being biased towards zero:

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$$
$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{2.33}$$

Eventually, parameters are updated according to:

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{\hat{v}_t} + \epsilon}\hat{m}_t \tag{2.34}$$

Other Algorithms

There is a set of existing optimisation algorithms, and many of them are based on gradient descend with an incorporated concept of momentum. The list includes but is not limited to: Adagrad [56], Adadelta [57], RMSProp (unpublished, proposed by Geoff Hinton), Adamax [55], Nadam [58], MaxProp [59]. Figure 2.8 shows a visualisation for possible variations of the training process on a cost function surface.

No optimiser fits all the problems better than others. Training an ANN model requires many iterations of trial and error based on a different combination of algorithms and hyperparameters.

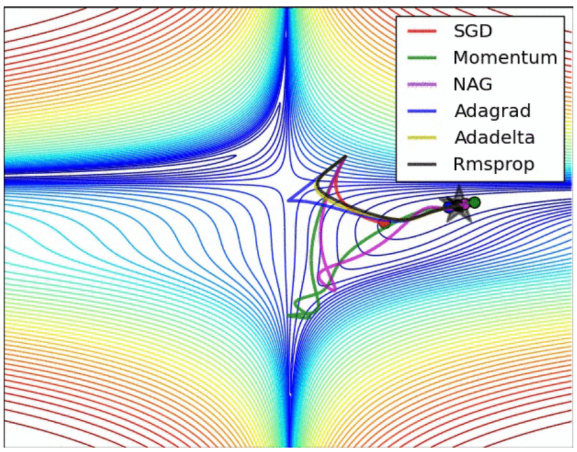

Figure 2.8: Visual comparison of optimisation paths for different algorithms on a loss surface contours [8]

2.2.6 Activation Functions

Activation functions are used in NNs are the functions that transform a value in a unit cell in, the most usually, a non-linear way. Applying such functions to the layer of a neural network will not only transform the values of the weights within layers but will also typically constrain the range that output can take.

There are many different activation functions with various properties (e.g. differentiable, monotonic, smooth, with monotonic derivatives, and have different ranges). Expectedly, there is no best one-fits-all activation function, and different applications could require the use of specific functions. Let us consider the most common types of activations functions: sigmoid, tanh, and ReLU.

Sigmoid activation function

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.35}$$

An activation function with a range of $[0, 1]$. Usually used alongside the normalisation between zero and one.

Hyperbolic tangent activation function

$$\tanh(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})} \tag{2.36}$$

An activation function with a range of $[-1, 1]$. Usually used alongside the normalisation between negative one and one.

ReLU activation function

$$f(x) = \begin{cases} 0 & \text{for } x \leq 0 \\ x & \text{for } x > 0 \end{cases} \tag{2.37}$$

ReLU is an abbreviation for "rectified linear unit". An activation function with a range of $[0, +\infty]$. Usually used alongside the normalisation by standard deviation.

2.2.7 Evaluation of Neural Networks

Loss function is a function that measures the performance of the model on the training data. The loss function is a function that is being optimised (i.e. minimised) by an optimisation algorithm. It is used to guide the optimisation of parameters to "right" direction [45]. Loss function measures how well is the model's performance on one training example.

There is also a cost function. It measures how well the parameters θ and b do on the training set. In this work, loss function will be denoted by $\mathcal{L}\left(y^{(i)}_{pred}, y^{(i)}_{true}\right)$ and cost function, defined as a sum of all losses over the set, will be denoted by $J(\theta, b)$. Cost function depends on the loss function as follows:

$$J(\theta, b) = \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}\left(y^{(i)}_{pred}, y^{(i)}_{true}\right) \tag{2.38}$$

Different models prefer different loss functions for their training. Loss functions vary for classification and regression problems. The simplest example of loss function for a classification problem is a logistic regression loss function:

$$\mathcal{L}\left(y^{(i)}_{pred}\right) = y^{(i)}\log y^{(i)}_{pred} + \left(1 - y^{(i)}\right)\log\left(1 - y^{(i)}_{pred}\right) \tag{2.39}$$

The most popular pick for a regression problem loss function is a mean squared error. It is preferable because it is easy to compute its gradient, and it provides an adequate error measure. An example of inadequate error measure is an absolute percentage error (APE). APE has lots of drawbacks and is not recommended (for both loss and metrics) for applications with fluctuating values. For example, it cannot be used when zero values are part of the output (due to division by zero). In addition, APE is not a very consistent metrics. Let us compare two cases: (1) $y_{true_1} = 0.001$ and $y_{pred_1} = 0.101$, and (2) $y_{true_2} = 100$ and $y_{pred_2} = 100.1$. In the both cases the difference between the results is 0.01, and $APE_1 = 10000\%$ and $APE_2 = 0.1\%$, and $MAPE = 5000\%$ (mean APE). While the error itself could be not critical for calculations, it can yield misinterpretation of the results, which could have happened in the presented work according to the error comparison. The same rules applies to metrics: different metrics formula should be chosen for different cases. Mean squared error (MSE), equation 2.40, is a common loss function for regression problems. MSE could also be used as a metric. For regression, it is also common to use mean absolute error (MAE), equation 2.42.

The formulae for MSE, MAPE, and MAE are:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(y_{true}^{(i)} - y_{pred}^{(i)}\right)^2 \tag{2.40}$$

$$\text{MAPE} = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_{true}^{(i)} - y_{pred}^{(i)}}{y_{true}^{(i)}}\right| \tag{2.41}$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|y_{true}^{(i)} - y_{pred}^{(i)}\right| \tag{2.42}$$

2.2.8 Improving Neural Networks

Training an accurate ML is an iterative process and has its own workflow. It all starts with creating a baseline model. Baseline line model is a naive approach: basically, the first idea should be implemented as fast as possible, and the results should be observed. In rare cases, the results could be satisfactory, and as such the ML model can be employed. However, these cases usually require a lot of experience and expertise. There is no rule of thumb for which model should be tried at first. The next step is to use error analysis to determine the next most promising steps and iteratively improve the accuracy of the model.

Training, Development, and Test sets

To train and validate a baseline model an ML engineer requires a training dataset, development dataset and test dataset. Training dataset is a set used to training a ML learning model. Development set is also used while training, but it is not being trained on. It is used to assess the error while training occurs. When both training and development errors are satisfactory, error on the test set can be computed to evaluate and validate a performance of machine learning model.

What is the split for training, development, and test sets? It depends on the amount of data available. For relatively small data sets ($\sim 100 - 1,000$ examples), it is common not to have a development dataset at all and employ a K-fold cross-validation [60] method on the whole training set. For the moderate amount of data ($\sim 10,000$ examples) it is common to see a 70%/15%/15% split for training/development/test data. It gets individual for a large amount of data. For example for $\sim 10,000,000$ examples it could be adequate to

use $\sim 150,000$ examples for each development and test sets. In this case, the split will be 97%/1.5%/1.5%. It is important to note that all three sets should come from the same distribution. Development and test sets' domain should be similar to the domain of the training set.

Shuffling Data

Before training a model, one has to load the data and shuffle it. Shuffling the data is usually very important before training a NN because it improves the predictive performance of a model. When using such optimisation methods as mini-batch gradient descent it is beneficial to have heterogeneous examples from the same distribution as the whole dataset in one batch.

Parameters and Hyperparameters

Parameters of the neural network are the weights θ and biases b. They are common to be denoted just by the Greek letter θ. Each layer of a network has weights and biases associated with it - optimiser updates these values such that the loss function takes on its optimal values. In contrast, hyperparameters are the values that are set by a machine learning engineer before training. They include learning rate, how many epochs (fitting iterations) of gradient descend, the depth of a neural network, how many hidden units are there in the neural network, momentum term, mini-batch size, regularises. One can refer to hyperparameter optimisation as tuning. Some hyperparameters are unique for each model, and some of them have preferred values (though they can be tuned as well).

Overfitting and Underfiting: Bias and Variance problem

Overfitting is a tendency of an ML model to perform significantly worse on a new data in comparison to its performance on training data. As for underfitting, it is when an ML model cannot determine the fundamental trend of the data at all. The phenomena of over- and underfitting are closely related to a bias-variance trade-off. High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting). High variance can cause an algorithm to model the random noise in the training data, rather than the intended outputs (overfitting).

The methods of reducing bias (underfitting) include: adding more of training data, making a deeper neural network, feature engineering, changing a normalisation method

(and/or activation functions), modifying the model architecture. As for the methods of reducing variance (overfitting), they include: adding more of training data, reducing the size of the neural network, adding regularisation, employing early stopping, feature engineering, changing a normalisation method (and/or activation functions), modifying the model architecture. As can be seen, many methods are similar, but they require individual approaches in each case. Feature engineering is the process of using the expert knowledge about the data and about the machine learning algorithm at hand. This process is employed to make an algorithm learn better. Proper feature engineering typically allows solving the problem with fewer resources and/or less data.

However, even after the feature set was optimised to its best, when training a machine learning model one can run into the problem of overfitting. It happens always when using deep neural networks, and met as a challange much more often than running into a problem of underfitting. Regularisation techniques address the problem of overfitting and help a model to achieve better accuracy. The regularisation techniques discussed is the next section, and they adress neural networks applications. Overfitting generally could be explained by the fact that the range of the possible hypotheses too large. When the neural network is very deep, it has numerous parameters. Hence, the space of possible solutions could be excessively broad. When there is not enough data to constrain the hypothesis, the hypothesis that performs well on the training data can be chosen. However, this does not guarantee us the hypothesis working well on the development and test data. One of the ways of conquering overfitting is to reduce the parameter space. When there are too many features and a relatively small amount of data, overfitting can become a problem. Reducing the number of features has its disadvantages and doing so yields elimination of meaningful the data, which could be useful for a problem.

Regularisation

Regularisation is an extra alternative that could be used to reduce overfitting. L_1 or L_2 regularisation techniques [61] are frequently adopted for applications in neural networks. When applying these regularisations, the parameters are left in place, but the values of the parameters θ are penalised. The modification of the cost function achieves the goal of reducing the values of weights by addition of a penalising term. Therefore, when using gradient descent, the cost function becomes a sum of all losses of the individual predictions over all the training examples, plus the regularisation term.

For L_1 regularisation, the cost function would be the following:

$$J\left(\theta^{[l]}, b^{[l]}\right) = \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}\left(y_{true}^{(i)}, y_{pred}^{(i)}\right) + \frac{\lambda}{2m}\sum_{l=1}^{L}\left\|\theta^{[l]}\right\|_1 \tag{2.43}$$

As for the L_2 regularisation:

$$J\left(\theta^{[l]}, b^{[l]}\right) = \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}\left(y_{true}^{(i)}, y_{pred}^{(i)}\right) + \frac{\lambda}{2m}\sum_{l=1}^{L}\left\|\theta^{[l]}\right\|_F^2 \tag{2.44}$$

In equations 2.43 and 2.44, λ is a regularisation parameter. The regularisation parameter is the other hyperparameter that needs to be tuned while training a neural network. The objective of training the neural network when using regularisation is to minimise the cost function with the additional regularisation term.

For L_1 regularisation, the term $\|\theta\|_1$ denotes the L_1 norm:

$$\|\theta\|_1 = \sum_j |\theta_j| \tag{2.45}$$

For L_2 regularisation, the term $\|\theta\|_F^2$ denotes squared Frobenius norm, the sum of all squared elements of a weight tensor:

$$\|\theta\|_F^2 = \sum_j \theta_j^2 \tag{2.46}$$

It has been shown that with the implementation of regularisation, sample complexity grows smaller with the amount of given features [62]. Hence, less amount of data need to be provided for successful training. As a note for the intuition of how the hyperparameter λ must be tuned: if the parameter λ is too high, then it would penalise the parameters θ so much, that their values are going to be close to zero. Since it is widespread to use activation functions that are linear-like near-zero (say, ReLU or tanh), if the parameters are close to zero, then the neural network would be similar in behaviour to a linear function. However, the neural network, similar to a linear function, would result in underfitting. L_2 regularisation technique is the most used in the deep learning applications and is also referred to as a "weight decay" [63] regularisation method.

Dropout [64] is another known regularisation technique used to prevent overfitting in neural networks. The term "dropout" is self-explanatory. It refers to probabilistic elimination of unit cells (both hidden and not hidden) out of the network while training.

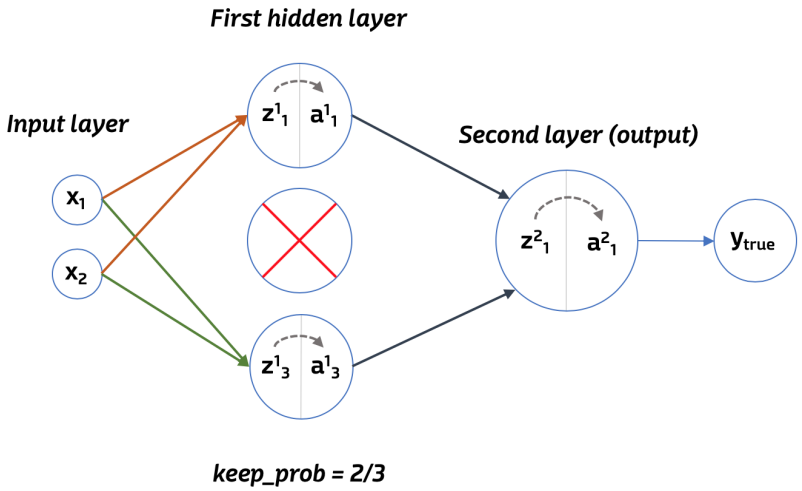

Figure 2.9: Visualisation of a network with implemented dropout on the first hidden layer; in this case, the dropout probability parameter is 2/3. The parameter is referred as "keep_prob" and corresponds to the probability of keeping a unit cell in the layer.

When implementing dropout, a dropout parameter is set for each layer in the network. This parameter defines a probability of keeping each unit cell: whether it is going to be kept or dropped out of the network. If a unit cell is excluded, then all the in-going and out-going connections are eliminated as well, and their associated weights are not being updated at the current iteration. At each iteration of the optimisation algorithm, the dropped out unit cells are re-chosen. The dropout parameter is the other hyperparameter that needs to be tuned while training the network. The most used dropout regularisation technique is an "inversed dropout". The "inverse" refers to the face that updated activation functions $a_{nl}^{[l]}$ are being adjusted by their division on the dropout parameter. The division assures that the expected value of the activation function in the layer remains the same through training.

The batch normalisation method is another common regularisation methodology. In [65], the authors proposed this method to help neural networks learn faster and in a more stable fashion. The essence of batch normalisation is normalising the activation layers by standard deviation, i.e., subtracting the mean activation (re-centring) and dividing the re-centred activations by their standard deviation value. Batch normalisation is a common regulariser and is often used in deep neural networks; it allows higher learning rates, makes neural networks less sensitive to the choice of trainable parameters' initial values and makes

the learning process "smoother" [66].

Finally, the other important regularisation technique is called "early stopping" callback. Early stopping is a callback parameter able to terminate training when a particular condition is met. When the neural network is very deep, it will (most likely) overfit on a training set. Besides, some neural networks take much time to be trained. Often, it can take not just hours, but days, weeks, or even months. In this case, early stopping could prevent overfitting when the early stopping terminates on a specific iteration (machine learning engineer must be already experienced with a network to know when it generally starts overfitting). Another condition could be an achievement of a set error on a training set. Keras [67] allows to save the best-achieved result on the training set using this callback, or stop training when the monitored quantity stopped improving. All the parameters that are related to the early-stopping setup are the hyperparameters that need to be tuned.

Adjusting the Learning Rate

The appropriate learning rate must be selected to achieve excellent model performance. Learning rate α is responsible for how large the step of parameter updates it. A common practice to set a learning rate to a specific value and train the network using it. However, on some loss surfaces, it is often beneficial to decay the learning rate (setting it to the smaller value for each training epoch or specific intervals of training epochs) while training. The rate of decay of a learning-rate is also one of the hyperparameters that need to be tuned.

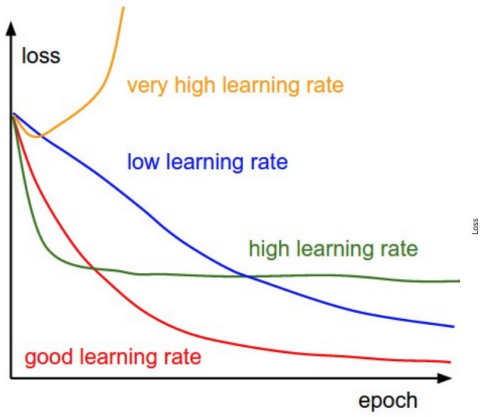

Figure 2.10: Visual presentation of how different learning rates affect learning curves [9]

Adjusting mini-batch sizes

Often a training dataset is very large and as such it is "fed" in relatively small portions to the NN during training. In deep learning, models often do not process an entire dataset at once because it does not fit into a memory slot assigned for this purpose. While training, data is split into mini-batches, and the batch-size could affect how fast and accurate optimisation would converge [68]. Instead of the decaying learning rate, one could increase the batch-size while training to improve the accuracy of a neural network.

2.2.9 Preparing and Storing Data for Machine Learning Projects

Regardless of how much data machine learning engineer has, the data alone is not enough. Feature engineering, structure and preprocessing of data plays a crucial role in successful model design. Before training, the data needs rearrangement to a shape that the network expects. Then, data needs to be normalised for training to be efficient.

Normalisation

Normalisation by standard deviation originates from the field of statistics and refers to adjusting data to bring the entire probability distributions into specific alignment.

When training, it could be problematic to feed data of values that take extremely different ranges and distributions. The network possibly is capable of fitting to heterogeneous data, but it would make optimisation more difficult. Let us take into consideration the case when training features are taking up very different values. For example a feature $x^{(1)} \in [0, 10]$ and another feature, $x^{(2)} \in [100, 10000]$. Then the corresponding parameters θ will take on very different values as well. In this case, cost function $J(\theta, b)$ will be much more complex than in the case when parameters have similar values. Thus, normalisation speeds up training. Two most common practices of normalisation are: by standard deviation (in the case of using ReLU-types activation functions), and an interval $[a, b]$. In the second case, the interval is usually $[0, 1]$ (in the case of using sigmoid activation function used in the network) or $[-1, 1]$ (in the case of using tanh activation function). The normalisation of a feature by standard deviation sets the mean of the data to zero and makes the variance equal to one.

Calculating the mean:

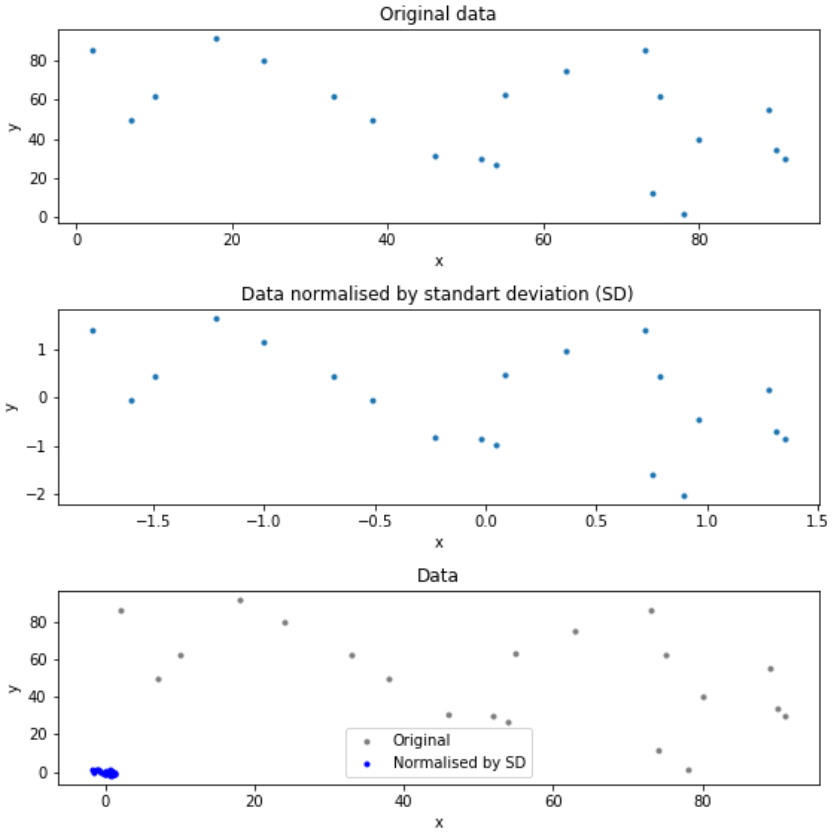

Figure 2.11: Visual comparison of original data with data normalised by standard deviation

$$\mu = \frac{1}{m} \sum_{i=1}^{m} x^{(i)} \tag{2.47}$$

Standard deviation of a discrete random variable:

$$\sigma = \sqrt{\frac{1}{m} \sum_{i=1}^{m} (x^{(i)} - \mu)^2} \tag{2.48}$$

Finally, normalisation of each feature $x^{(i)}$ by standard deviation:

$$x^{(i)}_{normed} = \frac{x^{(i)} - \mu}{\sigma} \tag{2.49}$$

To conclude, for each feature independently, one should first normalise it to have a mean of zero. Then, one should normalise it to have a standard deviation of one. Visualisation of how the normalised values change can be seen on the figure 2.11.

Normalise a feature in $[a, b]$ occurs according to a equation:

$$x^{(i)}_{normed} = (b - a)\frac{x^{(i)} - \min x^{(i)}}{\max x^{(i)} - \min x^{(i)}} + a \tag{2.50}$$

One should normalise training, development and test set in the same way, using same coefficients (μ, σ, a, or b from the examples above).

Storing Data

There are many ways to store data in scientific applications. When the available data is small in size, it is often convenient to use standard formats such as text (.txt) files, comma-separated values (".csv") files, or excel (".xlsx"). Small data files in such formats are easy to open, modify, access, and they fit in memory when working with them. However, when the data gets more substantial in size, one can hit a roadblock of fitting it into the memory and doing calculations on that data.

HDF5 technology suite [69] can be used as an efficient method of compression and storage of scientific data that is too large to store in memory. With the help of the HDF5 scientific suite, one can not only efficiently organise the data, but also dynamically write in the file, and dynamically read the data out of the file. HDF5 technology suite is a great environment to operate with substantial data matrices of high dimensions, that are typically hard to keep in memory. Working with the HDF5 data formats allows to fastly retrieve exactly the required at the moment part of the data. The data is stored in the ".h5" file format. The data format allows unlimited variations of data types, compliant and efficient I\O, and is suitable for very high volume and complex data. The data stored in this format is transferable, extensible, and can be accessed in many frameworks or by the aid of programming languages which include Python, R, FORTRAN, C++. The HDF5 technology suite also includes means for handling and inspecting data stored in the HDF5 format.

The architecture of the HDF5 scientific suite consists of HDF5 datasets (array of variables), HDF5 groups (groups of arrays of variables), and HDF5 data types. The HDF5

datasets are an essential part of the file. It consists of the data elements that are logically structured and are represented as a multidimensional array or arrays. The data arrays could be presented contiguously, as a whole sequence, or following a "chunked" option. In the second case, array elements are laid out as a collection of fixed size regular sub-arrays. Chunked HDF5 datasets may significantly improve data accessibility. However, chunk sizes must be tuned depending on the application. The HDF5 suite allows parallelism to achieve better performance for inputting and outputting data.

Apart from above-mentioned advantages, it allows very efficient data compression. This is one of the most important benefits for the research proposed in this document. The comparison of two files storing the same numerical data is the following: text data file had the size of 25.1 GB (gigabytes), and HDF5 data file had the size of 4.0 GB. Loading data time was also significantly improved: it took 18.5 minutes to access the data from text file, and only 590 μs to access data stored in HDF5 format. The numerical data stored in both files as an array with dimensions of $(100000000, 10)$.

The library addresses the most major data-related challenges. It allows efficient organisation and long-term preservation of complex scientific data, provides tools for easy access. Files are extendable and allow dynamical input and output, which is essential for the application related to this research.

Chapter 3

Literature Review

Historically, advancing in the modelling methods of deformation of solids has led to significant technological growth. Up to nowadays, advancements to these methods remain an important objective. The accurate modelling approaches are widely used in various fields of science, e.g. mechanical, structural, material, civil, or aerospace engineering, as well as in many areas of business: automotive industry, material production lines, additive manufacturing advances and other types of manufacturing processes. In particular, solid mechanics involves many challenges of the experimental and computational kind. Performing extensive physical testing can be very expensive and time-consuming. Solving mathematical and computational problems may involve computationally demanding operations such as solving sets of highly non-linear partial differential equations with numerical methods or finding inverses of large tensors. Therefore, the computational procedures may be sensitive to scalability, making some analyses unfeasible. Machine learning applications are known to be successful at such challenges. This section reviews existing approaches in material modelling and applications of machine learning computational material science and, in particular, in crystal plasticity.

3.1 Brief Review of Plasticity Modelling

Numerous approaches have been developed for modelling of deformation processes, from macro- to microscale. On the macro level, one of the approaches to describe deformation behaviour is with the use of yield functions. These functions are represented by five-dimensional surfaces in the six-dimensional space of stresses. Their projections can graphically represent these surfaces onto two- or three-dimensional spaces. When the state

of stress is inside of the yield surface, the material is considered to be undergoing elastic deformation, and, correspondingly, plastic when the state is on the surface itself. With increasing plastic deformation surface expands, so that the stress state cannot be outside of the yield surface. The Von Mises yield criterion is a very well-known yield criterion for defining plasticity of metals and alloys [70]. The criterion states that when the material's equivalent stress (represented by the equation 3.1) reaches a specific value σ_y called yield stress (which is a material parameter), plastic deformation occurs. This function was extended to many different forms, e.g. Hershey [71] yield function of non-quadratic form (equation 3.2). However, in some cases, isotropic yield criteria like the last mentioned do not meet accuracy requirements for some specific applications or materials of higher complexity (i.e. orthotropic or fully anisotropic materials).

$$\sigma_v = \sqrt{\frac{1}{2}\left[(\sigma_{11}-\sigma_{22})^2 + (\sigma_{22}-\sigma_{33})^2 + (\sigma_{33}-\sigma_{11})^2 + 6\left(\sigma_{12}^2+\sigma_{23}^2+\sigma_{31}^2\right)\right]} \tag{3.1}$$

$$\overline{\sigma}\left(\sigma_{ij}\right) = \left[\frac{(\sigma_1-\sigma_2)^m + (\sigma_2-\sigma_3)^m + (\sigma_3-\sigma_1)^m}{2}\right]^{1/m} \tag{3.2}$$

One of the examples of more complex yield functions was recently developed by Yanshan Lou and Jeong Whan Yoon [72]: they presented anisotropic yield function based on stress invariants for BCC and FCC metals. They extended Drucker [73] yield function into anisotropy based on the fourth-order linear transformation tensor. Apart from accuracy benefits, the contribution of the work was in presenting a "user-friendly" and easy-to-implement yield function. However, computational methods in deformation modelling of this sort could include convoluted operations like the inverse of the the fourth-order tensors. The yield function is constructed as a sum of several components. Numerical coefficients in those components are specific to a material, and applications of such convoluted yield functions are limited and require calibration for other materials.

Phenomenological plasticity models are typically not able to capture physics-based phenomena such as crystallographic slip and do not take into account such important material characterisation as microstructure. The microstructure can influence deformation behavior and the occurrence of strain localisation in specific parts of an investigated materials [74, 75]. The microstructure can be characterised by the constituent grains, their texture, size and morphology [76, 77], as well as a presence of precipitations [78, 79, 80] and multiple phases within material structure [81, 82, 83]. These aspects can play an important factor in material deformation behavior. Crystal plasticity can be applied to take into account those material phenomena in deformation modelling.

Crystal plasticity theories are developed to predict the anisotropic stress-strain response and the corresponding change in polycrystalline texture at finite elastic and plastic strains. Polycrystal deformation models are usually derived from single-crystal deformation models. Homogenisation schemes have been used to establish the link between the deformation of polycrystalline material and its constituent grains (single crystals). Sach's [84] homogenisation approach is one of the earliest approaches proposed (1928), and it assumes that each grain of a polycrystal undergoes the same stress. In contrast, the Taylor-type homogenisation scheme [85] assumes homogeneous strain in each constituent grain and the macroscopic stress is calculated as an average stress response in all grains. However, homogenisation approaches present simplifying stress or strain homogeneity hypotheses and hence cannot consider the complex intergranular relationship and their consequent effects on stress and strain partitionings within a material. Crystal plasticity finite element methods are able to take into account grain size, morphology and neighbouring effects by discretising the analysed sample into finite elements and therefore achieve even higher fidelity of material behaviour predictions. The CPFEM models enable predictions for complex internal (which are imposed by interactions between grains due to physics of grain mechanics) and external boundary conditions [86], which may be critical for virtual material design. CPFEM models can be built upon different constitutive formulations, allowing applications for different materials. With the complexity involved, crystal plasticity is a state-of-art method that can help design advanced materials with the exact desired properties for any field of application, however, its high-fidelity comes at a price of high computational cost.

3.2 Acceleration Methods of Crystal Plasticity Calculations

Multiple efforts has been made to accelerate crystal plasticity calculations. One of the earliest, Bunge-Esling approach (1984) [87] used a specific analytical expression for orientation distribution function (ODF). They used the assumption that texture formation process can be described as a flow or compressible medium in the Euler space (Clement and Coulomb [88] originally proposed this assumption). Bunge and Esling's contribution was in a representation ODF in the form of a series of harmonic functions of orientation. This approach helped to accelerate the computations of the flow field in the Euler space. A general methodology based on Bunge's conservation of texture volume was later developed and was used to predict texture evolution which showed good predictions for small strains as compared with predictions generated by Taylor-type crystal plasticity [89]. The study presented a novel approach to establish a relationship between the rate of change of the

Fourier coefficients of texture and the initial texture, and was validated for FCC materials with texture. Later this approach was extended to prediction of stresses, lattice spins and strain hardening rates for individual crystals [90]. It was suggested that the functions of predicted properties could be represented using Fourier coefficients and generalised spherical harmonics and be dependent on the crystal orientation. The presented possibility to reproduce the results of first-order crystal plasticity simulation efficiently resulted in reduction of computational time. The improved version of this model was also developed and involved local spectral interpolation method using discrete Fourier transform (DFT). The essence of this method was in storing the variables on a uniform grid in orientation space in a database. Then, application of DFT interpolation on that database allowed recovering the values of interest. This approach was demonstrated to speed up the crystal plasticity calculations for FCC metals compared to conventional crystal plasticity calculations as well as generalised spherical harmonics [91].

The other database approaches were employed to accelerate crystal plasticity computations. Material knowledge systems (MKS) was formulated to construct accurate bidirectional "linkages" designed to predict desired property or response. MKS is extensively discussed in the series of works by Kalidindi and his co-authors [92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108], and finds its applications for various materials and cases of mechanical processing of those materials. The presented approach allows to accurately capture critical information related to the microstructure evolution of material in spatiotemporal multi-scale simulations. There are existent MKS homogenisation and localisation workflows. For homogenisation workflow, MKS database is established based on the pre-generated synthetic dataset of representative microstructures. Then mechanical response for each microstructure is obtained with the help of an established numerical model. These steps yield the formation of structure-property linkages. The structure-property linkages are the mappings that allow predicting a property of interest based on a given structure. Linkages are expressed as meta-models or surrogate models. The examples of other types of linkages include process-structure-property (PSP) linkages, structure-response linkages [94], linkages for inelastic effective properties [95]. The information encapsulated in MKS allows obtaining such desired information as microstructure-sensitive predictions. For example, PSP relations, or microstructure-response linkages (stress-strain relationships, microstructure evolution). The properties and characteristics of interest are predicted orders of magnitudes faster compared to conventional simulation methods while preserving comparable accuracy.

3.3 Machine Learning Applications in Material Science and Mechanics of Solids

Machine learning (ML) based approaches can be fast once trained and become increasingly accessible. One of the first efforts in ML-based material modelling was proposed by Ghaboussi et al. in 1991 [109]. The authors suggested using artificial neural networks as an alternative to mathematical material behaviour models and have put the beginning to the applications of machine learning in computational material science. In their work, they presented a two-hidden-layers ANN which was used to model concrete deformation behavior. The network was trained on the experimental results. The study focused on two particular applications for path-dependant material behaviour: stress-controlled and strain-controlled model. In the stress-controlled model, the inputs were the current state of stress and strain and a stress increment; the outputs were the strain increments. The network was used incrementally, starting at a known stress-strain state. For the strain-controlled model the same data was used. Similarly, the stress increments were used as an output of the model. Due to the incremental approach of the application, the predictions of the network have shown deviation, but the trends of the network predictions were approximately correct.

One of the recent (2019) studies demonstrated the prediction of path-dependent material response in terms of stresses and plastic energy for two-dimensional representative volume elements (RVEs) with a sequence-learning model [10]. The authors proposed a recurrent neural network (RNN) for two applications in plasticity modelling. The first study was performed for an RVE with a curved inclusion. The inputs to the RNN model included representative volume element descriptors and temporal deformation paths. The deformation path were sampled as shown in figure 3.1. The outputs were the temporal stresses and plastic energy values over 100 increments, as well as yield surface evolution. The model was able to successfully capture mechanical response for an unseen test case scenarios, as shown in figure 3.2. In the second case study, the RNN model was able to accurately predict stress and plastic energy for a class of composites. This study have shown a capability of neural networks to predict complex path-dependant behaviour.

Machine learning methods find their applications in mechanics of solids to predict important mechanical features related to a material of interest. One of the works suggests that instead of calibrating a yield function the material hardening law could be predicted using an ML model [110]. The authors developed a hybrid experimental-numerical approach to model temperature- and rate-dependent plasticity for fracture analysis. In their work, the ML-based approach captured a hardening law that allowed to present the yield

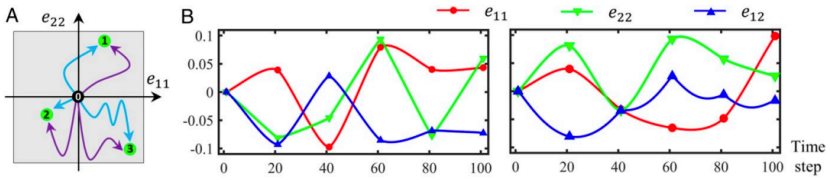

Figure 3.1: "Sampling the temporally varying loads. (A) Three end states are marked in the strain space spanned by e_{11} and e_{22}. For each end state, 2 deformation paths that connect it to the origin are illustrated. The gray area indicates the range of each strain component. (B) Two examples indicating the temporal evolution of the 3 strain components that, collectively, determine the deformation path to an en state. The markers on each path indicate the control points used in interpolation. Paths in B are not related to A." [10]

.

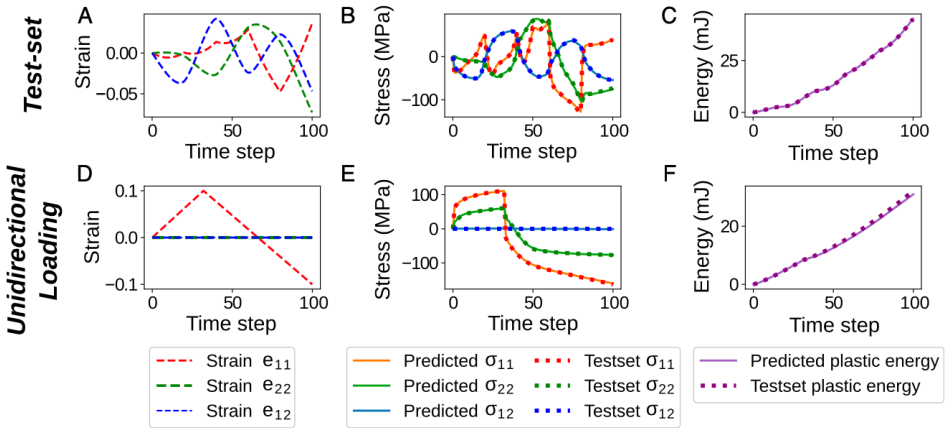

Figure 3.2: "Evaluation results for the trained model in case 1 . The top row demonstrates the applied average strains (A), the predicted and database average stresses (B), and the predicted and database plastic energies (C) for a test-set sample (unseen in the training process). The bottom row depicts the average strains (D), average stresses (E), and plastic energies (F) for the unidirectional loading test." [10].

.

stress as a function of the equivalent plastic strain, strain rate, and temperature. Apart from the computational benefits of this method, there was no need to look through the conventional hardening law forms and hence model bias was avoided prior to calibration of a model. The benefit of such model embedding is in the ability to apply a similar approach in other models by using transfer-learning techniques and avoid time-consuming calibra-

tion of a yield function. Another yield-function application used several different ANNs to replicate the predictions of anisotropic Yld2000-2d model for various loading conditions [111]. Excellent predictive capabilities of feed-forward neural networks and gated-recurrent units network for monotonic and non-monotonic loadings were demonstrated in stress-state predictions based on the applied strain.

The other ML application was related to mechanical features predictions [112]. ML-based model, the "deep material network", was developed to predict the fourth-order compliance tensor related to a two-dimensional RVE of a specific material. The network was trained based on the results obtained from offline direct numerical simulation (DNS) results for four types of heterogeneous materials: uniform, matrix inclusion, amorphous, and multi-phase anisotropic materials. The trained model served as a multi-scale material modelling method. It accurately predicted non-linear material loading-unloading behaviour for uniaxial and complex loading-unloading paths. The designed model was built of "mechanistic building blocks" which accounted for relevant physics. Besides the computational benefit indicated by the authors, the model was also valid for unknown materials and loadings and hence was applicable to a wide range of applications. ANNs were demonstrated to predict cold rolling texture of steels [113]. A neural network model was used to predict fiber texture evolution of steel during cold rolling. The input of texture data: fiber texture intensities, carbon content, carbide size, and amount of rolling reduction. The model was trained on a limited dataset of texture data of three different steels with varying carbon content and carbide dispersion. The work shows the importance of choosing the informative dataset, as the predictions within training set were accurate and the test set prediction showed deviation. The main conclusion from this work is that the training set should be representative of the data regime within chosen application.

A pattern recognition utility of ML is extendable to a generative capability. For example, new materials' discovery becomes possible. In the work on prediction molecular atomisation energies, machine learning displayed the speed and effectiveness of ML in *ab initio* calculations of quantum-chemical analysis of compounds [114]. A conventional approach for such computations could take hours or days, while the ML-based model can give the results online. Atomisation energy is a very important feature as it can be measured experimentally and also determines the molecular stability with respect to its atoms. Hence, the conclusion can be drawn: given a large chemical compound space in a training set, the ML model will not only save valuable computation time for known compounds, but also brings the analysis possibility of unknown ones. There are many other applications of ML, when essential mechanistic and atomic features are predicted fast and accurately. Next two paragraphs will feature some notable results. In the recent study, [115] demonstrated an ML framework utilised to predict the evolution of local strain distribution,

plastic anisotropy and failure during tensile deformation of 3D-printed aluminium alloy.

A lattice constant (LC) is an important characteristic that helps to determine material properties and, in some cases, it is very important for taking into account in production processes. Research has shown that ML methods can be used to predict this parameter for novel materials accurately. A Support Vector Machine model was demonstrated to achieve accurate and fast lattice constant predictions for a specific crystalline material (orthorhombic perovskites). The predictions are based on five atomic parameters [116]. In a similar research work, an ANN, generalised regression neural network (GRNN) and support vector regression (SVR) were employed to predict LCs for cubic and monoclinic perovskites accurately [117]. ML predictions were compared with the results that were obtained with conventional methods of calculation LC, such as density functional theory-based model and chemical stoichiometry based prediction tool. This work was extended to other types of perovskites, and the improved the performance of computational intelligence models was demonstrated. The presented model outperformed the relatively computationally cheap existing tools for LC prediction. The developed ML models were able to accurately predict LC for the unknown compounds of the same type that the model was trained on. The lattice constants were predicted based on the sizes of ionic radii of the compounds, electronegativity of the specific atoms, and oxidation state. Apart from the computational speed (in comparison to density functional theory and other methods), there were also other benefits. Based on the error analysis, it was possible to estimate the lattice distortion level for structurally known compounds. It was seen that some ML methods performed better than others in some cases. For a small feature set, it was shown that the methods of SVR and random forests were more accurate on the test set than linear regression methods or GRNN [118]. In general, it is a common practice to compare different methods' performance to see perform better for a given problem. It has been shown that the ML method could offer to the research not only a computational speed but also the better accuracy compared to some other methods and provide aid for a new compound design. Such methods could be transferred to a new material design applications which is vital in mechanical engineering.

The following research works are focused on predicting characteristics of materials and compounds. Machine learning methods were applied to a derivation of localisation relationships for elastic deformation in a high-contrast two-phase composite material [119]. In their work, the authors trained ML model for predictions of a localisation-structure-property relationship. As an input, a 3D microstructure and a loading condition were used, and a corresponding elastic strain field throughout was obtained as an output. Localisation methods, in contrast to homogenisation methods, are usually limited, and this work highlights the capability of ML to achieve accurate prediction results even in such

complicated applications. The improved version of the model was presented [120]: it outperformed previous best results and the possibility to perform parallel computations was introduced.

All the presented above works body forth a tremendous asset not only for chemical and pharmaceutical industries but for the industries reliant on advanced high-performance materials. Crystal plasticity links macroscopic performance of the materials with their microscopic properties, which can significantly assist in new material design.

3.4 Significance of Data and Feature Engineering in Machine Learning Applications

One of the fundamental aspects of training a reliable model is creating a dataset that is enough for a solution to a given problem. This question was considered by Bessa et al. [121]. The work addresses the "curse of dimensionality", the phenomenon that occurs during data analysis in high dimensional space. This phenomenon is encountered in fields of numerical analysis, sampling, combinatorics, and machine learning. With the increase of dimensionality of the feature space, the volume of this feature space increases too fast, and the available data becomes very scattered. This data sparsity phenomenon is problematic in terms of statistical significance which ML methods heavily account for. To counter the curse of dimensionality, Bessa and co-workers developed a framework for a data-driven analysis in design (e.g. predicting global optimum for a material design) and modelling (e.g. alternative constitutive modelling). In their work a three-step approach was proposed: (1) Design of experiment (DoE): taking into account material geometry, properties, and external conditions; (2) performing an efficient computational analysis of each sample, yielding a material response database; and finally, (3) ML methods applied to a database to obtain the new optimal design or response model of interest. These are the logical steps in a data-driven analysis applied in material science. DoE refers to the work that results in the characterisation of the material and conditional features. These features include microstructure, material properties, boundary conditions, etc. In the field of applied ML this process is usually referred as "feature engineering" [122, 123, 124, 125] and describes a process of selection the most descriptive features from the feature space. The feature space is formed by all available descriptors for a given problem and is limited by the ranges of those descriptors. The skills and knowledge in the application field are advantageous in the process of selecting the most informative descriptors, as well as when assessing the reliability of the model. Bessa and co-workers provided insightful examples of the process of feature engineering in applications to material modelling and phenomenological modelling.

Apart from selecting the most helpful descriptors, it is required to define the domain of each feature, and by how many examples should each feature be represented. The feature space can be sampled either using the grid approach or random sampling. However, grid sampling can cause having unfeasible number of samples in a dataset as the samples quantity will grow exponentially with increase of feature space dimensions. Random sampling allows choosing the amount of samples, but can result in uneven coverage of the feature space. Selecting a strategy for an optimal filling of a feature domain could be beneficial for training an optimal model.

The optimal filling of a feature domain can be achieved using Sobol sequences method [126, 127]. Sobol sequence method, also known as LP_τ sequences, may be advantageous for filling the feature domain efficiently. The motivation behind utilisation of the Sobol sequence is to generate a sequence (set of numerical points) that is "well-spaced" in the s-dimensional unit cube $I^s = [0,1]^s$, such that:

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} f\left(x_i\right) = \int_{I^s} f, \tag{3.3}$$

where f is some real integrable function over I^s. The Sobol sequence generation algorithm generates data in the range $[0,1]$. The benefit of Sobol sequences in its low-discrepancy meaning that the points in the sequence are almost equidistributed. This fact ensures that the domain is filled approximately uniformly but yet (quasi-) randomly, henceforth the better diversity of examples (rather than yielded by feature domain "gridding") delivers increased diversity of information. Figure 3.3 presents the comparison for the sampling of 343 points in three-dimensional space using: grid sampling, random sampling, and sampling with Sobol sequences. For better comparison, all the sampled points were projected onto $x-y$, $y-z$, and $x-z$ planes. Grid sampling is an excellent method to sample data in two-dimensional space. However, the number of samples in a dataset will grow exponentially with the increase of dimension. In addition, with the grid sampling method, the data is sampled in a specific pattern, which could potentially interfere with a model's learning performance. Random sampling allows choosing the number of samples; however, it may form clusters of sample points within a sampled space. Alternatively, some regions of a sampled space could be not included during sampling. Clusters of points could result in overfitting the model while lacking data points in specific ranges and (or) combinations could result in a model's poor generalisation capability. Sobol sequence allows forming a robust dataset that almost uniformly covers all the regions of a sampled space and does not result in clustering while sampling.

The size of the used dataset also plays an essential role in training a machine learning

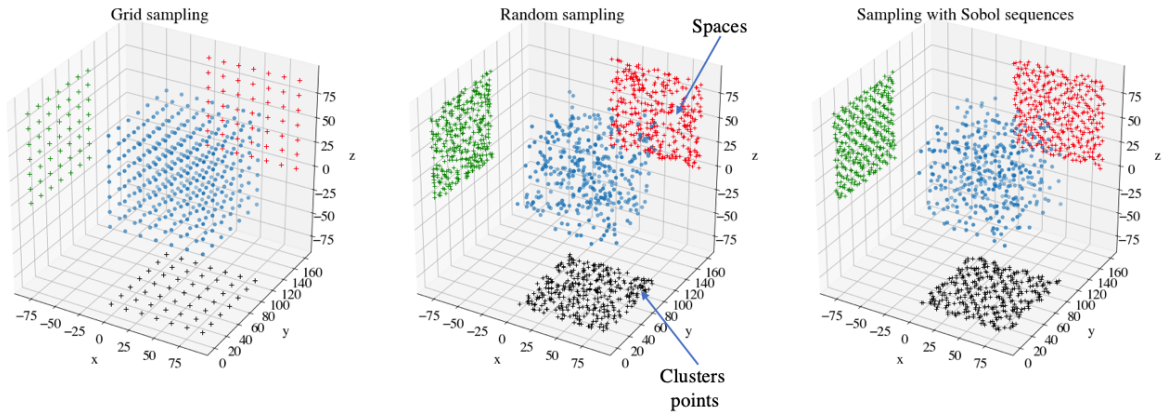

Figure 3.3: Visual comparison of 343 data points sampled with: sampled with grid sampling, random sampling, and Sobol sequence sampling. All the sampled points were projected onto $x-y$, $y-z$, and $x-z$ planes. Sobol sequence allows for more optimal space filling compared to two other methods.

.

model, and it often is not feasible to create a sizeable dataset in material science. Therefore, there is also a need for a strategy for machine learning applications with small datasets. The datasets in material science are known to be relatively small and diverse simultaneously. In [128], have addressed the problem of having small datasets in material science and presented a study on relating the degree of freedom (DoF) of model and the precision of prediction. They have discovered that precision increases at the cost of a higher degree of freedom of a model. In their work, they associate this relationship with the effect of the bias-variance trade-off, the concept of which is closely related to the problem of underfitting and overfitting. When countering this trade-off in ML modelling, it is necessary to tune the model (adjust the parameters of the model) such that it: (1) accurately captures the irregularities of a training data; and (2) generalises (interpolates or, sometimes, extrapolates) well on unseen data. Reducing both bias and variance is a common challenge in designing an ML model based on a small dataset. One of the most decisive actions that could be done is suitable dataset input-output design (this process is called "feature engineering"). Adding more inputs for the machine machine learning model, i.e. increasing its DoF, can lead to better predictability. However, according to the authors, this could potentially lead to a "highly complex model difficult to interpret the embedded physics" and increased cost due to additional experiments. Therefore, increasing the DoF makes it harder to construct an accurate ML model due to the curse of dimensionality, as discussed

earlier in this subsection.

To counter the curse of dimensionality phenomenon, it has been suggested adding crude estimation of property (CEP) to a feature space when training an ML model [128]. CEP is defined as a prediction of a targeted property which is obtained by other methods than an ML model. This prediction is often less accurate, and the methods used should require zero or near-zero computational efforts. These methods could be non-expensive experimental measurements, empirical models, or other estimation methods. In addition, for small datasets, it would be beneficial to perform k-fold cross-validation (CV) technique. One of the examples presented by the authors was the prediction of the band-gap of binary semiconductors. In this case, CEP was band gap simulated at generalised gradient approximation (GGA) level, which is cheap to compute. The model with CEP had a 33% reduction in error in comparison to the model without CEP. The performance of the ML model was benchmarked by comparison with a corresponding GW approximation based on 49 compounds. The ML model showed a smaller error, demonstrating excellent predictive capability. Another CEP application study have been presented in study which focused on estimation of a band-gap predictions [129]. In their work, an additional feature of band-gap CEP was used. The authors implemented a computationally reasonable estimation of a band-gap using Perdew-Burke-Ernzerhof exchange-correlation functional [130]. Other efforts to improve learning capabilities when only a small dataset is available include data augmentations techniques. In [131], the authors implemented cropping and re-scaling of the input image data of strain profiles for creating more samples for training and validation of a deep learning model. In addition to data augmentation methods, the transfer learning can be used for re-training existing models on the limited datasets [132, 133].

To conclude, datasets play an essential role in training accurate machine learning models. The methods such as adding a crude estimate for the target, data augmentation, using transfer learning, and employing efficient data sampling techniques can increase the learning capability during training. Having an abundant dataset with an employed optimal sampling methodology for its features is generally favourable for training an accurate machine learning model.

3.5 Machine Learning Applications in Crystal Plasticity

Numerous studies have demonstrated machine learning applicability in computational material science and conventional phenomenological plasticity. The research has also been focused on crystal plasticity applications, as covered in this subchapter.

In [134, 135], the authors used classification ML methods to investigate the formation of stress hotspots in polycrystalline materials under uniaxial tension. The full-field CP based deformation model was integrated to gain data-driven insights about microstructural properties. In these works, they focused on FCC [134] and HCP [135] materials. A synthesised dataset of three-dimensional polycrystalline microstructures under uniaxial tension, including microstructural descriptors for each grain, and the stress and strain fields resulting from CP simulations, were provided by the researchers [136]. The microstructures were generated using six representative textures for FCC materials and eight representative textures for HCP materials. A classification random forests algorithm was employed to predict the stress-"hot" grains given microstructural and geometrical descriptors such as orientation distribution function, Schmid factor, misorientations, grain shape, and grain boundary types. As a result, an area under the receiving operating characteristic curve score of 0.74 was achieved for FCC materials and 0.82 (equal CRSS cases) and 0.81 (unequal CRSS cases) for HCP materials.

The regression model case studies have been presented for crystal plasticity applications. In [137], an ANN-based framework to predict the flow response and texture evolution of polycrystals subjected to multiaxial and non-proportional loading was proposed. In their work, authors accurately predicted the constitutive response of AA6063-T6 alloy under uniaxial tension followed by simple shear, which demonstrated the capability of neural networks to learn polycrystalline material behaviour under various loading conditions. Other crystal plasticity applications included: linear regression and neural networks applied to predict a cyclic stress-strain relation for steels [138], sequence learning methods were applied to predict microstructural texture evolution for an FCC copper sample under uniaxial tension [139], ANNs applied to predict the relation between CP parameters and the microstructural texture for a titanium alloy under uniaxial tension [140], a classification ML algorithm using spatial strain correlation to infer and classify prior deformation history in thin aluminium films [141]. In [142], the authors used ANNs to predict flow behaviour and final deformation texture. The predictions could be achieved for four loading conditions (compression, tension, shear, rolling), twelve initial textures, loading rate, and a range of Voce hardening parameters. These works have demonstrated the capabilities of machine learning to predict complex material behaviour, with the following limitation: the models performing well within the "bounding box" of the dataset available for training. Furthermore, the models designed for particular materials under specific loading conditions have restricted applications. Available training datasets are typically split into training and test constituent parts, and while it is crucial to validate a model's performance on a test set, it is also vital to validate the model for real applications.

A limited number of studies show applications of machine learning in crystal plasticity

to produce finite element (FE) modelling simulation results. These studies adopt convolutional neural networks (CNNs) as they are instrumental in making predictions based on two- or three-dimensional input values, and (or) making two- or three-dimensional predictions based on those input values. In [131], CNN with residual connections was implemented as an alternative method of graphical strain analysis for CP predictions. For the dataset, they performed discrete dislocation dynamics simulations of metal samples with elastic properties similar to aluminium. In the performed simulations, the specimens were subjected to uniaxial compression with varying loading orientation at small deformation levels (under 1%). Each sample had a width ranging from 62.5 nm to 2 μm. The samples were characterised by low, medium, and high initial prior strain deformation levels. Based on the strain images as an input to the CNN, the authors were able to classify the initial strain deformation level of a sample based on three classes ("low", "medium", or "high" strain). Using the hidden layers of the proposed convolutional network, the authors computed the dislocation density measure of a crystal, which allowed to identify a crystal's strength and, therefore, find the statistical prediction of a sample's stress-strain curve. In their work, the authors used data augmentation techniques on the dataset consisting of 20 simulations for providing enough data for training an accurate model.

The other works analysed the relationships between microstructure and material properties and behaviour. In [143], the authors used three-dimensional convolutional neural networks (3D-CNN) to relate a material's microstructure to material properties of interest. The proposed model analysed relations between microstructure and the material's elastic property. The input to the model was the $51 \times 51 \times 51$ voxel microstructure of two-phase high-contrast composite material, and the output of the model was the effective elastic property in the form of a C_{11}^{*} component of the effective elastic stiffness tensor. The authors used the dataset of 5900 microstructures with varying volume fractions of the constituent thermodynamic phases. The outputs for the dataset were obtained with FEM simulations. Their work has demonstrated that the effective properties can be predicted using 3D-CNN with dramatically increased computational speed and nearly no sacrifice in accuracy.

In [11], the authors used two-dimensional convolutional neural networks for establishing microstructure-property linkages in a material. In their work, they related a dual-phase microstructure to a stress value at a strain close to the yield point ($\varepsilon_{11} \approx 1 \times 10^{-3}$). This target quantity was chosen as it is similar to the yield stress. For the dataset, microstructures with soft and hard phases were generated. Volume fractions of each phase varied from 5 to 95 per cent. The uniaxial tension simulations were performed for $4,750$ samples, which were used to train and validate the CNN model. The trained model demonstrated excellent predictive capabilities and a dramatically improved computational time perfor-

mance. In addition, the authors compared the local stress response of polycrystal to the visualised features of the trained network. The visualised features were obtained using specialised software that highlights the important regions in the input image [144]. Some degree of correlation between CNN features visualisation and local stress hot-spots of the deformed microstructure was observed (figure 3.4), which confirms that CNNs can capture microstructural features that are important for local stress and strain predictions.

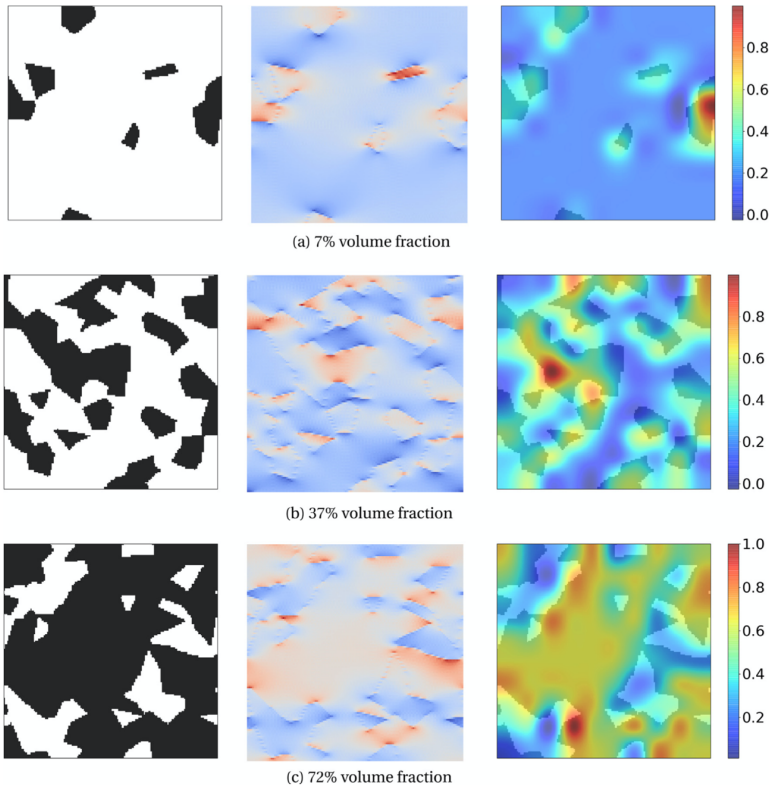

Figure 3.4: "Stress localization as predicted by crystal plasticity simulations and predicted by CNN model. In each set, the image on the left shows initial microstructure, the image in the center shows stress localization predicted by crystal plasticity model and the image on the right shows stress heatmap predicted by CNN model." [11]

.

In [145], the authors utilised a hybrid of convolutional and recurrent neural networks for application in oligocrystals of an annealed austenitic stainless steel. Their work utilised the initial microstructure and the strain values as an input to the machine learning model to predict the resultant stress corresponding to the input strain, up to 6 per cent strain. The

model was trained based on the dataset consisting of $6,630$ oligocrystalline microstructures with dimensions of $20 \times 20 \times 20$. Crystal plasticity simulations for the dataset were performed for a quasi-static uniaxial loading condition. The resultant neural network demonstrated excellent predictive capabilities for stress prediction during the elastic regime, and the prediction accuracy degraded as the plastic flow occurred. A year later, the work was extended to predict the local evolution of the dominant stress component for two-dimensional oligocrystalline microstructures [12]. The convolutional and recurrent neural networks hybrid was trained based on the dataset of $16,000$ samples. The samples were created using quasi-static uniaxial tension crystal plasticity simulations up to 0.3% strain. The input to the network was two-dimensional 32×32 oligocrystalline microstructure and a strain value. The grains within each microstructure were characterised with an orientation angle $\phi \in [0, 0.5\pi]$. The output was a stress partitioning corresponding to the input strain value. The authors have demonstrated the capability of the CNNs to predict stress partitioning accurately; figure 3.5 shows an example for a median prediction for a test set. The CNN demonstrated good agreement with the predictions of CP simulaitons for oligocrystals.

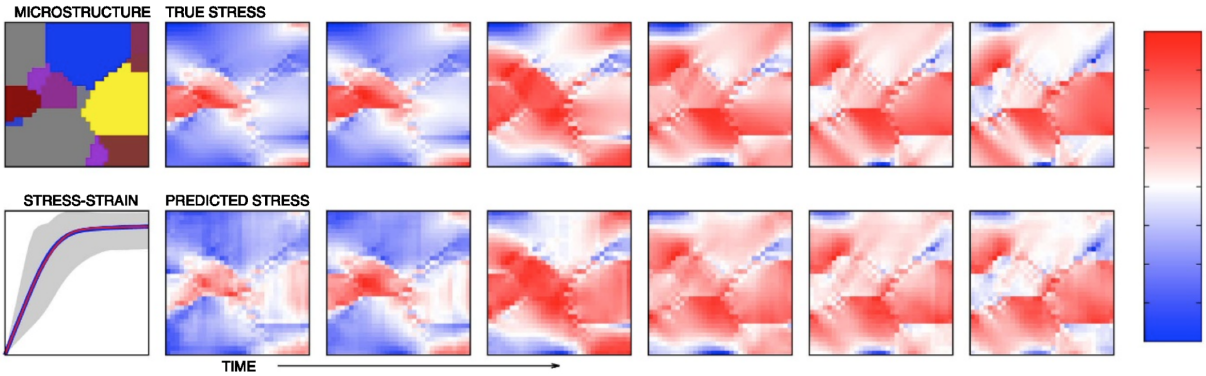

Figure 3.5: "Evolution of stress for three realizations (a, b, c) corresponding to the median total root mean square pixel-wise error. Initial grain structure (left upper panel, using the same color scale as figure 1 where gray corresponds to an orientation of 0 and $\pi/2$, and yellow is $\pi/4$), comparison of the oligocrystal average stress-strain response (left lower panel, CP: blue, NN: red, min-to-max CP range of the stress field: gray), and stress field evolution (true/CP: upper right panels, predicted/NN: lower right panels) at uniformly sampled strains $\varepsilon \in \{0.02, 0.06, 0.10, 0.14, 0.18, 0.22\}\%$ are shown. Note that the blue-white-red color scale range of the true and predicted stress field at a fixed strain is adjusted to the minimum and maximum of the true CP stress to show contrast." [12]

.

As seen in the literature, the capability of machine learning applications, particularly convolutional neural networks and their hybrid models, is yet to be investigated in their full-field predictions capabilities. The overall summary of deficiencies in literature is discussed in section 3.6 of this thesis.

3.6 Summary of Deficiencies in Literature

Numerous studies have demonstrated various capabilities of machine learning models in applications for crystal plasticity. The studies discussed in the section 3.5 presented applications to specific crystalline materials; however, they have only been limited to these materials and the loading conditions that they were trained on [137, 138, 139, 140, 142]. While the computational advantage of ML models have been clearly demonstrated for many applications, no comprehensive model was demonstrated for high-fidelity predictions for an exhaustive range of materials under any imposed loading condition. It is important to have a comprehensive model to make it usable for real applications. Such a model or framework must meet the following requirements to predict material behaviour for:

- An arbitrary texture of a crystal and must not be limited to a number of specific textures;
- Materials with different parameters (such as hardening parameters);
- An arbitrary strain path and must not be limited to a number of specific loading conditions;
- Large strains;
- All stress and strain tensor components evolution, as well as for texture evolution;
- The cases that fall outside of the distribution of the dataset used for training and validation, i.e. the extrapolation capabilities of the model or framework must be validated.

Another literature gap is related to the microstructural effects on the behaviour of materials. Crystal plasticity models are widely used to describe these effects; however, crystal plasticity finite element computations can be very computationally demanding [80]. Convolutional neural networks can be instrumental in making fast deformation behaviour

predictions for microstructures, as they are suitable for two- and three-dimension inputs. The research in applications of CNN to predict a deformation behaviour of a material with a specific microstructure is very limited. For dual-phase contrast materials, the applications included: in [143], microstructures were related to a component of the effective elastic stiffness tensor; in [11], microstructures were related to stress at a minor strain value. Stress partitioning evolution prediction was demonstrated only for oligocrystals with the predictions up to 0.22% of ε_{11} strain [12]. Machine learning models have not been demonstrated to predict evolution for local stresses and strains, for all stress and strain tensor components, of polycrystalline microstructures. For some materials, the number of grains within microstructure could be large for the microstructure to be representative of the material; e.g. a microstructure with over 200 grains is required to present sufficient microstructural and textural characteristics of an AA5457DC sheet [41]. Furthermore, a suitable strategy for data generation should be presented, such that the resulting model is capable of making predictions for real materials. This thesis addresses the identified gaps in research.

Chapter 4

Research Overview

4.1 Research strategy

The research work has been divined into two major parts as discussed in section 1.2. This section present strategies to address each of the set goal, as outlined next.

4.1.1 Development of a machine learning based framework to predict stress-strain behaviour and texture evolution for FCC single crystal and polycrystalline materials.

The goal of this work is to develop a machine learning-based framework that enables rapid and high-fidelity predictions for stress-strain behaviour and texture evolution. The framework must be applicable for a wide range of materials characterised by texture (crystal orientation) and hardening parameters. The machine learning model selection is performed. The Taylor-type crystal plasticity model simulations act as a learning base for the machine learning model within the framework. The input and output features of the model and their corresponding sampling strategies are selected. The data processing methods are introduced. The proposed model must capture accurate material behaviour under non-monotonic strain paths. Since the number of possible non-monotonic strain paths is not limited, an emphasis is put on the dataset design. To achieve accurate stress, strain and texture evolution predictions for non-monotonic loadings, the proposed framework incorporates a crystal plasticity update algorithm. The resultant framework is validated against crystal plasticity simulations for several monotonic and non-monotonic loading

for a number of different materials. The importance of this work is also highlighted by its feasibility: the framework can predict complex strain paths without having to train machine learning models on the infinite set of possible non-monotonic loading scenarios. The runtime comparison of the proposed framework with the Taylor-type crystal plasticity model is performed. This part presents a macro-level model that allows predictions for single and polycrystals.

4.1.2 Development of a machine learning based framework for prediction of local stress and strain evolution predictions for FCC polycrystalline materials.

The second part of the research is a micro-level model that allows predicting local stresses and strains for a given microstructure. The goal of this work is to develop a machine learning-based framework that enables rapid and high-fidelity predictions for local stress-strain behaviour. The framework must be applicable for a wide range of materials characterised by a microstructure (a number of grains with crystal orientations) and hardening parameters. The machine learning model selection for this application is performed. The finite element crystal plasticity model simulations act as a learning base for the machine learning model within the framework. The input and output features of the model and their corresponding sampling strategies are selected. The data processing methods are introduced. The proposed model must capture accurate material behaviour for various microstructures. To achieve that, the model was trained on crystal plasticity finite element model simulation for synthetically generated microstructures. The methodology behind microstructure generation is presented. The resultant model is first validated against crystal plasticity finite element simulations for the completely new (unseen during training) synthesised microstructures. To demonstrate the flexibility of the resultant model, it was validated against crystal plasticity finite element simulations for aluminium alloy microstructures. The runtime comparison of the proposed framework with the crystal plasticity finite element model is performed.

4.2 Summary of contributions

The following two chapters of this thesis present the work carried out to meet the research objectives. Chapter 5 is based on the following published manuscript:

Ibragimova, O., Brahme, A., Muhammad, W., Lévesque, J., & Inal, K. (2021). A new ANN based crystal plasticity model for FCC materials and its application to non-monotonic strain paths. International Journal of Plasticity, 144, 103059.

The authorship contribution statement for this work is following:

Olga Ibragimova: Conceptualisation, Investigation, Methodology, Formal analysis, Software, Visualisation, Writing - original draft.

Abhijit Brahme: Conceptualisation, Investigation, Methodology, Supervision, Writing - review & editing.

Waqas Muhammad: Conceptualisation, Investigation, Methodology, Supervision, Writing - review & editing.

Julie Lévesque: Supervision, Project administration, Funding acquisition.

Kaan Inal: Supervision, Project administration, Funding acquisition.

Chapter 6 is based on the following submitted manuscript:

Ibragimova, O., Brahme, A., Muhammad, W., Connolly, D. S., Lévesque, J., & Inal, K. (2022). A new CNN based crystal plasticity finite element framework and its application to polycrystalline metals. Under Review at International Journal of Plasticity.

The authorship contribution statement for this work is following:

Olga Ibragimova: Conceptualisation, Investigation, Methodology, Formal analysis, Software, Visualisation, Writing - original draft.

Abhijit Brahme: Conceptualisation, Methodology, Supervision, Writing - review & editing.

Waqas Muhammad: Conceptualisation, Investigation, Methodology, Supervision, Writing - review & editing.

Daniel Connolly: Conceptualisation, Methodology.

Lévesque: Supervision, Project administration, Funding acquisition.

Kaan Inal: Supervision, Project administration, Funding acquisition.

In addition to these applications, the research conducted herein also facilitated the following publications. These published works do not have a direct impact on the overall research objectives of this thesis and are referenced only as supporting work:

- Muhammad, W., Brahme, A. P., **Ibragimova, O.**, Kang, J., & Inal, K. (2021). A machine learning framework to predict local strain distribution and the evolution of plastic anisotropy & fracture in additively manufactured alloys. International Journal of Plasticity, 136, 102867.

Chapter 5

Artificial Neural Networks-based Crystal Plasticity Model for FCC Materials and its Application to Non-monotonic Strain Paths

In this chapter, a machine learning framework to accurately predict stress-strain behaviour and texture evolution for complex strain paths is developed. The proposed framework is composed of an ensemble of artificial neural networks and a CP-update algorithm to allow accurate predictions of material behaviour for complex strain-paths. The framework was successfully implemented and validated against the Taylor-type CP model for monotonic and non-monotonic strain paths for single crystal and polycrystalline aggregates. The chapter presents an accepted manuscript of an article published in International Journal of Plasticity [1].

[1] International Journal of Plasticity 144 (2021): 103059. https://doi.org/10.1016/j.ijplas.2021.103059

Overview

Machine learning (ML) methods are commonly used for pattern recognition in almost any field one could imagine. ML techniques can also offer a substantial improvement in computational time when compared to conventional numerical methods. In this research, a machine learning- and crystal plasticity-based framework is presented to predict stress-strain behaviour and texture evolution for a wide variety of materials within the face-centred cubic family (FCC). Firstly, the process of the framework design is described in detail. The proposed framework was designed to be built of ensemble of artificial neural networks (ANN) and a crystal-plasticity based algorithm. Next, the dataset constituent of crystal plasticity simulations was collected. The dataset consisted of examples of monotonic deformation cases, was prepared for training using mathematical transformations, and finally used to train ANNs used in the framework. Then, the ML framework was demonstrated to predict full stress-strain and texture evolution of different FCC single crystals under uniaxial tension, compression, simple shear, equibiaxial tension, tension-compression-tension, compression-tension-compression, and, finally, for some arbitrary non-monotonic loading cases. The proposed framework predicts the stress-strain response and texture evolution with a high degree of accuracy. The results demonstrated in this research show that the proposed machine learning- and crystal plasticity-based framework exhibits a tremendous computational advantage over conventional crystal plasticity model. Finally, one of the most important contributions of this work is to show the framework's feasibility. The work demonstrates that machine learning methods can help predict complex strain paths without having to train machine learning models on the infinite set of possible non-monotonic loading scenarios.

5.1 Introduction

Metals are used in a large variety of structural applications and are of high commercial importance. Any metal part, before its production, is modelled and tested to ensure its proper performance. Testing is often carried out by computer-aided engineering, which invokes the usage of complex simulation tools such as finite element (FE) modelling software. Depending on a system, the deformation mechanics can be intricate and can include a large variety of internal state variables in addition to the observable state variables such as geometry, deformation, or temperature [146]. While the use of internal state variables can help describe the response mechanisms more clearly [147, 148, 149, 150], it comes at a price of high computational effort and is, as a result, time-consuming.

Crystal Plasticity (CP) models are mesoscopic physics-based models and allow exceptionally accurate estimations of material mechanical response on multiple lengths of scales: from single grains in materials to engineering parts [151]. In the last thirty years, CP modelling has developed into a very versatile tool that predicts deformation for a wide range of materials such as steel alloys [152], aluminium alloys [79], magnesium alloys [153], titanium alloys [154], and other crystalline materials. CP constitutive equations can be used to incorporate numerous deformation aspects such as dislocation density [155, 79], grain boundary strengthening [156], and dynamic recrystallisation [157], and account for material behaviour at elevated temperature conditions [158] and high strain rates [159]. To account for all of the desired effects, one must include all necessary variables and equations into the constitutive model and then calibrate it for a specific material. Even a basic uniaxial tension CP simulation can be very time consuming [160], as the computational process includes the solution of a number of highly non-linear partial differential equations via numerical methods for hundreds of thousands of integration points. With the complexity involved, CP is a state-of-art method that can help design advanced materials with the exact desired properties for any field of application.

The ability to predict metal deformation accurately and swiftly is crucial, especially when we want to prototype new materials rapidly. Various approaches were investigated to accelerate crystal plasticity calculations, including parallel computing [161], and the wavelet transformation algorithm for accelerated CP simulation of cyclic loading [162, 163]. Kalidindi et al. demonstrated a spectral database method to accelerate crystal plasticity constitutive calculations [164, 165, 166, 167]. The presented database consists of discrete Fourier transforms coefficients that accurately capture the main solutions of CP constitutive equations, thereby avoiding direct computation of solutions. Another work of Kalidindi and co-workers is the Material Knowledge System (MKS) framework – a database approach to help establish linkages within material properties [92, 168, 169, 103]. MKS employs methods of data science, machine learning, statistics, and other fields to create structure-property-processing relationship for different materials and is used for various applications such as prediction of texture sensitive flow responses [94] and microscale plastic strain rate fields [107] in composite materials and other multiscale phenomena in material modelling [105, 106].

Machine learning (ML) based approaches can be fast once trained and become increasingly accessible. They are designed to capture highly non-linear relations and do not require auxiliary databases for their implementations [45]. Ghaboussi et al. [109] suggested using artificial neural networks (ANNs) as an alternative to mathematical models of material behaviour and have put the beginning to the applications of machine learning in computational material science. ML methods have been demonstrated to accurately predict various

quantities in material science in the past thirty years. Ghaboussi et al. [170] used ANNs to model the constitutive behaviour of materials. Lefik et al. [171] used an ANN as an incremental constitutive model implemented in an FE code and demonstrated the predictions of stress paths for a given strain history. A similar prediction approach for flow response was utilised later in other studies [172, 173]. Brahme et al. [113] demonstrated the capability of ANNs to predict cold-rolling fibre texture of steel based on the initial texture data and rolling reduction parameter. Li et al. [110] presented an ML model that predicted material hardening; this approach was more efficient than the conventional calibration of a yield function. One of the significant studies was conducted by Mozaffar et al. [10], who demonstrated the prediction of path-dependent material response in terms of stresses and plastic energy for two-dimensional representative volume element (RVE) with a sequence-learning model. Gorji et al. [111] used several different ANNs to replicate the predictions of anisotropic Yld2000-2d model for various loading conditions. Their worked demonstrated excellent predictive capabilities of feed-forward neural networks and gated-recurrent units network for monotonic and non-monotonic loadings, predicting stress states based on the applied strain. In the recent study, Muhammad et al. [115] demonstrated an ML framework utilised to predict the evolution of local strain distribution, plastic anisotropy and failure during tensile deformation of 3D-printed aluminium alloy.

Numerous studies involved the utilisation of neural networks and other machine learning algorithms in CP applications. Mangal and Holm [134, 135] used classification ML methods to predict the formation of stress hotspots in polycrystalline materials under uniaxial tension for face-centred cubic (FCC) and hexagonal close-packed (HCP) materials. A dataset of synthetic three-dimensional (3D) microstructures, including microstructural descriptors for each grain, and the stress and strain fields resulting from CP simulations, was also provided by the researchers [136]. Ali et al. [137] proposed an ANN-based framework to predict the flow response and texture evolution of polycrystals subjected to multiaxial and non-proportional loading. They accurately predicted the constitutive response of AA6063-T6 alloy under uniaxial tension followed by simple shear. Other applications of this method included: linear regression and neural networks applied to predict a cyclic stress-strain relation for steels [138], sequence learning methods were applied to predict microstructural texture evolution for an FCC copper sample under uniaxial tension [139], ANNs applied to predict the relation between CP parameters and the microstructural texture for a titanium alloy under uniaxial tension [140], a classification ML algorithm using spatial strain correlation to infer and classify prior deformation history in thin aluminium films [141].

The key to the successful training of an ML model for computational material science applications is to form a meaningful dataset about material behaviour. Most of the men-

tioned studies highlighted the computational advantage of machine learning models and high accuracy within the bounding box of a training set. The downside of using those models is that their accuracy is typically limited to a range and scope of a training set [45] as ML typically performs poorly in the extrapolation tasks, although being very good at generalisation. This study will show that it is possible to use machine learning as a tool to achieve accurate predictions of complex phenomena such as plastic deformation and texture evolution far beyond the range and scope of a training set.

This paper introduces an approach to couple machine learning and crystal plasticity modelling method. The framework resulting from this approach enables fast and accurate predictions for complex loading for a broad range of face-centred cubic materials. The proposed framework is based on an ensemble of feed-forward neural networks (FFNNs), namely multi-layer perceptrons, trained on monotonic loading scenarios, implemented with a CP update-algorithm. This algorithm allows making predictions far beyond the set of loading conditions included in a training set. It is capable of accurate predictions of stress and texture evolution for a given crystalline material. In addition, the framework predicts evolving value of critical resolved shear stress (CRSS) and accumulated shear as internal variables, which are used to ensure accurate predictions in the cases of non-monotonic loading scenarios.

The organisation of this paper is as follows. The current section presented an introduction to the proposed work. Section 5.2 introduces crystal plasticity constitutive model and the development process of the proposed framework. Section 5.3 presents the predictions obtained with the framework and comparing them with crystal plasticity simulations. Finally, Section 5.4 summarises the work and concludes the obtained results.

5.2 Methods & Development of Machine Learning Framework

The goal of the present work is to develop and implement a machine learning framework to address the computational drawback associated with crystal plasticity calculations. The proposed framework uses a machine learning- and crystal plasticity-based method that is built upon the data of CP simulations to compute the evolution of stress and texture during deformation under any complex strain path for a wide range of FCC materials. Figure 5.1 illustrates the generalised schematics of the proposed framework. The framework involves:

- selection of necessary variables,

- CP simulations for those variables,
- constructing a dataset based on the performed simulations,
- data normalisation to achieve better training results,
- machine learning model selection, training, and evaluation,
- and implementing a CP update algorithm.

The main idea of the implementation of the CP update algorithm, in order, is to a) identify the evolving parameters in the CP framework, b) use the trained ML algorithm, calculate the update on the evolving parameters using the obtained predictions, and c) use the updated parameters as inputs to next the iteration of the ML algorithm.

The next subsections provide details on the CP constitutive model used in this work, dataset generation, the coupling of ML and CP, and artificial neural networks.

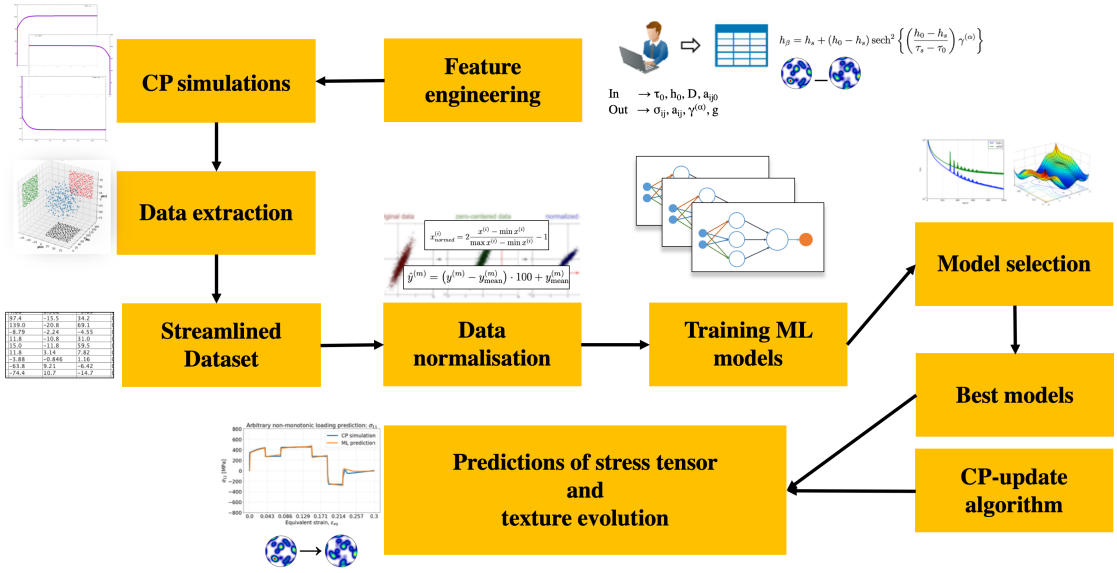

Figure 5.1: Schematic representation of the proposed framework

5.2.1 Crystal Plasticity Constitutive Model

Flow behaviour of materials can be modelled using various approaches. Phenomenological plasticity models are most commonly used and are capable of capturing material behaviour under variety of loading conditions [174, 175]. However, these models are not designed to predict a texture change within deformed material, in contrast to crystal plasticity models which use crystallographic slip mechanism in their formulations. The growing body of literature demonstrates the power of crystal plasticity models for face-centered cubic [41, 176], body-centered cubic [177, 178, 179], and hexagonal closest packed crystal structure materials [180, 181].

The rate-dependant CP formulation based on the work by Asaro and Needleman [36] is employed in this work. In this formulation, the plastic deformation occurs due to the mechanism of crystallographic slip. For an FCC metal, e.g. Aluminum, slip occurs along three unique $< 110 >$ slip directions and in the four unique [111] slip plane. Consequently, there are twelve unique slip systems for materials with FCC crystal structure. The elastic constitutive equation is expressed as:

$$\overset{\nabla}{\sigma} = \mathcal{L}D - \dot{\sigma}^0 - \sigma \operatorname{tr} D \tag{5.1}$$

where $\overset{\nabla}{\sigma}$ is the Jaumann rate of Cauchy stress, D is the strain-rate tensor, $\mathcal{L}$ is the elastic stress tensor, and $\dot{\sigma}^0$ is the viscoplastic type stress state.

For each slip system α slip rates are calculated as:

$$\dot{\gamma}^{(\alpha)} = \dot{\gamma}_0 \operatorname{sgn} \tau^{(\alpha)} \left| \frac{\tau^{(\alpha)}}{g^{(\alpha)}} \right|^{1/m} \tag{5.2}$$

where $\dot{\gamma}_0$ is the reference shear rate, $\tau^{(\alpha)}$ is the resolved shear stress on slip system α, m is the index for strain-rate sensitivity, and $g^{(\alpha)}$ is slip system hardness parameter.

The single crystal work hardening is expressed as:

$$\dot{g}^{(\alpha)} = \sum_{\beta} h_{\alpha\beta} \dot{\gamma}^{(\beta)} \tag{5.3}$$

where $h_{\alpha\beta}$ are the hardening moduli described as:

$$h_{\alpha\beta} = q_{\alpha\beta} h_{\beta} \quad \text{(no sum on } \beta\text{)} \tag{5.4}$$

where $q_{\alpha\beta}$ is the term for latent hardening behaviour of a crystal, and h_β is the single crystal hardening rate. The term for latent hardening expressed as:

$$q_{\alpha\beta} = \begin{bmatrix} A & qA & qA & qA \\ qA & A & qA & qA \\ qA & qA & A & qA \\ qA & qA & qA & A \end{bmatrix} \tag{5.5}$$

where A is a 3×3 matrices fully populated with ones, and q is the ratio of latent hardening to self hardening and is set to one.

Finally, single slip hardening used in this work is governed by a hardening law proposed by Chang and Asaro [182] and is expressed as:

$$h_\beta = h_s + (h_0 - h_s)\,\mathrm{sech}^2 \left\{ \left(\frac{h_0 - h_s}{\tau_s - \tau_0} \right) \gamma^{(\alpha)} \right\} \tag{5.6}$$

This hardening model depends on the parameters that define material flow behaviour: the initial hardness τ_0, the saturation shear stress τ_s, the initial hardening modulus h_0, and the parameter that controls the slope at the saturation regime h_s.

Following the framework presented by Rossiter et al. [86], equations 5.1-5.6 are used to update the stress and the texture. These equations form the core of CP theory and entail a computational drawback associated with CP simulations. The present work aims to enable machine learning to eliminate this computational complexity by achieving reliable predictions comparable to crystal plasticity simulations. The ML framework proposed in this work is used to compute the stress tensor, texture evolution (i.e. rotation matrix), the accumulated shear, and the evolving critical resolved shear stress.

A comprehensive description of this CP model can be observed in the works by Inal et al. [40] and Rossiter et al. [86]. The presented crystal plasticity constitutive model is used within an in-house crystal plasticity code and was employed to generate the training dataset presented in the subsection 5.2.2.

5.2.2 Dataset Generation

A training dataset has a strong influence on the accuracy of the ML model [183]. It is vital that the dataset used to construct an accurate machine learning model contains enough

information that is representative of a given problem. It could be computationally expensive to collect such a dataset, especially in the case when the model has many input features, as the difficulty of the problem increases with the increase of the number of input features [184, 185]. Therefore, the input to the model needs to have the minimum amount of features that are necessary to perform accurate predictions, and the selection of those features requires the understanding of the field of application. In this case, constructing robust dataset with all necessary information requires understanding of CP and its computational drawbacks.

The process of selecting features out of available data and transformation of those features for a machine learning model is referred to as "feature engineering" [122, 123, 124, 125]. It is a fundamental process used in ML, as a training dataset has a strong influence on model accuracy. In this work, the process of feature engineering included a selection of a subset of variables from available variable space, which should be sufficient to train an accurate machine learning model f: $f(\text{input}) = \text{output}$. The numerical range for each selected variable was defined and is explained later in this section.

The CP model explained in subsection 5.2.1 was utilized to generate the dataset for training. The model used the hardening law proposed by Chang and Asaro [182] and is given in equation 5.6. This hardening law offers the most flexibility in capturing the flow behaviour for a wide variety of conditions. The input features used for training the machine learning model include material parameters used in this hardening model viz. initial hardness, initial hardening modulus, as these parameters have a significant influence on the material deformation behaviour. In addition, the input included the strain-rate tensor and initial texture of the crystal. The range on the input parameters is chosen such that it covers a wide variety of material response within the FCC family. This is the minimal set of features that could be used for training a machine learning model, as hardening parameters regulate a material behaviour, and initial texture and the strain path are the essential descriptors of a single crystal and the deformation that it is undergoing. Therefore, the feature space cannot be reduced to fewer features. Output features of the machine learning framework are the CRSS, stress tensor, texture evolution in the form of rotation matrix, and accumulated shear value. Input and output variables are summarised in tables 5.1 and 5.2 correspondingly. The data was generated for 50 timesteps with $\Delta t = 0.001$ which gives a maximum strain equal to 0.05.

Optimal filling of a feature domain is beneficial for training an optimal model. In this work, no prior knowledge of the feature space is assumed therefore it is advantageous to sample it uniformly. Using the method of grid-sampling would result in a uniform and robust dataset, but it would be very large due to the number of input parameters to the model, and both data-mining and training would be computationally expensive.

Table 5.1: Input features, their ranges, and description of the feature and the physical parameters it controls.

Input variable	**Range**	**Description**
τ_0	$100.0 \ldots 200.0$ MPa	Initial hardness
h_0	$70.0 \ldots 500.0$ MPa	Initial hardening modulus
D_{ii}, $i = 1 \ldots 3$	$-1.0 \ldots 1.0$	Diagonal components of a strain-rate tensor
D_{12}	$-1.0 \ldots 1.0$	12-component of a strain-rate tensor
$a_{0_{ij}}$, $i, j = 1 \ldots 3$	$-1.0 \ldots 1.0$	Initial texture in a form of rotation matrix

Table 5.2: Output features and their description.

Predicted variable	**Description**
g	Critical resolved shear stress (MPa)
σ_{ij}, $i, j = 1 \ldots 3$	Stress response (MPa) (6 values, due to symmetrical nature of stress tensor)
a_{ij}, $i, j = 1 \ldots 3$	Texture evolution (9 values of rotation matrix)
$\gamma^{(\alpha)}$ (ashear)	Accumulated shear value

To overcome this problem and have a robust dataset, Sobol sequences were employed to sample the data. Sobol sequences, also known as LP_τ sequences, have been shown to sample a domain efficiently [126, 127, 186, 187]. Literature highlights Sobol sequences for their beneficial space-filling properties [188, 121]. The significant advantage of the sequences is in their low-discrepancy property, i.e. the points in the sequence are almost equidistributed. This fact ensures the domain is filled uniformly but yet quasi-randomly, which is beneficial for training a machine learning model. The algorithm on Sobol sequence generation is explained in Bratley and Fox [186]. The motivation behind utilisation of the Sobol sequence is to generate a sequence (set of numerical points) that is "well-spaced" in the s-dimensional unit cube $I^s = [0, 1]^s$, such that:

$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} f(x_i) = \int_{I^s} f, \tag{5.7}$$

where f is some real integrable function over I^s. The Sobol sequence generation algorithm generates data in the range $[0, 1]$. Each feature must then be converted to its range. Figure 5.2 depicts the visual comparisons of three distinct datasets generated for 343 Euler angles (the number was picked for convenience of the grid sampling approach), and their projections on the coordinate planes $phi1 - Phi$, $Phi - phi2$, and $phi1 - phi2$. The first set was sampled with Sobol sequences, the second – with random sampling from a uniform distribution, and the third one – with grid sampling. One can observe that the first sampling approach provides a more equi-distributed set of points as compared to that generated by the random sampling which may result in clusters of points. These clusters can results in overfitting during training. Finally, grid search does not provide diversity in sampling, in comparison to sampling with Sobol sequences or random sampling, especially in the case of high-dimensional feature space. Therefore, using Sobol sequences in sampling the input data is beneficial for machine learning tasks.

To summarise, the final dataset included $1,451,161$ samples corresponding to unique loading conditions, and the input parameters were sampled using Sobol sequences. Each sample consisted of an input-output set and corresponded to a unique monotonic loading. The total time it took to collect all the data was eight and a half hours for 32 parallel processes, each running on a separate central processing unit core.

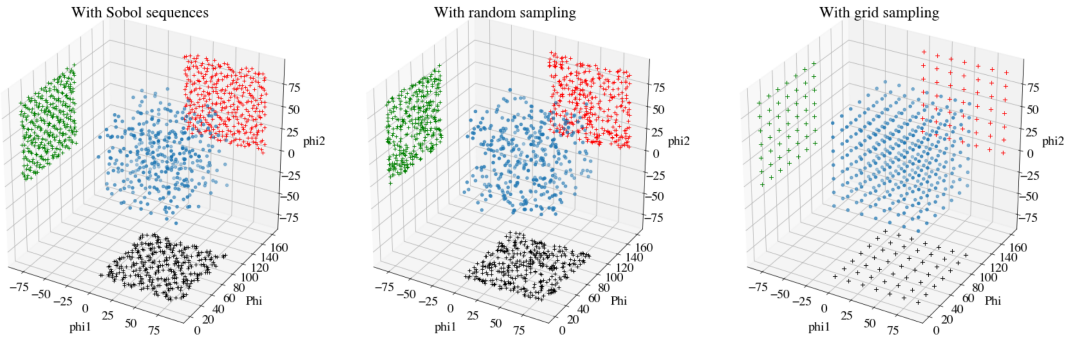

Figure 5.2: Visual comparison of sampling with Sobol sequences, random sampling from a uniform distribution, and grid sampling approaches.

Finally, it is very important to efficiently store large datasets. When the available data is small in size, it is often convenient to use standard flat formats such as text (.txt) files, comma-separated values (.csv) files, or excel (.xlsx) files, as such files are easy to open, modify, and access. However, when the data gets more substantial in size, one can hit a roadblock of fitting it into the memory and performing calculations on it. The Hierarchical

Data Format (HDF5, where five stands for the version number) technology suite [69] is an efficient method of compression and storage of scientific data that is too large to store in memory. This method helps to efficiently organise the large amounts of data, allows compliant and efficient input and output, makes the dataset easily transferable, extensible, and accessible in many frameworks or by the aid of various programming languages including Python (second and third versions), R, FORTRAN, and C++. Within HDF5 data files, the data is stored hierarchically, in forms of multidimensional arrays. The technology suite allows very efficient data compression. A comparison of two files storing the same single-precision floating-point numerical data array of shape $(100000000, 10)$ was performed. The comma-separated data file storing this array used 25.1 gigabytes (GB) of memory, and the HDF5 data file used only 4.0 GB. Data loading time was also significantly improved: it took 18.5 minutes to access the data from comma-separated value file, and only 590 μs to access data stored in HDF5 format. The latter method of dataset storage was used in the research.

5.2.3 Coupling Machine Learning and Crystal Plasticity

The strength of CP is the ability to predict material response along any complex strain path once it is calibrated for the given material [137, 189]. As mentioned earlier, its main drawback is computational cost, especially when one has to account for complex microstructure with a relatively large number of grains/orientations. Machine learning methods can help eliminate a computational drawback associated with crystal plasticity. The proposed framework couples application of machine learning and crystal plasticity. Training a machine learning model to achieve predictions for any complex strain path, even though possible, is not feasible due to the infinite number of complex loading conditions that will have to be generated, and the model has to be trained on that set. Training an ML model on monotonic loading examples would not result in the model capable of predicting material behaviour under complex non-monotonic strain path because non-monotonic loadings lie in the different distribution of strain paths [190]. The challenge is to devise a framework that can predict the response using relatively simple/monotonic strain paths and use those to predict complex strain paths.

In this work, we used the knowledge of CP to develop a framework that is pseudo-recursive in nature. The design of the framework relies on the correct parameterisation of the CP model and the selection of a feature set that encompasses the parameter set presented in table 5.1. To perform the correct parametrisation, the identification of CP constitutive model evolving features and their ranges of variation were identified. Then, to predict a result of material deformation under a complex strain path, this complex strain

path is to be broken into monotonic parts. The ML algorithm was used to predict output variables for first monotonic part of the strain path and pass them to the CP-update algorithm, receive the updated variables, and repeat the process the needed number of times. The essence of the CP-update algorithm is in keeping the evolving material parameters updated throughout the deformation process like initial hardness τ_0 and hardening modulus h_0. The detailed explanation of prediction process is presented below.

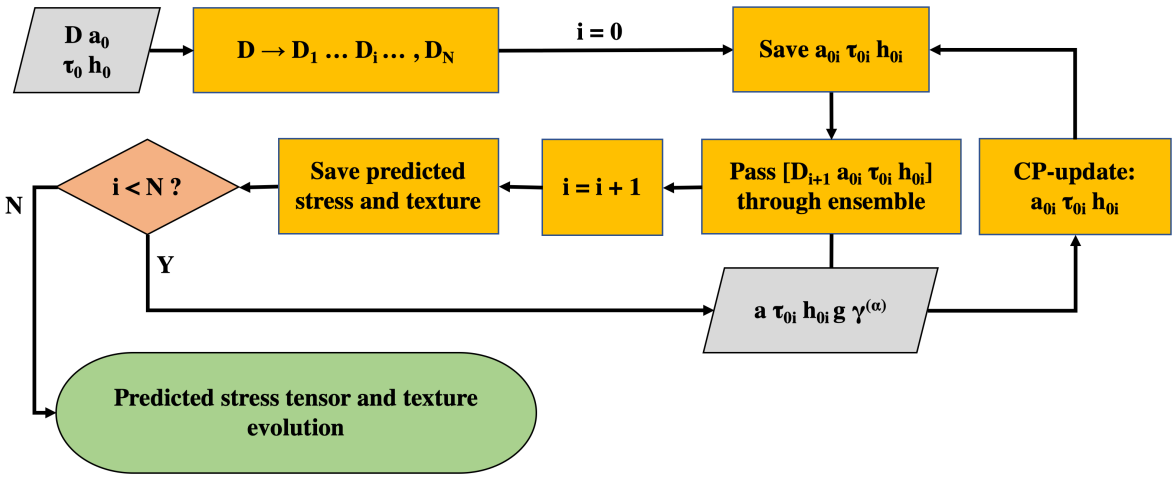

Figure 5.3: The flowchart describes the process of obtaining the predictions for a given strain path. Parallelogram units represent the data that is used as input. Rectangle units show processes in the framework. The diamond unit indicates the decision point. Finally, the rounded rectangle unit is the terminal process that outputs final stress-strain and texture evolution predictions.

Figure 5.3 presents the flowchart that describes the process of prediction for stress and texture evolution for any given strain path. If the strain path of interest is monotonic and the total deformation is less than 0.05 in strain, then the inputs shown in the top left corner of the flowchart (figure 5.3) are sufficient to obtain the predictions for the stress and the evolved texture. The limitation of 0.05 in strain arises from the fact that the implemented ML algorithm was trained on the dataset consisting of monotonic loadings that were 0.05 in strain. It should be mentioned that, for a monotonic strain path of interest less than 0.05 in strain, the predictions of the evolved quantities like $\{g, \gamma^{(\alpha)}\}_{output}$ are inconsequential and only the stress tensor and the texture outputs need to be preserved.

In the case when total strain is larger than 0.05 or in the case when the strain path

is non-monotonic, i.e. the strain-rate tensor, D, changes during loading, the following procedure is employed. As the first step, the strain path is split into monotonic parts that are less than 0.05 in strain. For each of the monotonic parts, the prediction procedure is identical to that of the single strain path, i.e. $\{\tau_0, h_0, D, a_{0_{ij}}\}_{input} \rightarrow \{g, \sigma_{ij}, a_{ij}, \gamma^{(\alpha)}\}_{output}$. It should be mentioned that the input of the initial hardness, hardening modulus, and initial texture for the first monotonic part is identical to the given initial parameters: τ_0, h_0, $a_{0_{ij}}$. Beside the material parameters, the other input is the initial strain-rate D_1. Passing the above input through the ensemble of ANNs produces the following outputs: $\{g_1, \sigma_{1_{ij}}, a_{1_{ij}}, \gamma_1^{(\alpha)}\}_{output}$, updated texture as well as the variables g_1 and $\gamma_1^{(\alpha)}$, which are used to update CP parameters. Note that this procedure is repeated until the desired strain levels are reached for each strain path individually (in parts). As for the CP parameter update, the initial hardness τ_0 is updated with the predicted value of CRSS, g. The predicted values of CRSS exceeding the value of the saturation shear stress, τ_s, are assigned to the value of τ_s. Finally, the hardening modulus is adjusted according to:

$$h_{0_{i+1}} = \begin{cases} h_s + (h_{0_i} - h_s)\,\mathrm{sech}^2\left\{\left(\frac{h_{0_i} - h_s}{\tau_s - \tau_{0_i}}\right)\gamma_{(i-2)}^{(\alpha)}\right\} & \text{if } g_i < \tau_s \ (i = 1 \ldots N \text{ and } \gamma_{(-1)}^{(\alpha)} = 0), \\ h_s & \text{otherwise} \end{cases} \tag{5.8}$$

where i represents the number of a pass.

The flowchart for the entire procedure is illustrated in figure 5.3, and the used ensemble of ANNs is schematically presented in figure 5.4. Accordingly, an input vector is fed to each network in the ensemble. Each of the networks computes its own feature. The output of the ensemble consists of networks' predictions.

The next subsection presents the details about ANNs used in this framework.

5.2.4 Artificial Neural Networks

There are various ANNs currently being adopted. The list of proposed types of ANNs includes, but is not limited to, the multilayer perceptron feed-forward neural networks, convolutional neural networks (CNNs), and recurrent neural networks (RNNs).

Each class of ANNs usually corresponds to a particular kind of problem. For example, FFNNs are commonly used in any field and are also well-known as "universal approximators" [191]. FFNNs are typically used with the tabular data and can be applied to a broad spectrum of supervised machine learning tasks [192]. CNN is a class of ANNs commonly used in computer-vision application, object recognition, imagery analysis, and other

image-related studies [193]. Note that usage of CNNs is not limited to these tasks. CNNs usually majorly consist of convolutional layers, which are more complex than fully connected layers of FFNNs. RNN is a class of ANNs that is commonly used for the prediction and classification of sequential data of a varied length (such as text or sound) and is noted to have the ability to cope with a broad spectrum of temporal dependencies [194]. Note that the usage of RNNs is not limited to the tasks of sequence learning. There is a number of different sub-types of RNNs, such as Long-Short Term Memory (LSTM) networks [195] or Gated Recurrent Unit (GRU) networks [196]. These networks are more complex than FFNNs in architecture since the connections between the nodes of RNNs form loops, which allow previous outputs to be used as inputs while having hidden states [197]. In addition, these networks can have variations such as deep RNNs, and Bi-directional RNNs [198].

In contrast to CNNs and RNNs, FFNNs are simpler in their architecture (only layers and neurons in hidden layers are subject to variation) and, therefore, easier to be tested in their variety. The data used in the current research was used in the tabular form and related initial material parameters to the corresponding stress-strain response and texture evolution. Hence, the feed-forward neural network with a backpropagation algorithm was used in this work. From here on, the term "ANN" has been used to refer to feed forward neural network. Figure 5.5 demonstrates a schematic representation of such neural network.

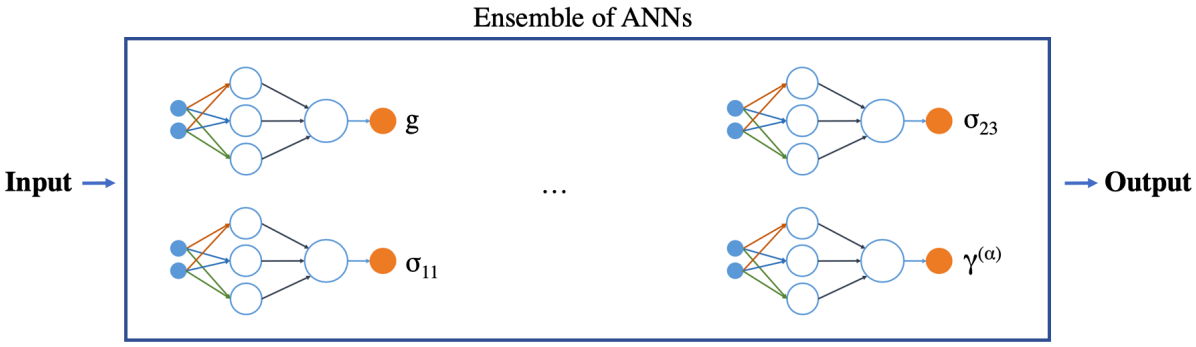

Figure 5.4: The figure represents a schematic of ensemble of artificial neural networks. An input vector is fed to each network in the ensemble. Each of the networks computes its own feature. The output of the ensemble consists of networks' predictions.

The result of training the network is a continuous function, that is formed from a set of interconnected unit cells [199]. The assembly of the unit cells into layers is referred to as network architecture. Each unit cell performs a mathematical transformation of weights and biases and produces an output. The forward-propagation through a feed-forward

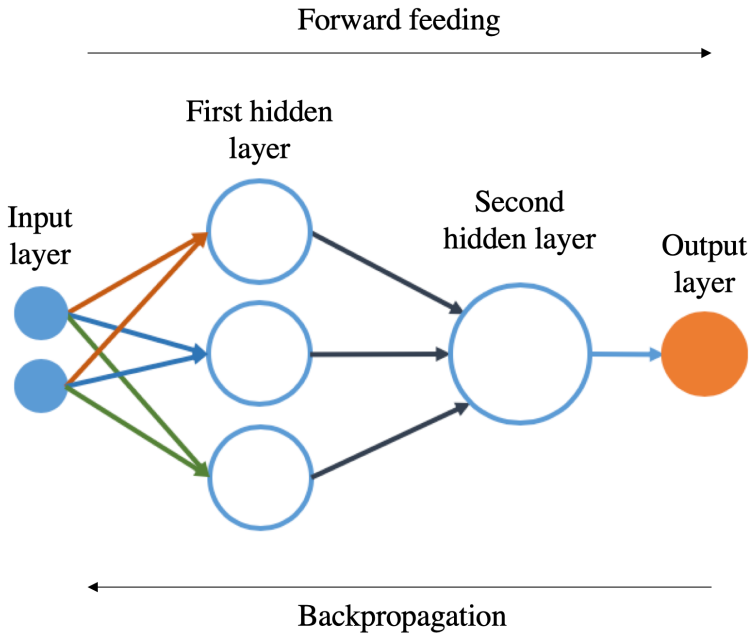

Figure 5.5: Schematic representation of a feed-forward artificial neural network with two input unit cells, two hidden layers and one output unit cell. The network has three unit cells in the first hidden layer, and one unit cell in the second hidden layer.

neural network can be described as:

$$\begin{aligned} a^{[1]} &= g\left(\theta^{[1]}x + b^{[1]}\right) \\ a^{[2]} &= g\left(\theta^{[2]}a^{[1]} + b^{[2]}\right) \\ &\dots \\ a^{[l-1]} &= g\left(\theta^{[l-1]}a^{[l-2]} + b^{[l-1]}\right) \\ y &= \theta^{[l]}a^{[l-1]} + b^{[l]} \end{aligned} \tag{5.9}$$

Where $\theta^{[l]}$ is a weight matrix in the l^{th} layer, $b^{[l]}$ is a bias vector in the l^{th} layer, $a^{[l]}$ is an output vector in the l^{th} layer, g is an activation function, x is an input vector, and y is an output vector of a neural network. Details about activation functions, normalisation, evaluation, and training of the neural network, used in this work are discussed in further subsections.

Activation Function & **Normalisation**

Activation functions transform a value in a unit cell, typically in a non-linear way [200]. In this study, the hyperbolic tangent activation function was used:

$$\tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})} \tag{5.10}$$

This activation function is continuous and differentiable and has a range of $[-1, 1]$. In the neural network design, this function is often preferred, as its gradients vary in their directions. Furthermore, it is zero centred unlike the sigmoid activation function [201]. In this work the input and output variables were normalized between -1, 1. The hyperbolic tangent activation function performs better with this normalisation. To normalise the i^{th} input or output feature in $[-1, 1]$ the following mathematical transformation must be applied to it:

$$x^{(i)}_{normed} = 2\frac{x^{(i)} - \min x^{(i)}}{\max x^{(i)} - \min x^{(i)}} - 1, \tag{5.11}$$

where $\min x^{(i)}$ is a minimum and $\max x^{(i)}$ is a maximum value of the i^{th} feature of the input vector or the output vector in the training (i.e. separate from testing and validation parts) dataset.

The other normalisation used in this research was normalisation by range adjustment. This normalisation method is proposed in order to improve the texture evolution (a_{ij}, $i, j = 1 \dots 3$) learnability. Since all components in the orientation matrix fall within the range of $[-1, 1]$, normalisation is not required. However, the statistical range (i.e. the difference between the largest and smallest values) of each output curve a_{ij} is very small. Maximal statistical ranges of each curve do not exceed 0.03. Therefore, to improve the learnability of texture evolution, it is beneficial to adjust their range by its enlargement. To enlarge the statistical range for each curve by the factor of 100, the following mathematical expression was applied:

$$\hat{y}^{(m)} = \left(y^{(m)} - y^{(m)}_{\text{mean}}\right) \cdot 100 + y^{(m)}_{\text{mean}}, \tag{5.12}$$

and the backward transformation is described by:

$$y^{(m)} = \left(\hat{y}^{(m)} - \hat{y}^{(m)}_{\text{mean}}\right) / 100 + \hat{y}^{(m)}_{\text{mean}}, \tag{5.13}$$

where m is a number of samples in a dataset. Note, that the mean value for the original and the transformed curves will be the same (i.e. $\hat{y}^{(m)} = y^{(m)}$ for each m). Further normalisation is not required, as the last layer's normalisation function is a linear (pass-through) activation function and can predict the values that lie outside the $[-1, 1]$ range.

Normalisation for input strain rate tensor is not required, as all its values fall within the $[-1, 1]$ range. Table 5.3 summarises the normalisation methods for each variable.

Table 5.3: Features and their normalisation methods.

Variable	**Normalisation method**
$\mathrm{D_{ij}}$, $i, j = 1 \dots 3$	No normalisation
τ_0	Normalisation in $-1 \dots 1$
$\mathrm{h_0}$	Normalisation in $-1 \dots 1$
g	Normalisation in $-1 \dots 1$
σ_{ij}, $i, j = 1 \dots 3$	Normalisation in $-1 \dots 1$
$\mathrm{a_{ij}}$, $i, j = 1 \dots 3$	Normalisation by range adjustment
$\gamma^{(\alpha)}$	Normalisation in $-1 \dots 1$

Evaluation of Artificial Neural Networks

The cost function, $J(\theta, b)$, and loss function, $\mathcal{L}\left(y^{(i)}_{pred}, y^{(i)}_{true}\right)$, measure the "goodness" of the trained network. The cost function measures the performance of the model on the dataset. The value of the function is minimised by an optimisation algorithm during training by updating the weights' and biases' values in the "right" direction [45]. The loss function measures how well the model performs on a single training example. Cost function depends on the loss function as follows:

$$J(\theta, b) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\left(y^{(i)}_{pred}, y^{(i)}_{true}\right), \tag{5.14}$$

where m is a number of samples in an evaluated dataset.

Mean squared error (MSE) is a common loss function for regression problems and is used as both loss function and evaluation metrics in this research. The formula for MSE is the following:

$$\mathcal{L}_{\text{MSE}}\left(y_{pred}^{(i)}, y_{true}^{(i)}\right) = \frac{1}{n}\sum_{i=1}^{n}\left(y_{true}^{(i)} - y_{pred}^{(i)}\right)^2, \tag{5.15}$$

where n is a number of elements in an output vector.

Optimisation Algorithm

Adaptive Moment Estimation (Adam) [55] is one of the recently developed gradient descent-based backpropagation optimisation algorithms and is commonly used. This algorithm was used for training ANNs. The algorithm computes adaptive learning rates for each parameter. It stores an exponentially decaying average of past squared gradients v_t like algorithms such as Nesterov's accelerated gradient method [54], MaxProp [59], AdaDelta [57], and others. It differs in the way it updates an exponentially decaying average of past gradients:

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1)\, g_t, \\ v_t &= \beta_2 v_{t-1} + (1-\beta_2)\, g_t^2, \end{aligned} \tag{5.16}$$

where m_t and v_t are approximations of the first moment (the mean) and the second moment (the non-centred variance) of the gradients. Bias-corrected first and second moment estimates are computed to counter moments being biased towards zero:

$$\begin{aligned} \hat{m}_t &= \frac{m_t}{1-\beta_1^t}, \\ \hat{v}_t &= \frac{v_t}{1-\beta_2^t}. \end{aligned} \tag{5.17}$$

Eventually, parameters are updated according to:

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{\hat{v}_t} + \epsilon}\hat{m}_t, \tag{5.18}$$

where β_1, β_2, α, and ϵ are the algorithm hyperparameters and are subject to tuning.

Artificial Neural Networks Architectures

In order to select ANNs for the proposed framework, several networks with different architectures were compared in their performance. The effect of the total number of layers and the number of neurons per layer on the accuracy of the network was studied. Figure 5.6 demonstrates the evolution of the mean squared errors for training and validation dataset during training for different architectures of the model for prediction of a σ_{11}-stress component. The network number, in the legend of the figure, corresponds to a different architecture which are presented in the table 5.4 along with the corresponding validation errors. It can be noted, that the ANNs (5) and (6) have the same number of hidden layers and neurons within hidden layers, but the sequence of hidden layers is different. It can be observed that the network (6) which has a descending number of unit cells from input to output is more accurate than the network in which unit cells in hidden layers ascend from input to output. It is observed that the training and validation error decreases proportionally to the depth on the network. The network (9) is a deep neural network and has 15 hidden layers, and it is the most accurate out of all other networks. An ANN with this architecture was chosen for prediction of σ_{11} stress tensor component based on these observations.

Table 5.4: The architectures used in the architecture search for training an artificial neural network to predict a σ_{11} component of a stress response

Architecture number	**Architecture**	**Validation MSE**
(1)	$[512]$	$1.09E-02$
(2)	$[1024]$	$1.31E-02$
(3)	$[1024-512]$	$6.30E-03$
(4)	$[1024-512-512-26]$	$4.25E-03$
(5)	$[128-256-512-1024-1024-2048]$	$6.25E-03$
(6)	$[2048-1024-1024-512-256-128]$	$2.69E-03$
(7)	$[4096-2048-1024-1024-1024-512-512-512]$	$2.72E-03$
(8)	$[1024-1024-1024-512-512-512-256]$	$2.97E-03$
(9)	$[2048-1024-1024-512-512-512-512-$ $-256-256-256-256-128-128-128-128]$	$1.37E-03$

For brevity, process of model selection is shown only for σ_{11} stress tensor component. Similar procedure was performed for all other output variables, and the most accurate

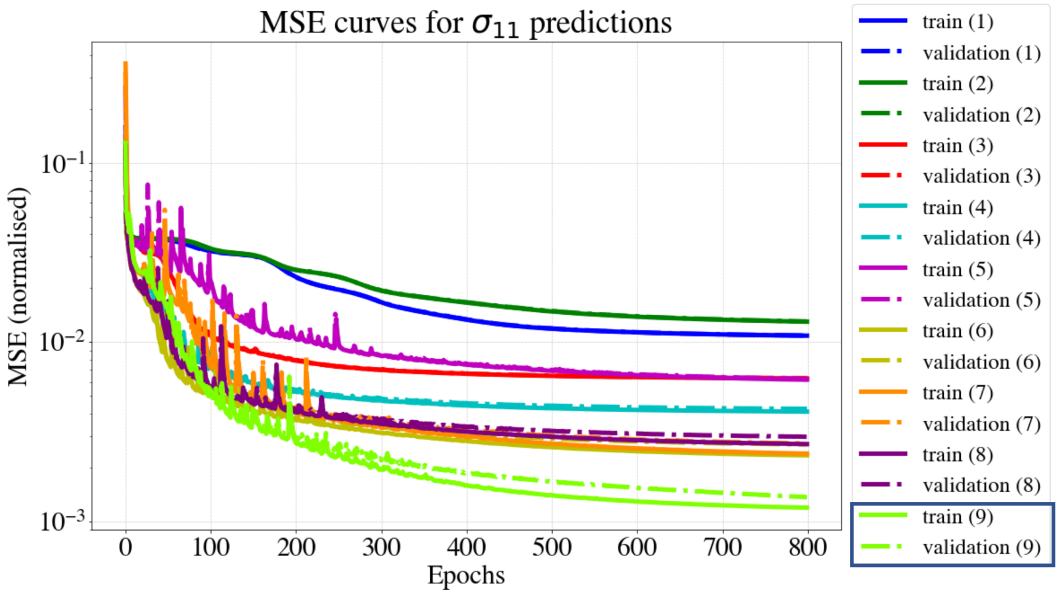

Figure 5.6: Neural networks learning rates showing the evolution of losses for training and validation datasets. Each network is numerated, and each number corresponds to unique network architecture. For each ANN, the architecture and validation error are summarised in table 5.4.

models were incorporated into the framework. Final architectures for each model are reported in table 5.5.

To train neural networks, the available dataset was split into three parts: training set, validation set, and test set. CP simulations were performed to obtain corresponding labels for the inputs sampled using the Sobol sequence method. This method allows the filling of the samples' space gradually and uniformly, as shown in figure 5.7, which demonstrates the first 10, 50, and 100 points sampled from 2D space with Sobol sequence. Therefore, additional shuffling of the collected dataset is not required. As discussed in subsection 5.2.2, the dataset consisted of $1,451,161$ samples. The last 10% of all samples in the dataset were used for the test set. The rest of the data was randomly split into training and validation datasets in proportion of 8:1. Figure 5.8 demonstrates the effect of the number of samples in the dataset on the performance of the model and confirms that the amount of the training data was adequate to train an accurate model (for brevity, only the study for σ_{11} is shown). Table 5.5 reports the test set root mean squared error for each ANN, and each error is converted back to the scale of each variable. The formula for

Table 5.5: Final architectures for the networks implemented in the framework. Architecture column displays the number of unit cells in each hidden layer. Test RMSE columns presented the error on the test set prediction.

Predicted variable	**Architecture**	**Test RMSE**
σ_{ij}, $i, j = 1 \dots 3$	$[2048 - 1024 - 1024 - 512 - 512 - 512 - 512 - 256 - 256 - 256 - 256 - 128 - 128 - 128]$	σ_{11}: $8.78E-02$ [MPa] σ_{22}: $8.34E-02$ [MPa] σ_{33}: $2.80E-02$ [MPa] σ_{12}: $4.34E-02$ [MPa] σ_{13}: $5.13E-02$ [MPa] σ_{23}: $1.02E-01$ [MPa]
a_{ij}, $i, j = 1 \dots 3$ g $\gamma^{(\alpha)}$	$[1024 - 1024 - 1024 - 512 - 512 - 512 - 256 - 256 - 256]$	a_{11}: $2.87E-02$ a_{12}: $3.06E-02$ a_{13}: $2.21E-02$ a_{21}: $1.98E-02$ a_{22}: $3.14E-02$ a_{23}: $2.72E-02$ a_{31}: $2.62E-02$ a_{32}: $1.87E-02$ a_{33}: $3.09E-02$ g : $1.04E-02$ [MPa] $\gamma^{(\alpha)}$: $1.36E-02$

RMSE is demonstrated by the equation 5.19, where N is the number of observations in a test set, y_i^{CP} is the prediction by crystal plasticity simulation, and y_i^{ANN} is the prediction by an artificial neural network, and i is the number of the sample in the test set.

$$RMSE = \sqrt{\sum_{i=1}^{N} \frac{\left(y_i^{CP} - y_i^{ANN}\right)^2}{N}} \tag{5.19}$$

For the stress-strain, texture, CRSS, and accumulated shear predictions, the test set prediction errors lies far within the 1% error gap, thus confirming the accuracy of each ANN.

Each artificial neural network was trained for a maximum of 800 training iterations. Initial learning rate, α, was 10^{-3}, decreasing by a factor of 0.95 in the case of no training

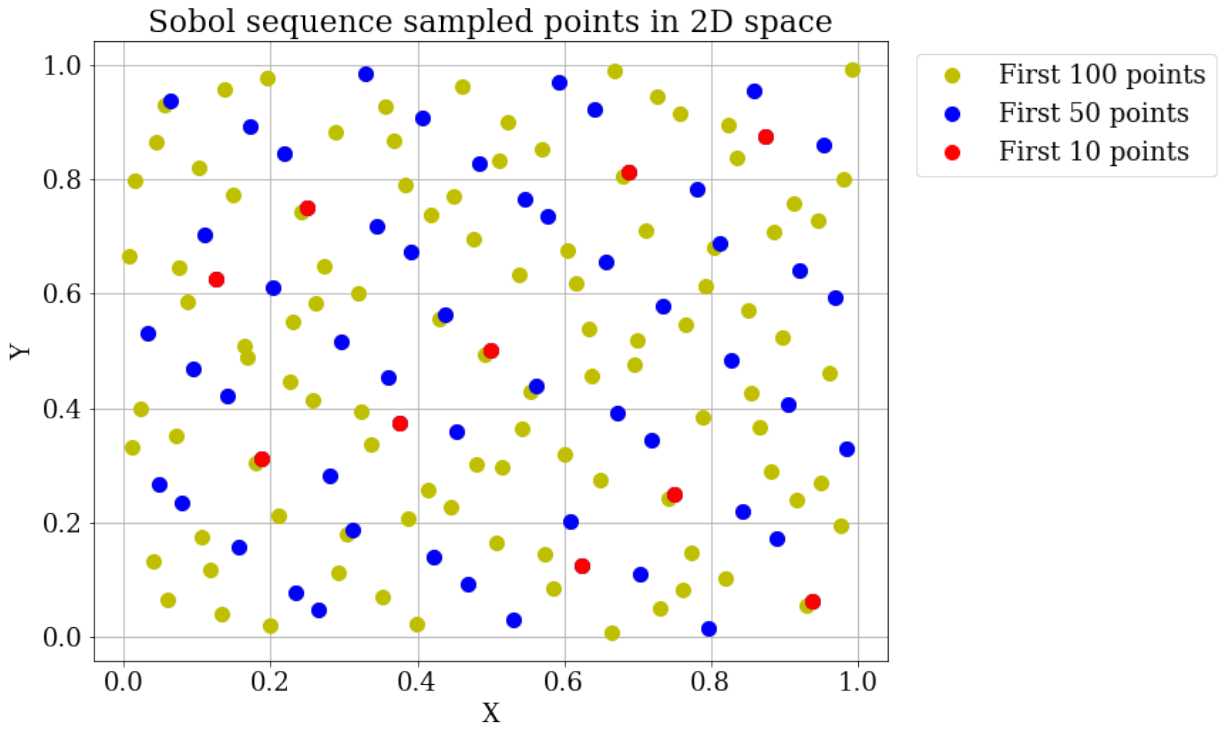

Figure 5.7: First one hundred points sampled with the Sobol sequence in 2D space. The sampled points cover the space evenly.

progress being made over five consecutive training epochs (i.e. number of passes of the training set). The other optimiser hyperparameters were set to their standard values: $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$. For each training, early stopping was used to stop training if the change in learning metrics did not exceed 10^{-8} over ten consecutive training iterations. The batch size (i.e. the number of training examples used in one iteration) of 2^{16} was used for training the networks. An approximate time to train each network was 150 minutes. After test errors validated the accuracy of ANNs, the test and validation set was also shuffled into training data, and the models were further trained for another 50 epoch. Finally, the obtained ANNs were used in the framework to predict the response and the computational inefficiency is compared with the CP model.

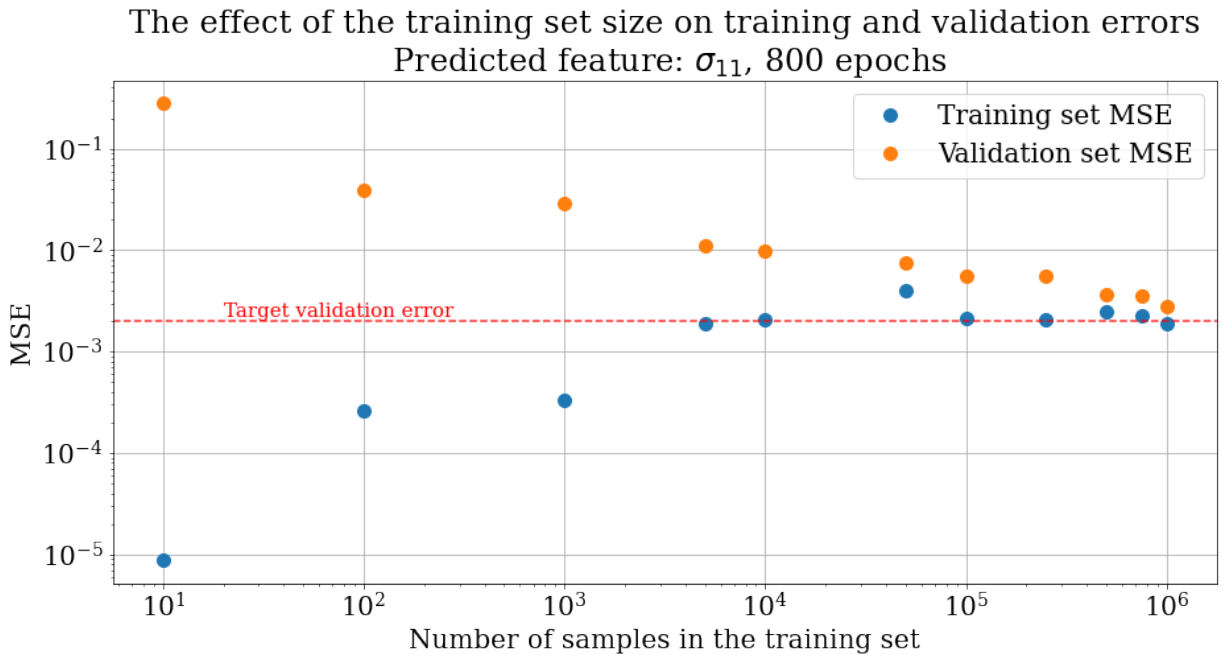

Figure 5.8: The effect of the amount of data on the performance of the network trained for prediction of σ_{11} feature. The X-axis presents the number of samples in the dataset used for training. Y-axis presents the MSE after 800 training epochs. Both axes are shown on a logarithmic scale. The selected dataset was randomly split into training and validation datasets in proportion of 8:2.

5.3 Results & Discussion

As discussed earlier, this framework is capable of predicting full stress-strain behaviour for any strain path and for any crystal orientation. This section presents predictions of the framework and verifies the framework performance against CP simulations. Crystal plasticity simulations were computed for different strain paths, both monotonic and non-monotonic, for different single crystal orientations as well as polycrystal ensembles, and were used to verify the predictions of the framework. In this section, the errors are quantified with the RMSE metrics, as it provides a comprehensive single number performance evaluation given in the same units as the measured variables. The RMSE was calculated across all points on each flow curve and then was averaged across all stress components. The RMSE for texture evolution was calculated using RMSE for each predicted timestep for each Euler angle, and then was averaged across all Euler angles. All the predictions

are shown graphically in figures. The subsequent subsections present the results for:

5.3.1 monotonic strain path predictions

5.3.2 non-monotonic strain path framework predictions

5.3.3 polycrystal deformation predictions for monotonic and non-monotonic strain paths

Finally, a runtime comparison between CP and the ANN was performed and is discussed in subsection 5.3.4.

5.3.1 Predictions for Monotonic Strain Paths

Once trained the ANN framework can be used to predict the flow behaviour for any arbitrary texture for any strain path. This subsection will demonstrate the ability of the framework to predict the texture evolution and as well stress evolution during monotonic loading. In addition, this will also demonstrate the ability of the framework to predict the evolved texture and stress state outside the bounds initial training data set. To illustrate the predictive capability different single texture components are chosen for each of the monotonic strain paths. The following single crystal orientations are used: Cube (0°, 0°, 0°) under uniaxial tension and simple shear, Goss (0°, 45°, 0°) under compression, and Copper (90°, 35°, 45°) under equibiaxial tension. For each strain path the total strain prescribed was equal to 0.3, except for the case of simple shear where the prescribed strain was equal to 0.4 for demonstration of texture prediction framework's capabilities. Note that the training data was limited to include a maximum of 0.05 strain.

Figures 5.9 and 5.11 show the uniaxial tension and uniaxial compression stress-strain predictions for a Cube-oriented and for the Goss-oriented crystal for crystal plasticity and the ANN framework respectively. The framework predictions show excellent agreement with CP simulation results. The total RMSE for the uniaxial predictions is 1.64 MPa, while that for the compression predictions is 2.02 MPa which confirms the accuracy of the framework. The error is also indicated in the table 5.8.

Figures 5.10 and 5.12 show the texture evolution for the cases of uniaxial tension and uniaxial compression respectively. During uniaxial tension and compression there is no significant texture evolution [202, 203] and, in both cases, this is accurately captured by the framework. The texture evolution prediction error is 10^{-4} degrees for both tension and compression; the error is reported in the table 5.8.

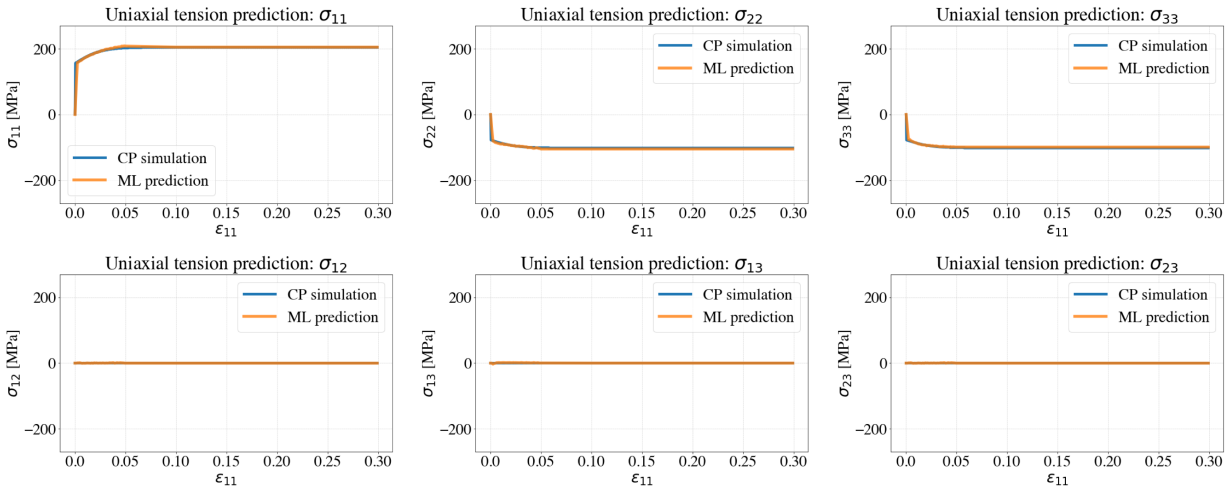

Figure 5.9: Comparison between crystal plasticity (CP simulation) and machine learning framework (ML prediction) single crystal stress-strain curves under uniaxial tension.

In order to obtain better understanding of framework capability to predict texture evolution, shear test was performed. Shear deformation adds more complexity in comparison to uniaxial tension and compression, as it significantly changes the texture throughout the deformation [204]. In total, 0.4 shear strain was prescribed to Cube single crystal. The evolution of texture is seen as a clockwise rotation of texture as shown in figure 5.14. The framework prediction is in excellent agreement with CP simulation. The prediction error for texture was evaluated at 0.38°, thus confirming the accuracy of the framework.

Stress-strain response prediction for simple shear was also performed and is displayed on the figure 5.13. The stress-strain predictions are also in good agreement with crystal plasticity simulation. The overall error of predictions is 1.94 MPa and is reported in the table 5.8. It is also noted, that simple shear stress-strain behaviour is more complex than in the case of tension and compression. The applied shear is in 12 direction. The framework captures sudden changes of stress in the end of the deformation for σ_{11} and σ_{22} components, as highlighted on the plot. Further, the framework was also able to predict the decrease in stress for σ_{12}. Most importantly, the shear stress σ_{12} was captured exceptionally well. Thus, the framework accuracy was confirmed as in the cases of uniaxial tension and compression.

The other strain path that has prominent effects on texture evolution is equibiaxial tension [205], and hence is also used to validate the framework performance. As in other cases, the predictions for stress-strain behaviour and texture evolution were performed.

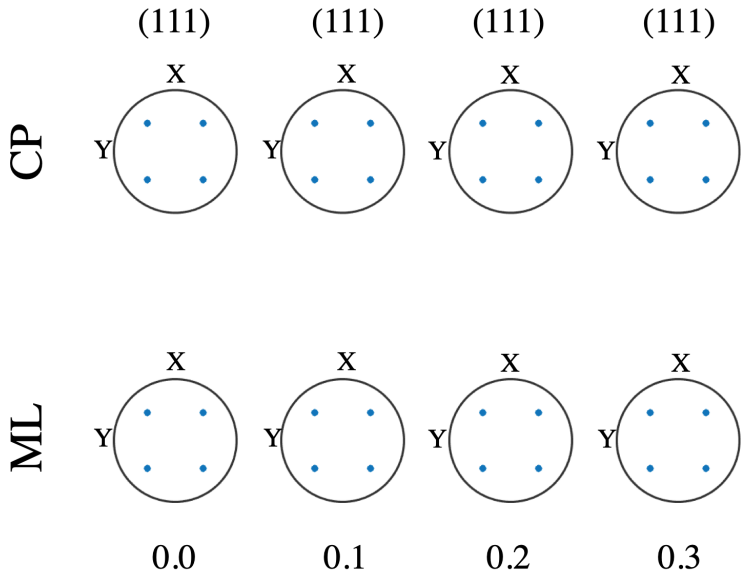

Figure 5.10: Comparison between crystal plasticity (CP simulation) and machine learning framework (ML prediction) Cube crystal $< 111 >$ pole figures displaying texture evolution under uniaxial tension. The result comparison is indicated for 0.0 (initial texture), 0.1, 0.2, and 0.3 values of ε_{11} engineering strain.

Figure 5.15 shows the crystal plasticity and the framework equibiaxial tension stress-strain predictions for a Copper crystal up to 0.3 for ε_{11} and ε_{22}. The overall stress-strain prediction error was evaluated at 2.47 MPa, and is reported in table 5.8. The framework was capable to capture the sudden stress changes in the beginning of deformation for σ_{22} and σ_{13} components of stress. Predictions for σ_{12} and σ_{23} are at zero, as the CP simulations for these components. The framework also exhibits remarkable accuracy in prediction of subtle decrease in stress for σ_{11} component and the increase for σ_{33} component, thus confirming the accuracy of predictions.

The prediction of texture evolution for the case of equibiaxial tension is exhibited on the figure 5.16. The overall error for predictions was evaluated at 0.63° and is reported in table 5.8. It can be seen that the texture undergoes some rotation, and is successfully captured by the proposed framework, thus, confirming its accuracy.

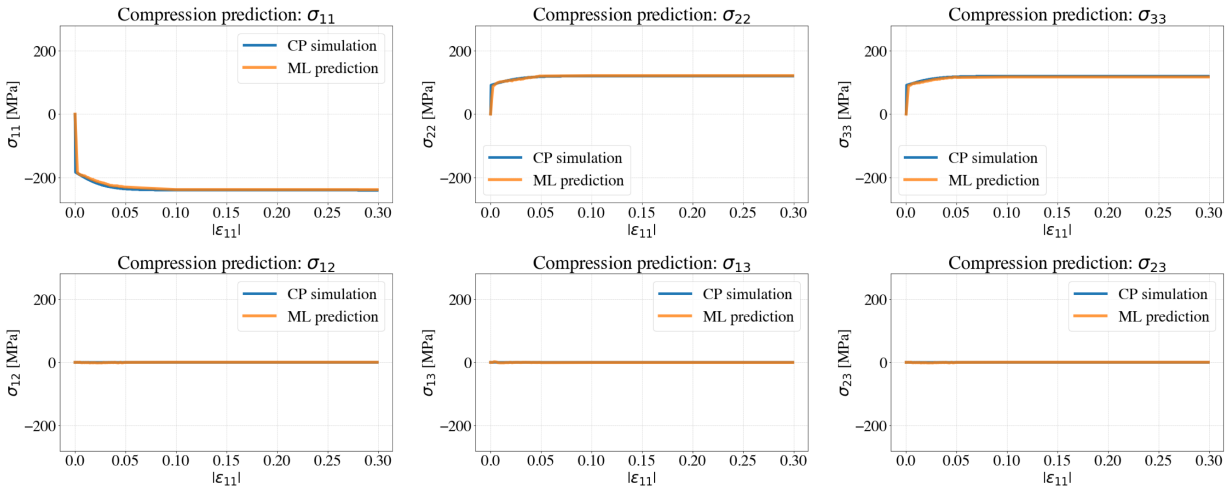

Figure 5.11: Comparison between crystal plasticity (CP simulation) and machine learning framework (ML prediction) single crystal stress-strain curves under compression.

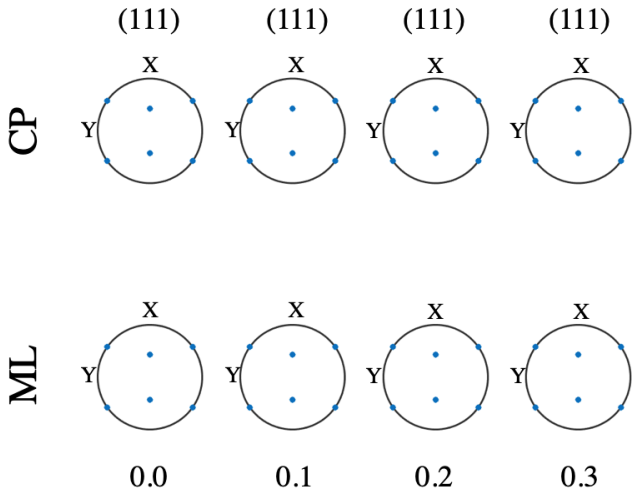

Figure 5.12: Comparison between crystal plasticity (CP) and machine learning framework (ML) Goss crystal $< 111 >$ pole figures displaying texture evolution under compression. The result comparison is indicated for 0.0 (initial texture), 0.1, 0.2, and 0.3 values of $|\varepsilon_{11}|$ engineering strain.

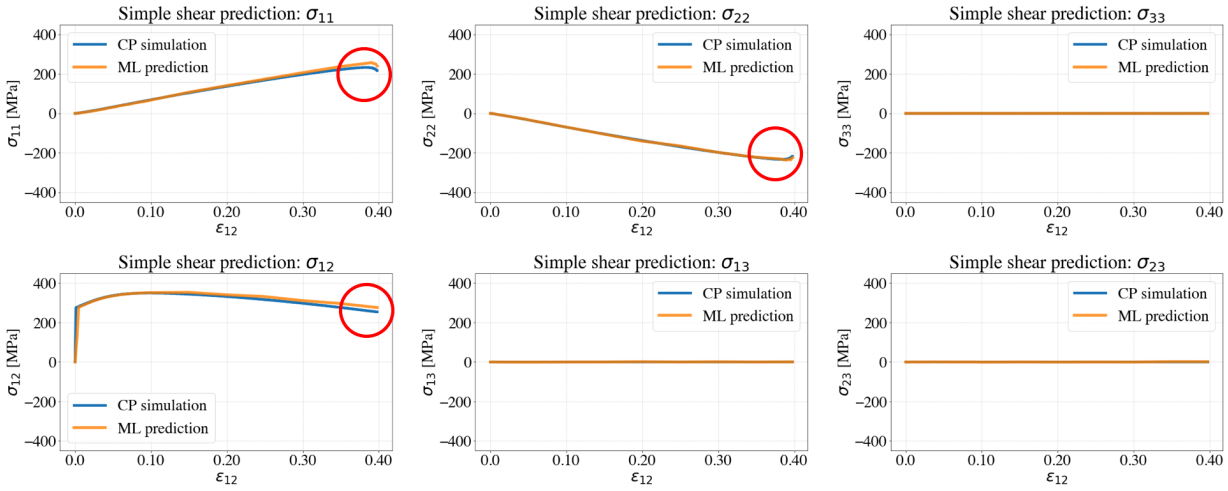

Figure 5.13: Comparison between crystal plasticity (CP simulation) and machine learning framework (ML prediction) single crystal stress-strain curves under simple shear.

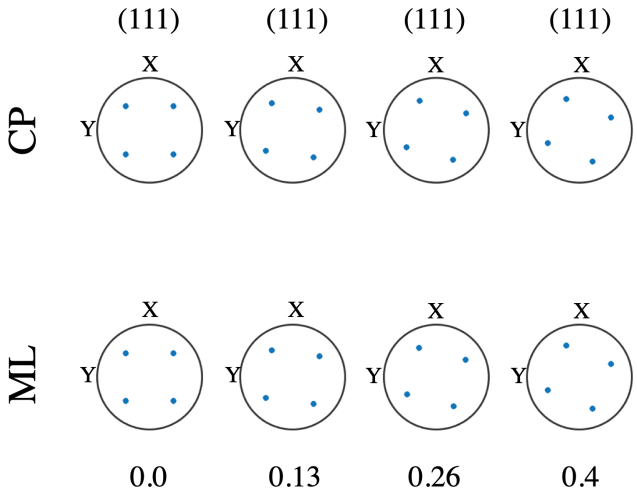

Figure 5.14: Comparison between crystal plasticity (CP) and machine learning framework (ML) Cube crystal $< 111 >$ pole figures displaying texture evolution under simple shear. The result comparison is indicated for 0.0 (initial texture), 0.13, 0.26, and 0.4 values of ε_{12} engineering strain.

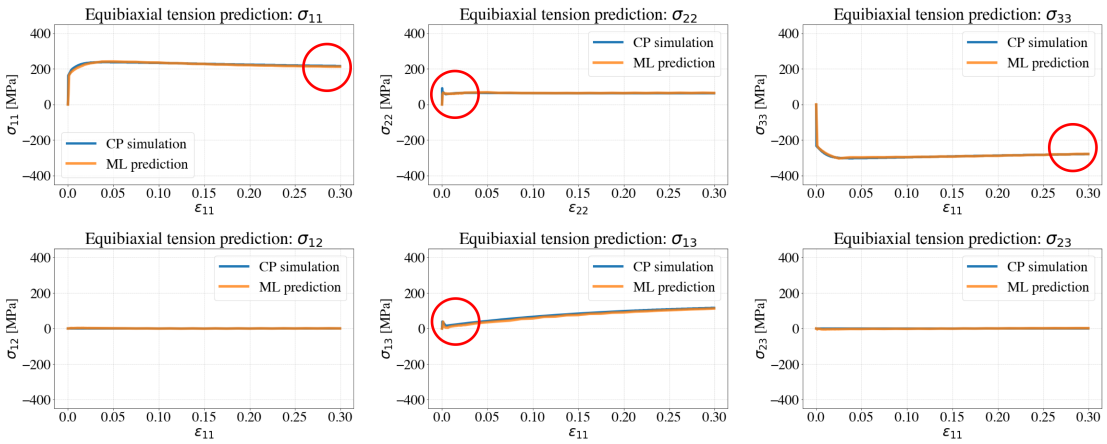

Figure 5.15: Comparison between crystal plasticity (CP simulation) and machine learning framework (ML prediction) single crystal stress-strain curves under equibiaxial tension.

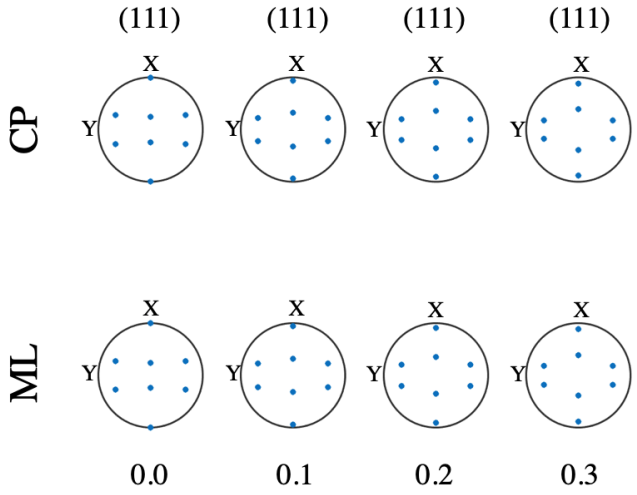

Figure 5.16: Comparison between crystal plasticity (CP) and machine learning framework (ML) Copper crystal $< 111 >$ pole figures displaying texture evolution under equibiaxial tension. The result comparison is indicated for 0.0 (initial texture), 0.1, 0.2, and 0.3 values of ε_{11} engineering strain.

5.3.2 Predictions for Non-monotonic Strain Paths

To further verify the framework, non-monotonic strain paths were chosen to demonstrate the predictive capabilities of the framework. The non-monotonic strain paths included three different cyclic loadings and an arbitrary non-monotonic loading. The arbitrary loading was also performed on the arbitrary crystal orientation. Note that, the ANN was never trained on data that is non-monotonic and the predictions are obtained using the update of CP parameters as outlined in the manuscript.

Figure 5.17 presents the crystal plasticity stress-strain response and the framework predictions for three different types of cyclic loadings. For brevity, only the main stress-strain components are displayed in this work. The results present tension-compression-tension (TCT) cycle simulated for Brass (35°, 45°, 0°) single crystal, compression-tension-compression (CTC) cycle simulated for Taylor (90°, 27°, 45°) single crystal, and cyclic shear was simulated Cube (0°, 0°, 0°). In all the cases, the results are in good agreement with crystal plasticity simulations. The overall prediction error for TCT loading was 5.52 MPa, the prediction error for CTC was 6.90 MPa, and for cyclic shear the error was 3.62 MPa. All the errors are summarised in the table 5.8. Despite the fact that the errors are higher that the errors for monotonic loadings, the stress-strain behaviour is captured accurately for all cases of cyclic loadings.

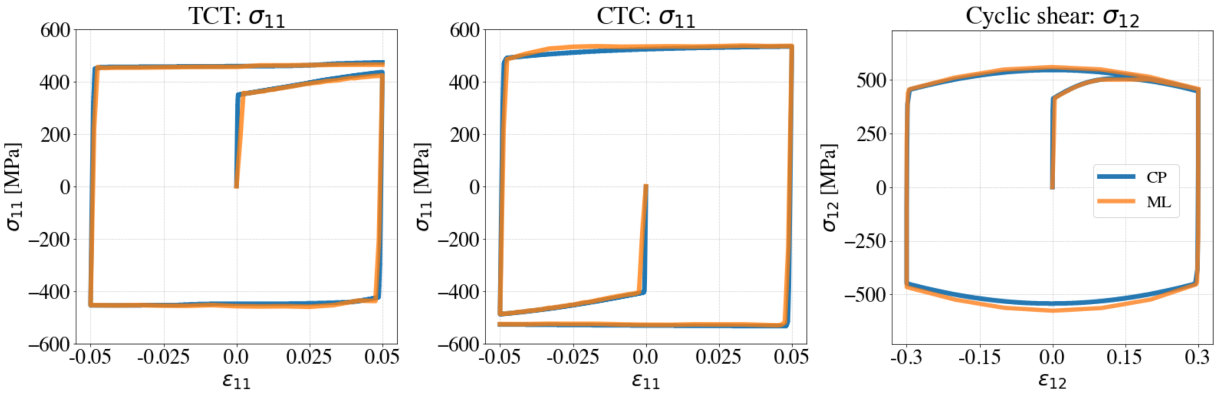

Figure 5.17: Comparison between crystal plasticity (CP simulation) and machine learning framework (ML prediction) single crystal stress-strain curves under TCT, CTC, and cyclic shear deformation modes.

The texture evolution predictions were also compared with CP predictions. Figures 5.18, 5.19, and 5.20 presents the texture evolution for TCT, CTC, and cyclic shear cor-

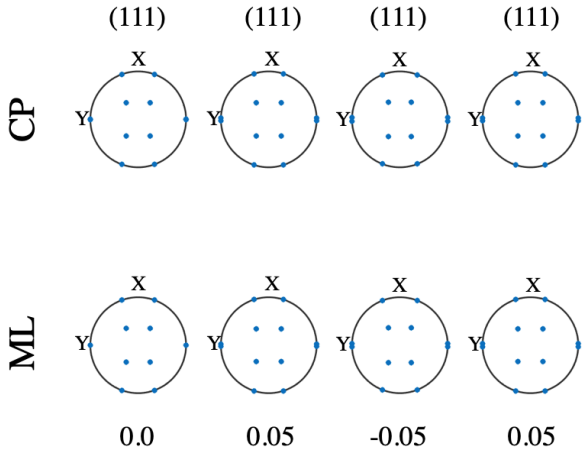

Figure 5.18: Comparison between crystal plasticity (CP) and machine learning framework (ML) Brass crystal $< 111 >$ pole figures displaying texture evolution under TCT deformation mode. The result comparison is indicated for 0.0 (initial texture), 0.05, -0.05, and 0.05 values of ε_{11} engineering strain.

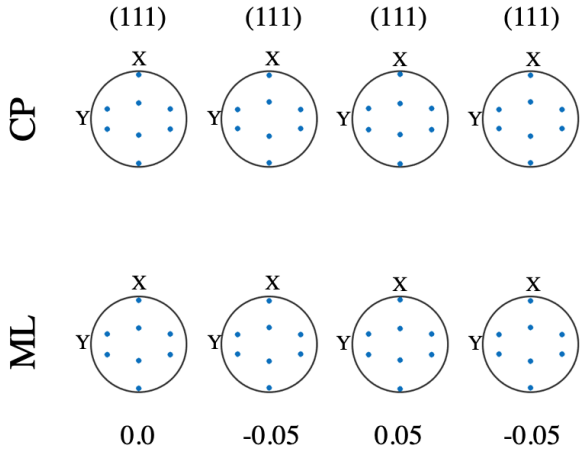

Figure 5.19: Comparison between crystal plasticity (CP) and machine learning framework (ML) Taylor crystal $< 111 >$ pole figures displaying texture evolution under CTC deformation mode. The result comparison is indicated for 0.0 (initial texture), -0.05, 0.05, and -0.05 values of ε_{11} engineering strain.

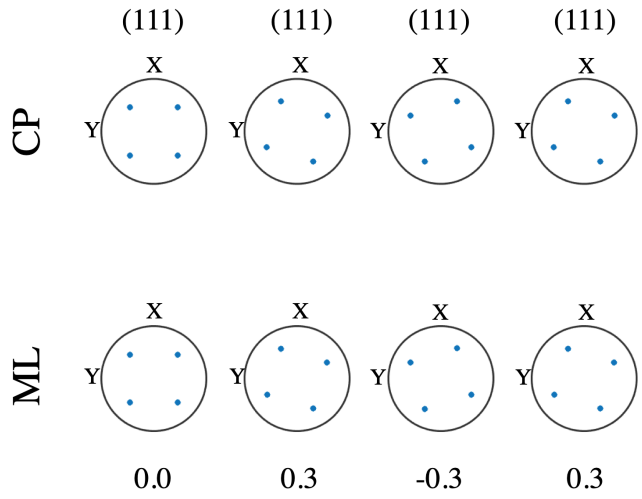

Figure 5.20: Comparison between crystal plasticity (CP) and machine learning framework (ML) Cube crystal $< 111 >$ pole figures displaying texture evolution under cyclic shear deformation mode. The result comparison is indicated for 0.0 (initial texture), 0.3, -0.3, and 0.3 values of ε_{12} engineering strain. A reference axis was included in the pole figures to help better observe texture rotation.

respondingly. In the cases of TCT and CTC the textures do not exhibit drastic change due to the nature of the strain paths. The predictions were accurate for both cases: the prediction error for TCT was 0.12°, and the prediction error for CTC was 0.23°. Figure 5.20 displays clearly visible rotation of texture components in the case of cyclic shear. The texture evolution is observed to rotate clockwise during forward shear, and then rotate counter clockwise when the shearing is applied in the other direction. These predictions also agree well with experimental data [204]. To highlight the importance of these results, it should be noted that no training on non-monotonic loading was performed, and the CP-update algorithm enables the framework to predict these non-monotonic strain paths. The error for cyclic shear texture prediction was evaluated at 0.34°, thus, confirming the accuracy of the framework. Error values for texture and stress are also reported in the table 5.8.

In order to fully verify the predictive capabilities of the framework, a randomly oriented crystal was simulated under a random strain path. The initial texture orientation was taken as (30°, 45°, 45°), and the arbitrary non-monotonic loading consisted of series of different loading and unloading steps. The applied loading path is summarized in table 5.6. The

Table 5.6: Strain path for the arbitrary loading (single crystal)

Strain-rate tensor, $[D_{11}, D_{22}, D_{33}, D_{12}]$	**Number of timesteps** ($\Delta t = 0.001$)
$[1.0, -0.5, -0.5, -0.3]$	50
$[0.7, 0.3, -1.0, 0.2]$	50
$[0.5, -0.3, -0.2, -0.25]$	50
$[0.7, -0.3, -0.4, -0.25]$	50
$[0.4, 0.3, -0.7, 0.25]$	50
$[-0.2, -0.6, 0.8, 0.0]$	50
$[0.0, 0.0, 0.0, 1.0]$	100

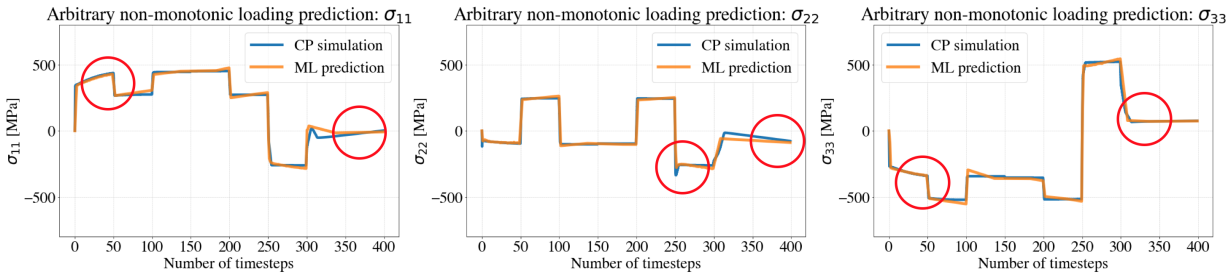

Figure 5.21: Comparison between crystal plasticity (CP simulation) and machine learning framework (ML prediction) single crystal stress-strain curves under arbitrary non-monotonic loading.

table shows strain path in the form of strain rate tensor and the number of timesteps with a fixed Δt. As can be seen in both table 5.7 and flow curve shown in figure 5.21, the single crystal is subjected to 7 different strain paths, chosen randomly. For this strain path, the total amount of equivalent strain during the deformation was equal to 0.3.

The CP simulation was compared with the framework predictions and the comparison is presented in figure 5.21. For brevity, only the principal stress-tensor components are displayed on the plot. The proposed framework was fully able to accurately capture all the subsequent changes in stress values, and results in accurate prediction of final value of stress. The overall prediction for stress values was equal to 3.56 MPa and is reported in the table 5.8. As a note, the results were obtained with the help of ML models that were trained on only monotonic loadings examples, and no additional training was performed. There is some inaccuracy in prediction, for example for σ_{11} component, but overall the stress-

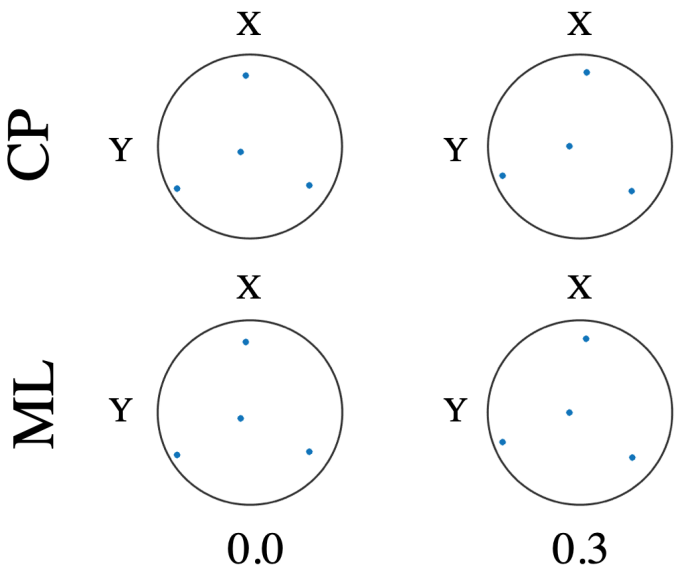

Figure 5.22: Comparison between crystal plasticity (CP) and machine learning framework (ML) $< 111 >$ pole figures for an arbitrary single crystal displaying texture evolution under arbitrary non-monotonic loading for a single crystal with random texture. The result comparison is indicated for initial and final equivalent plastic strain.

strain behaviour is captured very accurately. Finally, the texture evolution prediction for arbitrary strain path were also well-predicted by the framework. The texture evolution predictions compared with CP simulations can be observed on the figure 5.22. Due to the imposed shear strain, some rotation is present in the texture evolution, and it is well-predicted by the framework. The error for texture evolution prediction is $0.62°$, thus confirming the accuracy of the framework.

To conclude, in this subsection different non-monotonic strain path predictions were performed for different FCC single crystals. The results were in a good agreement with crystal plasticity simulations and have verified the accuracy of the framework. Therefore, the framework could be applied to predict the polycrystal behaviour under complex strain path. The polycrystal application is discussed in the next subsection.

5.3.3 Predictions for Polycrystals

Results for monotonic and non-monotonic strain paths demonstrated the predictive capabilities of the proposed framework. In this section, the framework was employed for prediction of a polycrystal material response. First, two distinct non-monotonic loadings were compared with CP simulations for a polycrystal constituent of 20% of Cube-oriented crystals, 10% of Goss-oriented crystals, 45% of S-oriented crystal, 20% of Copper-oriented crystals, and 5% of Brass-oriented crystals. Such polycrystal was chosen, as it consisted of common crystal orientations for FCC materials. Then, the framework was used to predict texture evolution of a polycrystal constituent of 246 randomly-oriented grains subjected to plane-strain compression. This case allowed to fully verify the predictive capabilities for texture evolution of the framework. To evaluate the accuracy of the framework, the corresponding CP simulation were performed and used for result comparison. To obtain the polycrystal stress-strain response Taylor type approach [161] was used.

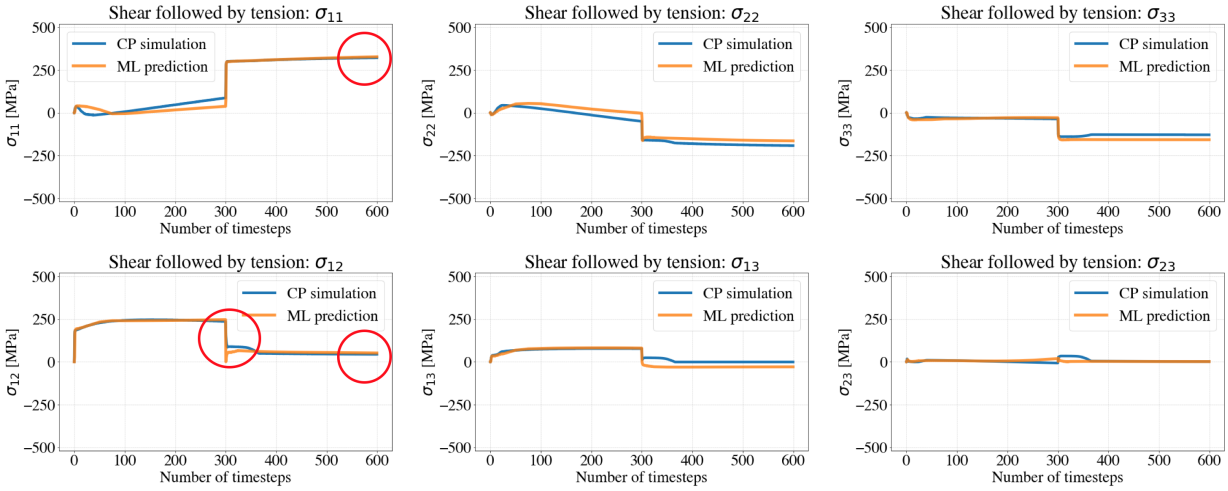

Figure 5.23: Comparison between crystal plasticity (CP simulation) and machine learning framework (ML prediction) stress-strain predictions for a polycrystal under shear followed by tension strain path.

In the first case, the polycrystal was loaded with simple shear up to 30% of shear strain, followed by uniaxial tension up to 30% of uniaxial strain. The deformation was simulated in 600 increments, with $\Delta t = 0.001$. The prediction comparison for all stress tensor components is displayed in figure 5.23. The overall prediction error was evaluated at 6.80 MPa, and is reported in the table 5.8. The predictions are in a good agreement with crystal plasticity simulations. The σ_{11} component of stress tensor is well-predicted, and the

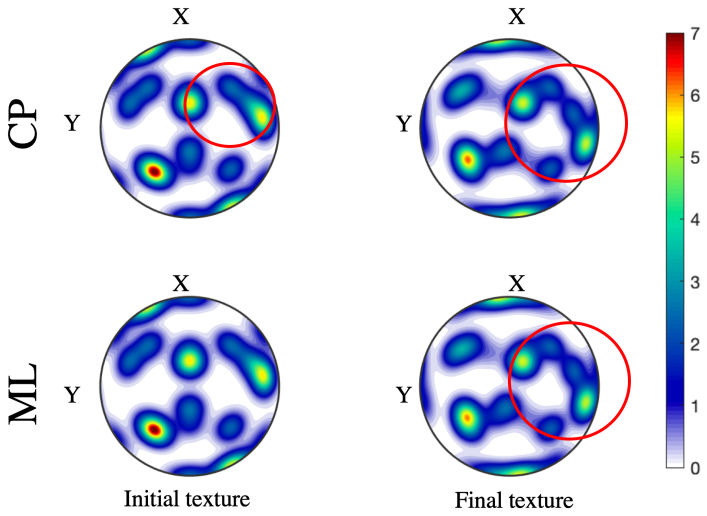

Figure 5.24: Comparison between crystal plasticity (CP) and machine learning framework (ML) $<$ 111 $>$ pole figures displaying texture evolution for a polycrystal under shear followed by tension strain path. The result comparison is indicated for initial and final equivalent plastic strain.

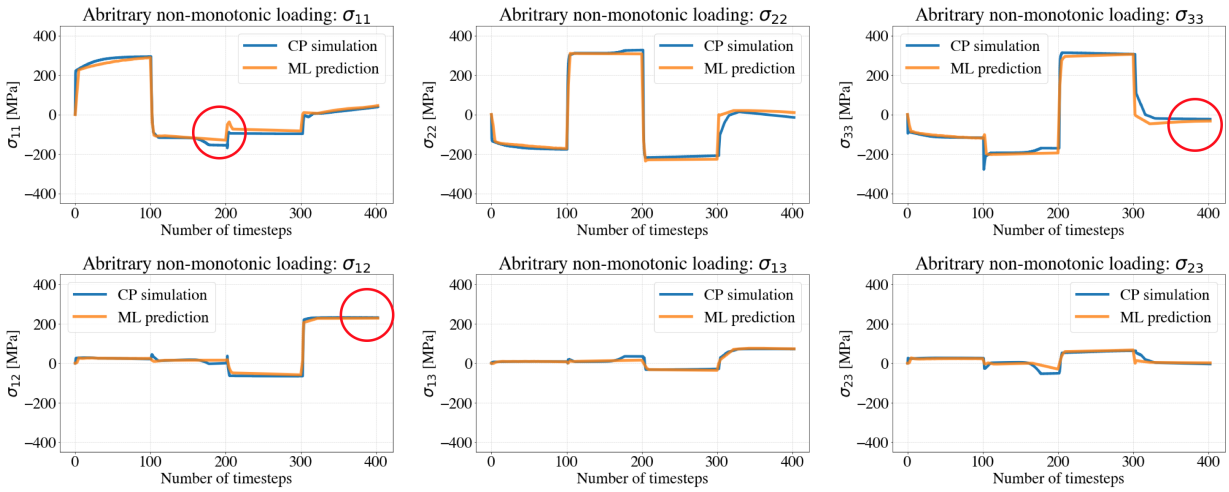

Figure 5.25: Comparison between crystal plasticity (CP simulation) and machine learning framework (ML prediction) stress-strain predictions for a polycrystal under arbitrary non-monotonic loading.

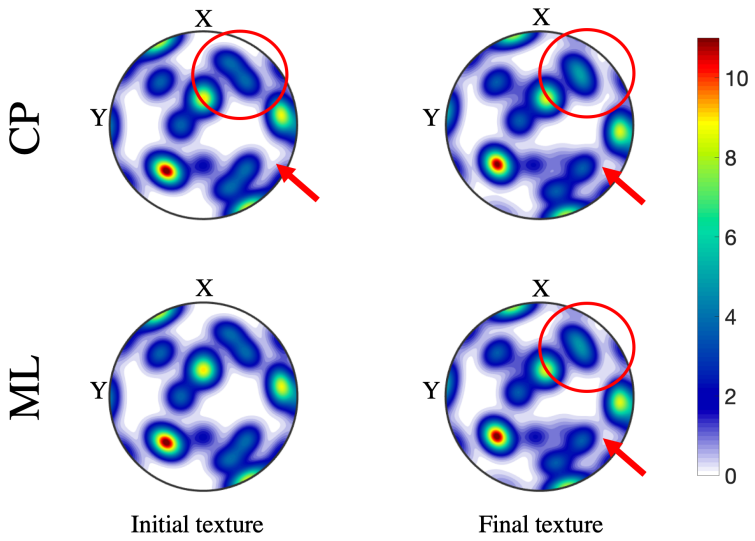

Figure 5.26: Comparison between crystal plasticity (CP) and machine learning framework (ML) $< 111 >$ pole figures displaying texture evolution for a polycrystal under arbitrary non-monotonic loading. The result comparison is indicated for initial and final equivalent plastic strain.

framework captures all the trends correctly. The tension part is predicted exceptionally accurately by the framework. As for the shear stress component, the predictions are in a excellent agreement with crystal plasticity: both shear and tension parts of the loading are predicted correctly. The rest stress components are also well-predicted in their trends and values.

The texture prediction is shown in figure 5.24. The prediction error for texture is 0.78°, and excellent agreement between the framework prediction and CP simulation is observed. Texture rotation is well-captured by the proposed framework, as well as other changes in texture, as indicated on the plot. The error for texture is also reported in the table 5.8.

In the second case, the arbitrary non-monotonic loading is applied on the same polycrystal. The exact loading path, that the polycrystal is subjected to, is summarized in table 5.7. The polycrystal is subjected to a total of 3 strain path changes with a different strain rate tensor for 100 timesteps, with each time step having a $\Delta t = 0.001$. Stress-strain comparison is performed for the framework prediction and CP simulation, and is presented in figure 5.25. The overall stress prediction error is equal to 5.31 MPa, and is reported in the table 5.8. The prediction results are in the good agreement with CP simulation. As indicated on the plot, the framework accurately captured the changes in stresses, result-

ing in accurate predictions. All the components for stress are captured very accurately, including the sudden changes in stress values for σ_{11}, σ_{12}, σ_{13} components, as indicated on the plot.

Table 5.7: Strain path for the arbitrary loading (polycrystal)

Strain-rate tensor, $[D_{11}, D_{22}, D_{33}, D_{12}]$	**Number of timesteps** ($\Delta t = 0.001$)
$[1.0, -0.5, -0.5, 0.0]$	100
$[-0.3, 1.0, -0.7, 0.2]$	100
$[-0.1, -0.9, 1.0, 0.25]$	100
$[0.0, 0.0, 0.0, 1.0]$	100

Texture prediction for the prediction of arbitrary loading case is also in a good agreement with CP simulation. Figure 5.26 presents the pole figures displaying the predictions. The texture prediction error was evaluated at 0.75°, and it is reported in the table 5.8. Due to imposed shear strain during loading, the rotation is visible on the pole figure and is captured by the framework. As indicated on the plot, the other details are also well-predicted by the framework.

Finally, texture evolution for a random polycrystal subjected to plane-strain compression with 50% reduction in thickness was predicted by the framework and compared to the CP simulation. This strain path was chosen as it is the major strain path during rolling which is a widely investigated process in sheet metal forming. The simulated polycrystal consisted of 246 randomly oriented single crystals. Texture prediction using ANN showed an excellent agreement with the crystal plasticity simulation, as demonstrated by the results given in figure 5.27. The framework successfully captures the development of two strong texture components, as pointed at in figure 5.27. The weaker texture band that can be observed in the middle of the final pole figure is also predicted accurately by the framework. The RMSE of the framework prediction in comparison to crystal plasticity simulation was 1.44° and is reported in table 5.8.

To conclude, the framework was validated for polycrystal prediction and has shown great predictive capabilities for stress and texture evolution prediction. The prediction were performed for monotonic and non-monotonic strain paths, and were in a good agreement with corresponding CP simulations. The ML models within the framework were trained on only monotonic loadings, and the implemented CP-update algorithm enabled the framework to perform accurate calculations for FCC-family crystals under any non-monotonic strain path.

Table 5.8: Errors for the proposed framework predictions.

Loading mode	**Averaged RMSE for stress-strain predictions (MPa)**	**Averaged RMSE for texture evolution (°)**
Uniaxial tension	1.64	0.0001
Compression	2.02	0.0001
Shear	1.94	0.38
Equibiaxial tension	2.47	0.63
TCT	5.52	0.12
CTC	6.90	0.23
Cyclic Shear	3.62	0.34
Arbitrary non-monotonic loading	3.56	0.62
Polycrystal (case 1)	6.80	0.78
Polycrystal (case 2)	5.31	0.75
Polycrystal (case 3)	–	1.44

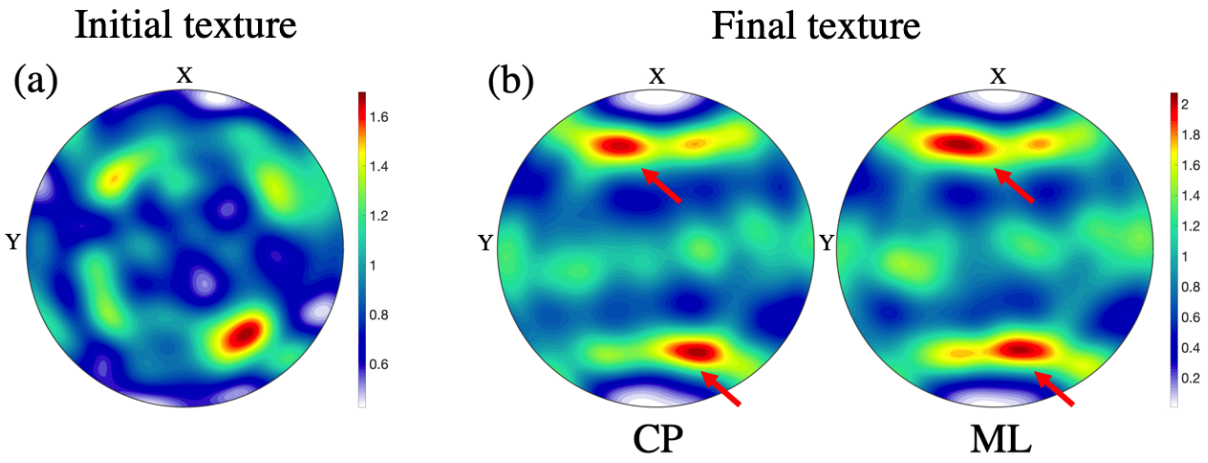

Figure 5.27: Comparison between crystal plasticity (CP) and machine learning framework (ML) $< 111 >$ pole figures showing texture evolution for a polycrystal with a random texture under plane-strain compression, (a) initial texture, (b) and CP and ML comparison for final texture after 50% thickness reduction.

5.3.4 Runtime Comparison

The results thus far demonstrated the predictive capability of the proposed framework. In addition, compared to crystal plasticity formulations, machine learning models are very efficient computationally. Crystal plasticity simulations, while being accurate in predictions of material behaviour [206], are computationally expensive due to solving highly-nonlinear differential equations. As mentioned in the introduction, there is a growing body of literature addressing different methods regarding acceleration of crystal plasticity. Therefore, the proposed framework and crystal plasticity runtime comparisons were performed.

Crystal plasticity models could require a substantial number of elements for accurate predictions [207]. In addition to that, an exponential increase in computational time with an increase in elements is observed [137]. Due to this fact, to perform the time comparison, crystal plasticity simulations were performed for $250,000$ individual crystal simulations. The crystal plasticity model simulation time was **1 hour and 20 minutes on 32 parallel processes**, each running in separate central processing unit core. The ANN framework only required one graphic processing unit, and computation time was **8.5 seconds**. Hence, the proposed framework results in a **computational time savings of over than 99.9**%.

To conclude, the results presented in this section highlight the high computational efficiency of the proposed framework. The framework, consisting of an ensemble of artificial neural networks and crystal plasticity based algorithm, has been shown to produce accurate crystal plasticity predictions of stress-strain properties and texture evolution of the deformed material, and demonstrated tremendous time savings in comparison to crystal plasticity model. One of the most important highlights of the demonstrated results was the ability to predict deformation behaviour on any strain path using the dataset constructed of monotonic-loading examples of deformation.

5.4 Chapter Conclusions

In this paper, a machine learning- and crystal plasticity-based framework is proposed to model stress-strain and texture evolution for a wide variety of the FCC family crystals under non-monotonic strain path. The primary goal of this work is to demonstrate the possibility to obtain accurate and fast crystal plasticity predictions for non-monotonic strain path with the aid of machine learning, while not needing to train the machine learning models on the infinite set of possible loading conditions. The approach of coupling crystal plasticity and machine learning is feasible for training, accurate, and is computationally efficient in its application. As far as the authors are aware, this is the first study

that demonstrates the machine learning-based framework capable of predicting the full flow stress behaviour and texture evolution of FCC crystals under any strain path for any initial texture configuration. The important observations and conclusions are as follows:

- Typically, machine learning methods are only useful in building the mappings that generalize predictions within the bounding box of the available dataset. These methods usually perform poorly on extrapolation tasks. The proposed framework consists of an ensemble of artificial neural networks trained only on monotonic loading examples. The implemented crystal plasticity-based algorithm makes it possible to predict the complex cases of non-monotonic loadings using those networks. The presented framework allows accurate material behaviour predictions on any, monotonic or non-monotonic, strain path. The predictions include full stress-strain predictions and texture evolution.

- The present application of ML for crystal plasticity involves training features of varied complexity. For instance, certain training features, such as the rotation matrix, evolve in a linear manner and other features, such as the stress tensor, can exhibit highly non-linear behaviour. This complexity and non-linearity can be modeled accurately using FFNN's with the proper choice of the number of layers, unit cells within each layer, and activation functions. Furthermore, the numerical implementation of FFNNs is relatively simpler compared to other complex methods such as recurrent neural networks.

- The proposed machine learning- and crystal plasticity-based framework has been demonstrated to accurately predict the stress-strain and texture evolution for different single crystals for the a wide variety of strain paths: uniaxial tension, compression, simple shear, equibiaxial tension, tension-compression-tension, compression-tension-compression, cyclic shear, and arbitrary non-monotonic loading.

- The framework demonstrated excellent predictive capabilities for polycrystal simulations. The two cases were considered in this research: shear followed by tension, and arbitrary non-monotonic loading. For the first case, the stress-strain prediction error was 6.80 MPa and the texture evolution error was 0.78°. For the second case, the stress-strain prediction error was 5.31 MPa and the texture evolution error was 0.75°. In both cases, the prediction error is acceptable and validates the accuracy of the framework.

- The framework demonstrated excellent ability to predict texture evolution under plane-strain compression which is the major strain path during rolling of sheet metals. The overall RMSE error between CP and ANN texture predictions was 1.44°,

confirming the capability and accuracy of the proposed framework to predict texture evolution during large strain plasticity applications.

- In other cases for monotonic and non-monotonic loadings predictions, the overall RMSE for stress-strain predictions did not exceed 10 MPa, and the overall RMSE for texture evolution did not exceed 1°. This error is of within acceptable values and, thus, verifies the accuracy of the proposed framework.

- In addition to accuracy, the framework is capable of prediction of subtle changes of values in stress behaviour. Texture predictions by the framework were in excellent agreement with CP simulations for most common orientations of FCC family crystals, and for random textures as well.

- The presented approach is computationally efficient and shows up to 99.9% computational speedup over conventional crystal plasticity model. As a direction to further research, the framework can be used to make predictions for a coupon- or component-level deformation, which was not feasible in the past due to high computational demands of the crystal plasticity model.

- One of the most important outcomes of the current work is the proof of feasibility: it is possible to utilize machine learning to capture the crystal plasticity predictions of stress-strain properties and texture evolution. The presented results suggest that machine learning methods can be successfully applied to perform crystal plasticity prediction of material deformation behaviour.

Chapter 6

Convolutional Neural Network-based Crystal Plasticity Finite Element Model for Aluminium Alloys and its Application for Finite Element Simulations of Real Material Microstructures

In this chapter, a machine learning framework is developed to achieve accurate local stress and strain evolution predictions for a wide range of material microstructures. The framework incorporates a single convolutional neural network trained to predict stress and strain partitionings of an aluminium microstructure under a proportional loading condition similar to uniaxial tension. Crystal plasticity finite element simulations for synthetically generated microstructures with defined material parameters were employed as a learning base for CNN model. The framework successfully implemented and validated against CPFEM for a new set of synthetic microstructures. The flexibility of the CNN was demonstrated by its validation for two microstructures of AA5754 and AA6061 aluminium alloys. The chapter presents a submitted manuscript of an article published in International Journal of Plasticity [1]

[1] International Journal of Plasticity X (2022): In submission.

Overview

Convolutional neural networks (CNNs) find vast applications in the field of image processing. This study utilizes the CNNs in conjunction with the crystal plasticity finite element method (CPFEM). This research presents a framework that enables CNNs to make rapid and high-fidelity predictions for materials under uniaxial tension loading. The inputs to the CNN model are material hardening parameters (initial hardness and initial hardening modulus), a global tensile strain value, and microstructure with varying number of grains, grain size, grain morphology and texture. This input selection allows performing simulations for a wide range of materials, as defined by microstructure and flow curves. The outputs of the CNN are the local stress and strain values. The proposed framework involves the following stages: feature engineering, generation of synthetic microstructures, CPFEM simulations, data extraction and preprocessing, CNN design and selection, CNN training, and validation of the trained network. The trained CNN was successfully demonstrated to predict local stress and strain evolution for the completely new dataset (test set) containing synthesised microstructures. The test set predictions were evaluated, and the median, worst, and best predictions were presented and discussed. Overall, the CNN demonstrated excellent agreement with CPFEM simulations, thus validating its accuracy. Then, the CNN was applied to predict the stress and strain evolution for AA5754 and AA6061 microstructures obtained using electron backscatter diffraction. These two microstructures were entirely new for the CNN and displayed size and grain morphology different from the synthesised microstructures. For both microstructures, the obtained stress and strain evolution predictions demonstrated excellent agreement with CPFEM simulations, thus confirming the flexibility of the trained CNN model. Then, the framework was extended to predict strain localisation and was evaluated on a AA6061 microstructure. The results presented in this research demonstrate a clear computational advantage of CNN without loss of accuracy. Finally, the research offers prospects for future advances.

6.1 Introduction

Metals are complex polycrystalline materials. Depending on its characterisation (texture, shape, size, and neighbours), each grain within a microstructure exhibits a distinct mechanical response. The continued advancements in material development and design require understanding the relationships between microstructure and flow behaviour. Microstructure constituents' effect on material performance remains an active topic of research. The developing computational simulation tools enable unravelling the impacts of material mi-

crostructure on behaviour, strength, and resistance to damage [208, 209, 210, 211], and allow for virtual design of materials with desired properties. Possible microstructures widely range in their variability due to the orientations and morphology of the grains. Therefore, there is a crucial requirement for a feasible simulation tool capable of predicting the mesoscale properties of metals subjected to a macroscopic loading.

A growing body of literature offers microstructure-based methods for modelling of material behaviour. Crystal plasticity finite element (CPFE) based methods are the most utilised due to their versatility in applications for the prediction of stress and strain partitioning, localisation, and failure for a large variety of materials. The crystal plasticity finite element method (CPFEM) is a mesoscale numerical tool that accurately predicts material deformation. CPFEM is a physics-based method that considers the complex nature of polycrystals, such as the orientation and morphology of the grains within an aggregate. The constitutive laws, kinematics, homogenization schemes and multiscale approaches are implemented within the CPFEM and are extended to different materials [151]. The CPFEM simulation tool can help achieve accurate predictions of materials response to various loading behaviours. Examples of such applications include prediction of shear bands due to dislocation [212, 213], twinning shear [214, 215], or of non-crystallographic nature [216], or due to transformation-induced plasticity [217, 218]. Shear bands prediction is significant, as they indicate possible places of damage initiation. The simulations can be performed in both two- [219, 220] and three-dimensions [221, 222] for different materials, such as steel [82, 223], aluminum alloys [224, 225], magnesium alloys [226, 227] and other materials [228, 229, 230]. The CP methods have proven to be a high-fidelity material modeling tool, and it continues to advance.

However, crystal plasticity models can be very computationally demanding. Different methods were implemented to expedite crystal plasticity simulations, such as: parallel computing [161], graphics processing unit-acceleration of CP [231, 232], using the wavelet transformation-based algorithms [162, 163], and a spectral database method [165, 233]. A spectral database method is a data-driven approach that uses discrete Fourier transform. This method helped to improve computational time in crystal plasticity computations significantly and have been demonstrated by Kalidindi and co-workers in numerous studies [234, 233, 167]. Kalidindi and co-workers have also presented another database approach, "Material Knowledge System", which enables accelerated establishment of material properties [97, 235, 236]. New efforts that have been used to expedite CP simulations involve machine learning (ML) methods. Instead of being explicitly programmed these methods present a function approximation tool; they capture non-linear dependencies in the datasets they are trained on [45]. The outstanding property of such methods is that a trained model is usually not computationally demanding, especially in comparison to numerical methods

used to solve highly-nonlinear partial differential equations used in CP integration schemes. Prior to applications in CP, ML methods have been extensively explored in phenomenological modeling and computational material science. One of the earliest ML applications in material modelling was presented in 1991. In [109], the authors enabled an artificial neural network (ANN), namely multi-layer perceptron, to predict the flow curves of concrete in the state of plain stress under monotonic biaxial loading and compressive uniaxial cycle loading. The application demonstrated prominent results and ML has been used in computational material science ever since. For example, in [237], the authors demonstrated an ANN-based material model to predict the stress-strain behaviour of composites under monotonic and cyclic loadings. In [238], the authors used ANNs to predict flow stress of carbon steel under certain processing conditions. Many other examples of neural networks applications were found in computational material science, for example for: prediction of strengthening metal matrix composites [239], prediction of flow behaviour of steel [240], design optimisation in metal forming [241], flow behaviour of particle-reinforced aluminium [242], prediction of cold rolling texture in steel [113]. One of the notable recent publications described the use of machine learning, in particular recursive neural networks, to predict path-dependant material behaviour [10]. In [111], the authors demonstrated the capability of feed-forward and recursive ANNs to predict stress-strain behaviour for various strain paths. ANNs also found their application in the developing field of additive manufacturing and were demonstrated to successfully predict strain partitioning during deformation in aluminium alloys manufactured using selective laser melting method [115].

A growing body of literature has investigated ML applications to CP models acceleration. Mangal and Holm demonstrated the power of a random forest classification algorithm to predict stress hotspots development in face-centred cubic (FCC) and hexagonal close-packed synthetic three-dimensional microstructures [134, 135, 136]. In their work, the authors determined whether a grain was "hot" or not depending on a number of microstructural parameters; the significance of the input features was also discussed. ANNs were applied in CP to predict flow behaviour of an aluminium alloy under complex strain path of tension followed by simple shear [137]. In [142], the authors used ANNs to predictions flow behaviour and final deformation texture. The predictions could be achieved for four loading conditions (compression, tension, shear, rolling), twelve initial textures, for loading rate, and a range of Voce hardening parameters. The other applications presented: prediction of cyclic stress behaviour in steels [138], relating crystal plasticity variables to the texture of titanium alloy under uniaxial tension [140], relating microstructure characterisation to yield stress of steel and aluminium [243], prediction of texture evolution of copper under uniaxial tension [244]. In their recent work, the have utilised a feed-forward neural network method trained only on single-crystal simulations under monotonic load-

ings [245]. The trained ML- and CP-based algorithm predicted the stress-strain behaviour and texture evolution for any complex strain path for any FCC crystal orientation. The framework was successfully validated against for multiple strain path for individual grains and a polycrystal. This work was very significant, as it is has shown that having a limited dataset of monotonic loadings, simulated up to 0.05 per cent of strain, can be utilised to significantly expedite CP simulations for arbitrary complex strain-paths up to any strain value. The study has demonstrated the extreme usability of ML tools for computational time savings, which is vital in modern material design.

A limited number of studies show applications of machine learning in crystal plasticity to produce finite element (FE) modelling simulation results. Convolutional neural networks (CNNs) are instrumental in making two- or three-dimensional predictions or making predictions based on two- or three-dimensional input values. In particular, in [143], the authors used three-dimensional convolutional neural networks to identify the relation of a microstructure of high-contrast composite material to the elastic property in the form of a component of the effective elastic stiffness tensor. In [131], the authors used a CNN with residual connections for CP predictions. For their dataset, they performed discrete dislocation dynamics simulations for samples deformed under uniaxial tension with varying loading orientations. Their simulations were used to generate a strain map to replicate digital image correlation. Based on these strain maps, the authors were able to predict the deformation level of a sample and achieve statistical prediction of a sample's flow curve using saliency maps of hidden layers of the trained CNN. In [11], the authors used CNNs to predict macroscopic stress at a strain close to a yield point based on the input image of dual-phase microstructure with varying volume fracture of phases. In [145], the authors utilised a hybrid of convolutional and recurrent neural networks for application in oligocrystals of an annealed austenitic stainless steel. Their work utilised the initial microstructure and the strain values as an input to the machine learning model to predict the resultant stress corresponding to the input strain, up to 6 per cent strain. In the other application for oligocrystalline materials, [12] utilised a similar hybrid model to predict the evolution of the dominant stress component up to 0.3 per cent strain. The input image represented the initial undeformed microstructure of synthetically generated oligocrystals. Their work demonstrated the capability of CNNs to predict overall stress partitioning based on the microstructure input for up to a limited strain value. As seen in the literature, the capability of machine learning applications, particularly convolutional neural networks and their hybrid models, is yet to be investigated in their full-field predictions capabilities.

This paper proposes a convolutional network model for rapid full-field stress and strain evolution predictions for FCC polycrystalline materials. Section 6.2 presents the methods used in this research and describes the proposed framework. The framework in-

cludes: CPFEM model for generating the simulations used for training, feature engineering, methodologies for synthetic microstructure generation, the hardening parameters sampling, and the background behind CNNs. Section 6.3 presents a CNN hyper-parametric study for finding an optimal architecture. Then the section validates the selected model using a test dataset with artificially synthesised microstructures that were unseen during training. Section 6.4 validates the model's flexibility for applications for real materials. The CNN was validated for AA5754 and AA6061 microstructures. Next, the CNN was assessed for prediction of strain localisation. The section also presents the runtime comparison of CNN and CPFEM models. Finally, section 6.5 summarises the work and presents the conclusions.

6.2 Methods & Development of Machine Learning Framework

In this work, the CNN-based framework which enables rapid full-field stress and strain evolution prediction is proposed. The resultant CNN is trained on CPFEM dataset and accurately predicts local stress and strain evolution for an input microstructure of a given material with defined hardening parameters. Generalised workflow of the proposed framework is displayed in figure 6.1. This section includes:

- Brief description of the CPFEM model used for dataset generation,
- Methodology for artificial microstructures synthesis and sampling material hardening parameters,
- Data preprocessing methods,
- CNNs and training process background.

6.2.1 Crystal Plasticity Constitutive Model and its Finite Element Implementation

Crystal plasticity constitute models are high-fidelity models used to simulate material flow as well as stress and strain partitioning behaviour. The rate-dependant formulation, first introduced by Asaro and Needleman [36], was used to generate CP finite element

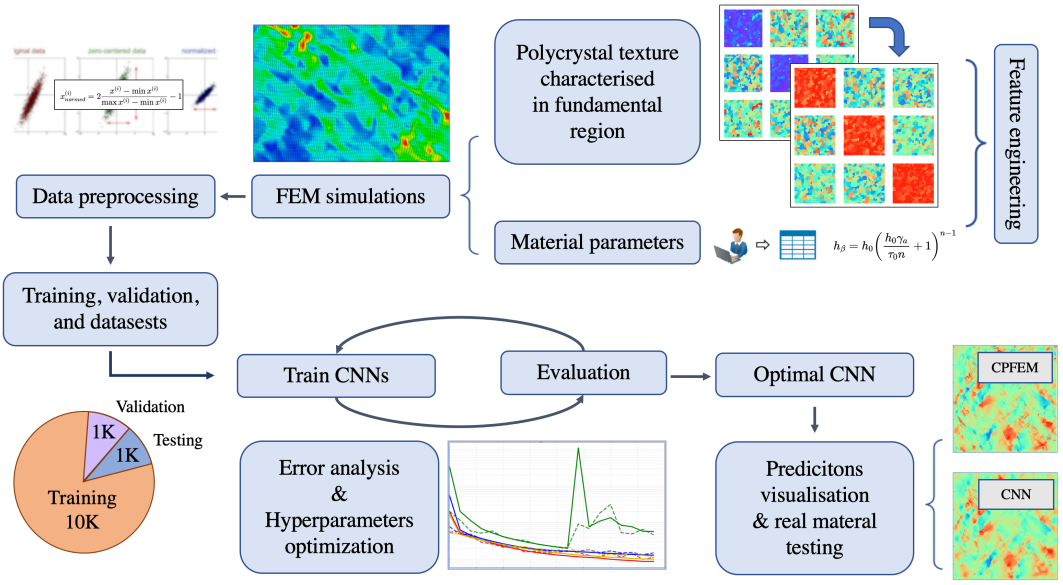

Figure 6.1: Schematic representation of the proposed framework

dataset for this study. In this formulation, the plastic deformation occurs due to the mechanism of crystallographic slip. For an FCC metal, e.g. Aluminum, slip occurs along three unique $< 110 >$ slip directions and in the four unique [111] slip planes. Consequently, there are twelve unique slip systems for materials with FCC crystal structure. The elastic constitutive equation is expressed as a function of the Jaumann rate of Cauchy stress, $\overset{\nabla}{\sigma}$, the strain-rate tensor, D, the elastic stress tensor, $\mathcal{L}$, and the viscoplastic stress state $\dot{\sigma}^0$:

$$\overset{\nabla}{\sigma} = \mathcal{L}D - \dot{\sigma}^0 - \sigma \operatorname{tr} D \tag{6.1}$$

The slip rates for each slip system, α, are found using the equation:

$$\dot{\gamma}^{(\alpha)} = \dot{\gamma}_0 \operatorname{sgn} \tau^{(\alpha)} \left| \frac{\tau^{(\alpha)}}{g^{(\alpha)}} \right|^{1/m} \tag{6.2}$$

where $\dot{\gamma}_0$ is the reference shear rate, $\tau^{(\alpha)}$ is the resolved shear stress on the slip system, m is the strain-rate sensitivity index, and the parameter $g^{(\alpha)}$ is related to the slip system hardness.

The single crystal work hardening is calculated using the equation:

$$\dot{g}^{(\alpha)} = \sum_{\beta} h_{\alpha\beta}\dot{\gamma}^{(\beta)} \tag{6.3}$$

where $h_{\alpha\beta}$ are the hardening moduli described as:

$$h_{\alpha\beta} = q_{\alpha\beta}h_{\beta} \quad \text{(no sum on } \beta) \tag{6.4}$$

where $q_{\alpha\beta}$ is related to latent hardening behaviour of a crystal, and h_{β} is the rate of a single crystal hardening. The latent hardening expressed in a form of a matrix:

$$q_{\alpha\beta} = \begin{bmatrix} A & qA & qA & qA \\ qA & A & qA & qA \\ qA & qA & A & qA \\ qA & qA & qA & A \end{bmatrix} \tag{6.5}$$

where A is a 3×3 matrices fully populated with ones, and q is the ratio of latent hardening to self hardening and is set to one.

Single slip hardening used in this work is governed by a hardening law is expressed as a power-law:

$$h_{\beta} = h_0 \left(\frac{h_0 \gamma_a}{\tau_0 n} + 1 \right)^{n-1} \tag{6.6}$$

where h_0 is the initial hardening rate, τ_0 is the initial critical resolved shear stress (CRSS) and n is the hardening exponent. This hardening model was chosen as it is commonly used in crystal plasticity simulations.

The stress update is formulated with equations 6.1-6.6. The texture evolution is updated using the algorithm presented in [246]. The initial orientation of the crystal is fed as an input to model, and then updated for each time step, in a form of a rotation matrix:

$$\boldsymbol{a}\left(\varphi_1, \Phi, \varphi_2\right) = \begin{bmatrix} \cos\varphi_1 \cos\varphi_2 - \sin\varphi_1 \sin\varphi_2 \cos\Phi & \sin\varphi_1 \cos\varphi_2 + \cos\varphi_1 \sin\varphi_2 \cos\Phi & \sin\varphi_2 \sin\Phi \\ -\cos\varphi_1 \sin\varphi_2 - \sin\varphi_1 \cos\varphi_2 \cos\Phi & -\sin\varphi_1 \sin\varphi_2 + \cos\varphi_1 \cos\varphi_2 \cos\Phi & \cos\varphi_2 \sin\Phi \\ \sin\varphi_1 \sin\Phi & -\cos\varphi_1 \sin\Phi & \cos\Phi \end{bmatrix} \tag{6.7}$$

where $(\varphi_1, \Phi, \varphi_2)$ are the Euler angles in Bunge's notation.

The presented equations form the essence of the CP model. Their solution is very computationally demanding, especially when solved for within finite element formulation. The presented CP model was incorporated into a two-dimensional FE framework [39, 40, 41], and its in-detail description can be found in [40, 86].

The Lagrangian formulation of the field equations was used in the basis of the FE algorithm with the usage of convected coordinates [40]. The elements of the FE mesh used in this research were akin to those presented in [42]. These elements have quadrilateral shape and each element contains four linear velocity triangular sub-element in order to employ a higher order integration scheme.

For the current application, the example of a sheet of aluminium alloy subjected to plane stress condition. The dimensions of the FE mesh were set to 100×100 elements. The loading ρ is imposed on the edges of FE mesh based on the following relationship:

$$\frac{\dot{\varepsilon}_{11}}{\dot{\varepsilon}_{22}} = \rho \tag{6.8}$$

where $\dot{\varepsilon}_{11}$ and $\dot{\varepsilon}_{22}$ are the logarithmic principal strain rates. The applied loading is similar to the uniaxial tension applied along rolling direction, and corresponds to $\rho = -0.5$. Plane stress condition was satisfied by assigning σ_{33} to zero. The computations are performed at a constant timestep $\Delta t = 0.001$ for 2500 timesteps. Figure 6.2 presents a schematics of a finite element mesh undergoing proportionate loading. The presented constitutive model and its FE implementation is used within an in-house code and was employed to generate the training dataset presented in the subsection 6.2.2.

6.2.2 Feature Engineering and Dataset Generation

Dataset plays an essential role in training an accurate ML model, and it is imperative that it has a sufficient amount of information representative to the set task [183]. To achieve the goal of prediction full-field stress and strain evolution, the minimum necessary parameters were selected as an input to the CNN model: global strain value ε_{11}^M, material hardening parameters, and the microstructure of the analysed material.

Due to the implementation of a constant timestep within the CPFEM model, the uniaxial macroscopic strain ε_{11}^M (volume averaged) evolution is the same for all performed simulations. After 2500 steps of loading, the value of the macroscopic strain, ε_{11}^M, was approximately 22%, as summarised in the table 6.1.

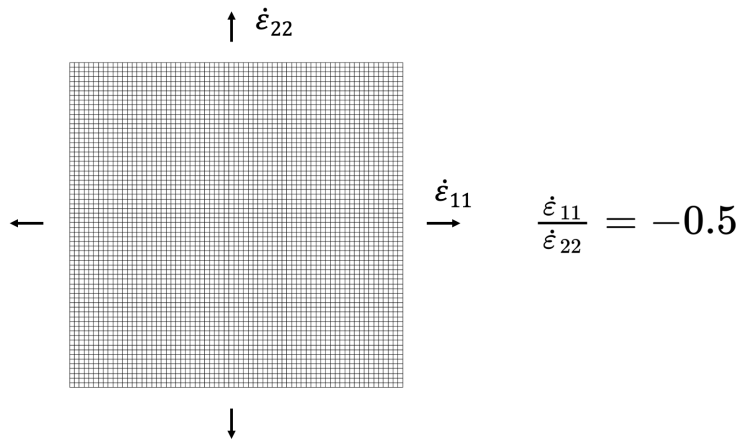

Figure 6.2: Schematic representation of unit FEM cell.

Then, the input material parameters were taken as the ones used in the hardening law (equation 6.6): the initial hardness, τ_0, and the initial hardening modulus, h_0. Their values were taken in the proximity of the hardening parameters for aluminium alloy AA5754 [206]. The initial hardness, τ_0, values were sampled within a range from 10 to 15 MPa, and the initial hardening modulus, h_0, values were sampled within a range 245 to 260 MPa, as summarised in table 6.1. Sampling strategy for these material parameters was implemented using Sobol sequences [126, 127]. This sampling methodology enables an optimal filling of the chosen data space. The details about the implementation of this algorithm can be found in [186]. This sampling method's main advantage is in its low discrepancy property, i.e. all the sampled points in a material parameters space are sampled in an almost equidistributed fashion. In addition, this method samples the data points uniformly and quasi-randomly across the domain, which ensures that some point in the finite proximity of each combination of material parameters will be included in the training data inputs, which is beneficial for training an accurate machine learning model.

Similar to the case of hardening parameters, the microstructures, which is another input to the CNN, were selected from all possible textures in the direct-chill cast AA5754 aluminium alloy. To achieve this, the orientations within microstructures were sampled from the experimentally measured textures. The experimentally measured textures were obtained using electron backscatter diffraction (EBSD) AA5754 map. The obtained set of individual textures contained $8,750$ different grain orientations. The texture data is presented in figure 6.3 displaying, (a), the extracted EBSD map, and (b) presents the pole

figure of the experimentally obtained texture. Due to the symmetrical nature of crystals within FFC materials, the crystal orientation space can be decreased to the fundamental region [30]. For FCC materials, crystal orientations exhibit full octahedral symmetry, which means that each crystal orientation can have 48 different texture representations. All the orientations outside of the fundamental region zone were converted to their corresponding values within that zone. Therefore, the crystal orientations were only sampled within the fundamental region, which decreased the diversity of texture representations. The diversity of the microstructures (i.e. grain size and grain morphology) for training set was achieved using the Euclidean distance-based Voronoi tessellations algorithm. The resultant microstructures contained between 220 to 320 grains, which is an adequate number for information on microstructural and textural characteristics representing the sheet [206]. The obtained grain morphologies were implemented into the FE mesh with the size of 100×100 elements. After the grain structure was meshed with N grains, N texture orientations were randomly sampled from the set of textures obtained from the AA5754 EBSD map, assigned to those grains, and stored in the form of rotation matrices. Each rotation matrix is characterised by nine components (equation 6.7). The material microstructure was used an input to CNN, as summarised in the table 6.1.

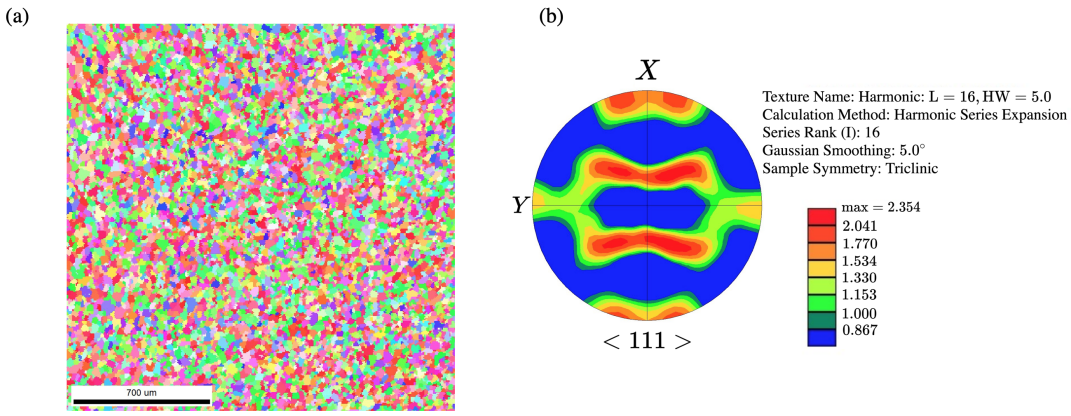

Figure 6.3: The extracted data for AA5754 aluminium alloy (a) EBSD map (b) pole figure of the microstructure.

The dataset contained $12,000$ different simulations performed on unique inputs. The output of the CPFEM model were local stress and strain values (i.e. elemental values) for FE mesh, which were taken as outputs to the CNN model. Each 100^{th} increment's data (of total 2500, as mentioned in subsection 6.2.1) was saved for training, yielding 25

deformation timesteps saved for each simulation. The collected data was split into input and output sets. The input features included: material hardening parameters (initial hardness and initial hardening modulus), initial microstructure, macroscopic strain value ε_{11}^{M}. This is the minimal set of features that could be used for training a machine learning model, as hardening parameters regulate the material behaviour, and initial microstructure is an essential descriptor of a material under consideration. Therefore, the feature space cannot be reduced to fewer features. The output features included local stress and strain partitionings corresponding to the input macroscopic strain value ε_{11}^{M}. The input features are summarised in table 6.1. The output features are summarised in table 6.2. Due to the plane stress condition, the stress partitionings were presented for σ_{11}, σ_{22}, and σ_{12} stress tensor components, and the strain partitioning were presented for ε_{11}, ε_{22}, ε_{33}, and ε_{12} strain tensor components.

Table 6.1: Input features, their ranges, and description of the feature and the physical parameters it controls. In total, there are 12 distinct features that serve as input to the CNN model: macroscopic strain value, initial hardness, initial hardening modulus, and a microstructure.

Input variable	**Range**	**Description**
ε_{11}^{M}	$0.0 \ldots 0.22$	Macroscopic strain
τ_0	$10.0 \ldots 15.0$ MPa	Initial hardness
h_0	$245.0 \ldots 260.0$ MPa	Initial hardening modulus
$\mathrm{a}_{0_{\mathrm{ij}}}$, $i, j = 1 \ldots 3$	$-1.0 \ldots 1.0$	Initial rotation matrix for texture

Table 6.2: Output features and their description. In total, there are 7 distinct features that serve as output to the CNN model: stress partitioning (each FE element is characterised by 3 components of stress tensor), and strain partitioning (each FE element is characterised by 4 components of strain tensor).

Predicted variable	**Description**
σ_{ij}, $ij = 11, 22, 12$	Stress response (MPa) (3 values due to plane stress condition)
$\varepsilon_{\mathrm{ij}}$, $ij = 11, 22, 33, 12$	Strain response (MPa) (4 values due to plane stress condition)

The distributions of stress and strain values obtained from simulations are presented by box and whisker plots. The line within a box shows the median, the ends of the box

represent the lower and upper quartiles, and the end of the whiskers show the overall data range. Figure 6.4 demonstrates the statistical distribution of the obtained strain values. Abscissae present the strain value ranges, and ordinates show the variation of the number of the finite elements characterised with the values in a specific range. Similarly, figure 6.5 demonstrates the statistical distribution of the obtained stress values. These observations show that data for both stresses and strains are not distributed evenly across the elements of FE meshes. The distributions differ for different variables. The CNN proposed in this research is able to capture predictions for local stress and strain evolution, and it was not required to train separate models for each output variable. The next section presents the background behind the CNN model, employed activation functions, optimisation algorithm, and regularisation methods.

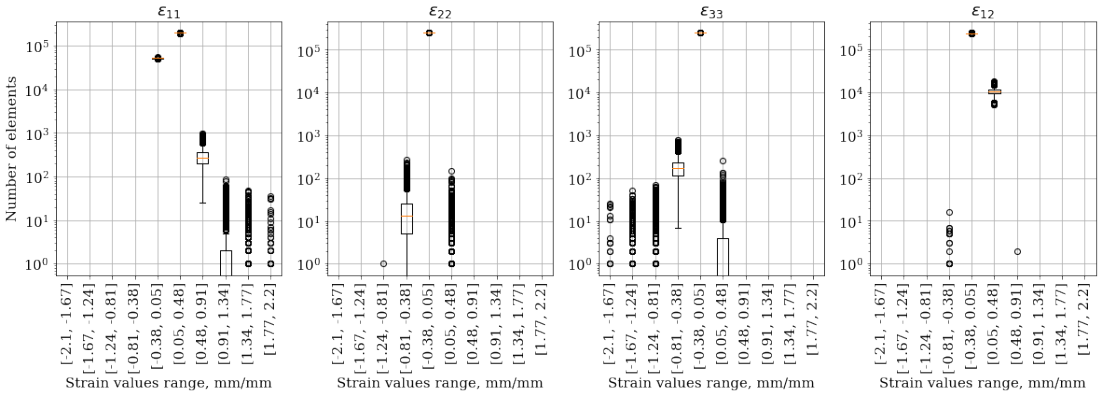

Figure 6.4: Distribution of strain values in the dataset.

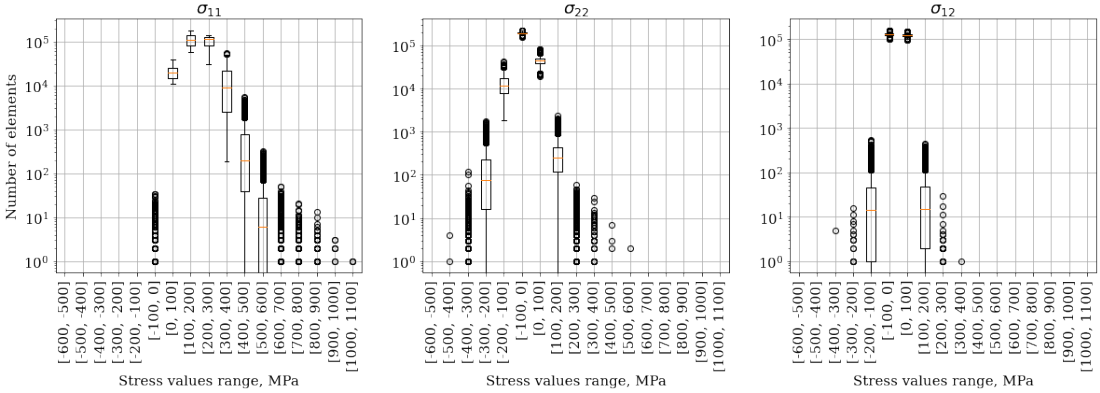

Figure 6.5: Distribution of stress values in the dataset.

6.2.3 Convolutional Neural Networks

Artificial neural networks are the most common ML models. There are many different types of ANNs, such as multilayer perceptrons, convolution neural networks, and others. Previously, the capability of multilayer perceptron models to predict stress-strain evolution for single crystals, as well as the polycrystal average response, was demonstrated [245]. However, multilayer perceptron models would not be suitable for predicting values such as local stress and strain partitionings, as the number of trainable parameters would be too large, especially if the dimensions of a FE mesh are large.

In contrast to multilayer perceptron models, CNN models suitable for two-dimensional input. Typically, they are applied in the field of visual imagery analysis, such as image segmentation [247, 248] and recognition tasks [249, 250]. In the current work, the authors did not use image input to the CNN and instead explored the CNN's capability as a regression model to predict the local stress and strain partitionings based on the input of actual material characteristics.

In this research work, the explored CNNs consisted of: input and output layers, convolutional layers, deconvolutional layers, regularisation layers, and activation layers. The input tensor's dimension is $(100, 100, 12)$. The first two dimensions correspond to the dimensions of the FE mesh, and the third dimension (number of channels the of input layer) corresponds to the number of input variable and includes: a channel fully populated with the macroscopic strain value ε_{11}^{M}, two channels fully populated with the material parameters τ_0 and h_0, and 9 channels populated with corresponding components of rotation matrices. The output layer has in total seven channels of size 100×100, each corresponds to the predictions stress and strain partitionings of the mesh. The predictions are made for the components σ_{11}, σ_{22}, σ_{12}, ε_{11}, ε_{22}, ε_{33}, and ε_{12}. Other stress- and strain-tensor components are equal to zero for all the elements due to plane-stress condition.

Convolutional and deconvolutional layers are formed by implementing a direct convolution and a transposed convolution operations on square neuron layers of CNNs. The operation of direct convolution can be described with the following equation:

$$x_{ij}^{[L]} = \sum_{\xi=0}^{m-1} \sum_{\zeta=0}^{m-1} \theta_{\xi\zeta} x_{(i+\xi)(j+\zeta)}^{[L-1]}, \tag{6.9}$$

where x is the pre-nonlinearity square layer, and L is a positive integer number which denotes the number of the layer ($L \in [0, \ell]$, where $L = 0$ is the index of an input and $L = \ell$ is the index of an output layer), and θ is the filter matrix of the dimension (kernel

size) $m \times m$ and it is populated with trainable parameters. Direct convolution results in downsampling of an input layer. The transposed convolution operation is similar to direct convolution, but serves as a trainable upsampling. In-detail description of convolution operations can be found in [251].

Padding parameter can be applied to control dimensions of output of convolution operations. The term "padding" refers to adding zeros to the borders of filters prior to subjecting them to a convolution operation. Two type of paddings are used in this research: "same" and "valid". "Valid" padding refers to no padding, and "same" results in dimensions of filters of input and output of convolutions being equal. In-detail explanation of the operations in the convolutional networks can be found in [252].

Activation Function & Normalisation

Activation functions are employed to add non-linear qualities to the neural networks [200]. In this study, two activation functions were used: the hyperbolic tangent activation fuction ("tanh") and rectified linear unit ("ReLU") activation function. The tanh activation is expressed as:

$$\tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})} \tag{6.10}$$

The benefits of function is that it a continuous and differentiable zero-centered activation, with its gradient varying in different directions. This function is defined for all real numbers and ranges from -1 to 1. The suitable normalisation for this activation function is a min-max normalisation between -1, 1. This normalisation is achieved by the following transformation:

$$x^{(i)}_{normed} = 2\frac{x^{(i)} - \min x^{(i)}}{\max x^{(i)} - \min x^{(i)}} - 1, \tag{6.11}$$

where $\min x^{(i)}$ is a minimum and $\max x^{(i)}$ is a maximum value of the i^{th} feature.

The ReLU activation function is piece-wise linear function, and is expressed as:

$$f(x) = x^+ = \max(0, x) \tag{6.12}$$

The additional data normalisation was not required for this activation since it was used after the batch normalisation, which serves to re-centre the weights in the network. The employed normalisation methods are shown in table 6.3.

Table 6.3: Features and their normalisation methods.

Variable	**Normalisation method**
ε_{11}^{M}	Normalisation in $-1 \dots 1$
τ_0	Normalisation in $-1 \dots 1$
h_0	Normalisation in $-1 \dots 1$
a_{ij}, $i, j = 1 \dots 3$	No normalisation
σ_{ij}, $ij = 11, 22, 12$	Normalisation in $-1 \dots 1$
$\varepsilon_{\mathrm{ij}}$, $ij = 11, 22, 33, 12$	Normalisation in $-1 \dots 1$

Evaluation of Artificial Neural Networks

The cost function, $J(\theta, b)$, and loss function, $\mathcal{L}\left(y_{pred}^{(i)}, y_{true}^{(i)}\right)$, are used to evaluate the accuracy of the trained CNN. The cost function is used to compute the accuracy of the CNN. An optimisation algorithm minimises a cost function during training by correcting the weights and biases of a network [45]. The loss function is used to calculate the accuracy of a trained model on one sample. A cost function is calculated by averaging the sum of losses for m samples in a dataset:

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}\left(y_{pred}^{(i)}, y_{true}^{(i)}\right), \tag{6.13}$$

Mean squared error (MSE) is used as a loss function in this research. It is expressed as:

$$\mathcal{L}_{\mathrm{MSE}}\left(y_{pred}^{(i)}, y_{true}^{(i)}\right) = \frac{1}{n} \sum_{i=1}^{n} \left(y_{true}^{(i)} - y_{pred}^{(i)}\right)^2, \tag{6.14}$$

where n is a number of elements in an output vector.

Optimisation Algorithm

Adaptive Moment Estimation (Adam) [55] is used an an optimisation algorithm in this research. It is a dradient-descent backpropagation algorithm with an adaptive learning rate. This algorithm preserves an exponentially decaying average of past gradients v_t and updates an then calculates a new value for decaying average of past gradients:

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1)\, g_t, \\ v_t &= \beta_2 v_{t-1} + (1-\beta_2)\, g_t^2, \end{aligned} \tag{6.15}$$

where m_t is the first moments of gradients, and v_t is the second moment of gradients. Then, to avoid these gradients being biased towards zero, their values are corrected according to:

$$\begin{aligned} \hat{m}_t &= \frac{m_t}{1-\beta_1^t}, \\ \hat{v}_t &= \frac{v_t}{1-\beta_2^t}. \end{aligned} \tag{6.16}$$

The network weights are then calculated as:

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t, \tag{6.17}$$

where β_1, β_2, α, and ϵ are the algorithm's hyperparameters that can be tuned during training.

Regularisation

One of the major objectives of training an accurate neural network is to avoid overfitting. In deep learning, the models can be very complex, and a large number of trainable parameters, while having increased capability to learn complex relations, also have the risk of overfitting on the training data. Introducing regularisation into a machine learning model can help accelerate the training process and prevent overfitting. Three different forms of the regularisation parameters were explored in the optimisation process of the neural network architecture: dropout regularisation, L_2 regularisation, and batch normalisation.

In [64], the authors proposed the dropout regularisation method to help machine learning models prevent overfitting. The essence of the dropout method is in introducing a probability parameter that accounts for each weight's probability to be excluded from an update at each training iteration. For each iteration, the re-selection of trainable parameters within the regularised neural network layer is performed according to the selected probability. This regularisation method was shown to significantly improve training and thus is included in the current study.

Another common regularization technique that prevents overfitting in neural networks is L_2 regularisation, which is a type of weight decay technique [63, 62]. This method introduces a penalty on the cost function by adding a weighted squared magnitude of the trainable parameters to the loss function. The cost function can be presented by an equation:

$$J(\theta) = \mathcal{L}_{\text{MSE}}\left(y_{pred}^{(i)}, y_{true}^{(i)}\right) + \lambda \sum_{i=1}^{N_{weights}} \theta_i^2 \tag{6.18}$$

,

where λ is the weight parameter for L_2 regularisation.

Lastly, the efficiency of the batch normalisation method was studied in this research [65]. This method was proposed to help neural networks learn faster and in a more stable fashion. The essence of batch normalisation is normalising the activation layers by standard deviation, i.e., subtracting the mean activation (re-centring) and dividing the re-centred activations by their standard deviation value. An in-depth explanation of the algorithm can be found in [253]. Batch normalisation is a popular regulariser and is often used in deep neural networks; it allows higher learning rates, makes neural networks less sensitive to the choice of trainable parameters' initial values and makes the learning process "smoother" [66]. This method has been shown to perform well in convolutional neural networks applications [254, 255] and was included in the study conducted in this research.

6.3 Convolution Neural Network Model and its Validation

In this section, an optimal architecture search is performed and is presented. A series of different CNN models were first trained on a fraction of the CPFEM data and compared. The best architecture was selected on the basis of those results. Training and test set errors were compared to assess the model's performance. Then, the best model is validated against CPFEM simulations for the completely test set with synthetic microstructures unseen during training. The model was demonstrated to accurately predict stress and strain partitionings of FE mesh. The CNN's prediction capabilities were demonstrated on for the median, worst, and best prediction cases.

6.3.1 Convolutional Neural Networks Architectures

An architecture study was performed to find a suitable network capable of making accurate predictions. This section analysed and reported the effect of the number of layers, number of trainable parameters, regularisation, and choice of activations on the training and validation errors. Ten different CNN models were trained on $5,000$ CPFEM simulations and validated on $1,000$ CPFEM simulations. Using $5,000$ samples of $12,000$ available helped achieve faster training comparison, as training deep CNNs is time-consuming. The CNNs were trained for 15 training epochs with the batch size of 4. The initial learning for the optimisation algorithm was set to 10^{-3}, and optimisation algorithm parameters were to $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$. The best model architecture was selected from all trained models based on the lowest training and validation errors. The first five of the trained CNN varied in the number of convolutional layers and filters of convolutional layers. First, the best performing network was selected out of these five trained networks. Then, this network was tested with different regularisation layers and activation functions. All the networks were compared, and the best performing network was selected for training a final model.

Table 6.4 presents learning results for models $(1)-(5)$. These models are of the encoder-decoder type, and they only differ by the number of constituent layers and their filters. They are composed of four different types of the architectural blocks, which are denoted as $C_1(n)$, $C_2(n)$, $C_1^T(n)$, and $C_2^T(n)$. Architecture block $C_1(n)$ is composed of a direct 2D convolutional layer with the "same" padding parameter and n filters of size 3×3, followed by batch normalisation layer, followed by activation layer with "tanh" activation function. $C_2(n)$ is the architecture block composed of two sequential sub-blocks. Each sub-block comprises the direct 2D convolutional layer with the "valid" padding parameter and n filters of size 3×3, followed by batch normalisation layer, followed by activation layer with "tanh" activation function. Superscript "T" indicates that a transpose 2D convolutional layer is used in the place of a direct convolutional layer. This notation is also explained in table 6.4 in a brief format. The learning curves for training and validation errors of models $(1) - (5)$ are shown in the figure 6.6. The training results have shown that model (1) had the most optimal architecture out of the 5 architectures presented above. This model had $783,271$ trainable parameters, which was neither the smallest number nor the largest number of parameters of all trained models. The comparison of trainable parameters for other models is following: model (2) had $762,471$ trainable parameters, model (3) had $152,103$ trainable parameters, model (4) had $1,795,623$ trainable parameters, and model (5) had $5,358,375$ trainable parameters. Therefore, it is necessary to search for suitable architecture as adding parameters and increasing the complexity of architecture do not

guarantee more accurate predictive capabilities.

Table 6.4: Architectures of the trained models and their corresponding training and validation errors, part 1. The notation used in the architecture representation is described after each architecture description.

Architecture number	**Architecture**	**Training / Validation MSE**
(1)	$C_1(32) - C_2(32) - C_2(32) - C_2(64) -$ $-C_2(128) - C_2^T(128) - C_2^T(64) - C_2^T(32) -$ $-C_2^T(32) - C_1^T(32)$	0.0010/0.0010
(2)	$C_2(32) - C_2(32) - C_2(64) - C_2(128) -$ $-C_2^T(128) - C_2^T(64) - C_2^T(32) - C_2^T(32)$	0.0010/0.0013
(3)	$C_1(32) - C_2(64) - C_2^T(64) - C_1^T(32)$	0.0013/0.0014
(4)	$C_1(32) - C_2(64) - C_2(256) -$ $-C_2^T(64) - C_2^T(256) - C_1^T(32)$	0.0010/0.0025
(5)	$C_1(32) - C_2(32) - C_2(32) - C_2(64) -$ $-C_2(128) - C_2(256) - C_2(256) - C_2^T(256) -$ $-C_2^T(256) - C_2^T(128) - C_2^T(64) - C_2^T(32) -$ $-C_2^T(32) - C_1^T(32)$	0.0012/0.0013
	Notation: $C_1(n) = \text{Conv2D}(n, \text{“same”}) \to \text{BN} \to \tanh$, $C_1^T(n) = \text{Conv2DT}(n, \text{“same”}) \to \text{BN} \to \tanh$, $C_2(n) = (\text{Conv2D}(n, \text{“valid”}) \to \text{BN} \to \tanh)^2$, $C_2^T(n) = (\text{Conv2DT}(n, \text{“valid”}) \to \text{BN} \to \tanh)^2$, where n is the number of 3×3 filters, and BN denotes batch normalisation layer.	

The second part of the architecture search studied how different activation functions and regularisation parameters affect the training results of the network. Table 6.5 presents the training and validation errors for the networks $(6) - (10)$ and provides descriptions for architectures. This study investigated how three different regularisation methods and two activation functions affect learning capabilities. Model (1), as the most accurate of models $(1) - (5)$, was selected as the basis for architecture. Model (6) was trained without batch normalisation, and the error was almost seven times larger than the error of model (1), which indicates that batch normalisation significantly improves the training process. Then, the batch normalisation regularisation method was compared with the L_2 regularisation in

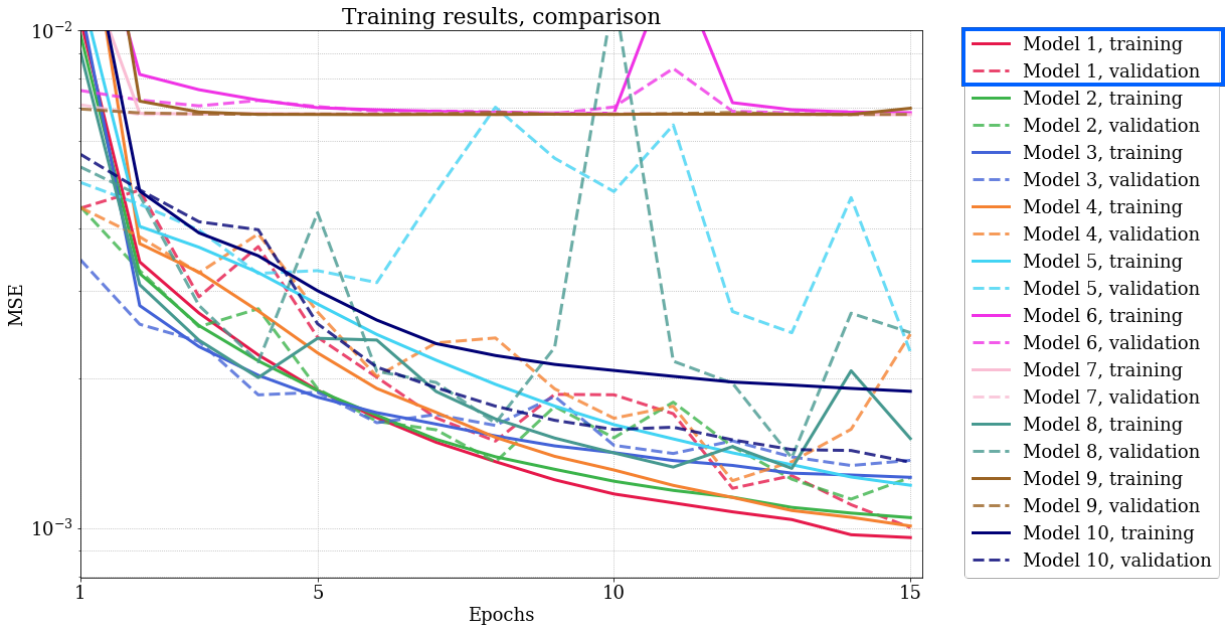

Figure 6.6: Neural networks learning curves showing the evolution of losses for training and validation datasets. Each network is numerated, and each number corresponds to unique network architecture. For each ANN, the architecture and validation error are summarised in tables 6.4 and 6.5

the model (7). The L_2 regularisation factor λ was taken equal to its standard value of 0.01. The error was comparable to one obtained for model (6) (no regularisation), which shows that L_2 does not contribute to the error decrease. The training of the model (8) realised the comparison of "tanh" and "ReLU" activation functions. In this model, batch normalisation was kept as in model (1), but all the "tanh" activation functions were replaced with "ReLU" activations. The error decay was observed, but with a lower rate when compared with the "tanh" activation function, and the resultant training error was 1.5 times higher than in the model (1). Models (9) and (10) introduced dropout regularisation without and with batch normalisation correspondingly. Using dropout without batch normalisation did not contribute to the training process, and using dropout and batch normalisation slowed down the learning process. Therefore, model (1) architecture was chosen for training on the whole dataset. Figure 6.6 presents the evolution of training and validation errors during training.

To conclude, ten different models were trained to find the optimal architecture. Training models (1)-(10) took approximately 250 minutes per model, while trained on 5000 samples. Admittedly, there were limitations to performing an extensive architecture search due to

Table 6.5: Architectures of the trained models and their corresponding training and validation errors, part 2. The notation used in the architecture representation is described in the bottom of the table.

Architecture number	**Architecture**	**Training / Validation MSE**
(6)	$C_1(32) - C_2(32) - C_2(32) - C_2(64) - C_2(128) -$ $-C_2^T(128) - C_2^T(64) - C_2^T(32) - C_2^T(32) - C_1^T(32)$ Notation: $C_1(n) = \text{Conv2D}(n, \text{“same”}) \rightarrow \tanh$, $C_1^T(n) = \text{Conv2DT}(n, \text{“same”}) \rightarrow \tanh$, $C_2(n) = (\text{Conv2D}(n, \text{“valid”}) \rightarrow \tanh)^2$, $C_2^T(n) = (\text{Conv2DT}(n, \text{“valid”}) \rightarrow \tanh)^2$.	0.0068/0.0068
(7)	$C_1(32) - C_2(32) - C_2(32) - C_2(64) - C_2(128) -$ $-C_2^T(128) - C_2^T(64) - C_2^T(32) - C_2^T(32) - C_1^T(32)$ Notation: $C_1(n) = \text{Conv2D}(n, \text{“same”}) \rightarrow L_2(0.01) \rightarrow \tanh$, $C_1^T(n) = \text{Conv2DT}(n, \text{“same”}) \rightarrow L_2(0.01) \rightarrow \tanh$, $C_2(n) = (\text{Conv2D}(n, \text{“valid”}) \rightarrow L_2(0.01) \rightarrow \tanh)^2$, $C_2^T(n) = (\text{Conv2DT}(n, \text{“valid”}) \rightarrow L_2(0.01) \rightarrow \tanh)^2$.	0.0068/0.0068
(8)	$C_1(32) - C_2(32) - C_2(32) - C_2(64) - C_2(128) -$ $-C_2^T(128) - C_2^T(64) - C_2^T(32) - C_2^T(32) - C_1^T(32)$ Notation: $C_1(n) = \text{Conv2D}(n, \text{“same”}) \rightarrow \text{BN} \rightarrow \text{ReLU}$, $C_1^T(n) = \text{Conv2DT}(n, \text{“same”}) \rightarrow \text{BN} \rightarrow \text{ReLU}$, $C_2(n) = (\text{Conv2D}(n, \text{“valid”}) \rightarrow \text{BN} \rightarrow \text{ReLU})^2$, $C_2^T(n) = (\text{Conv2DT}(n, \text{“valid”}) \rightarrow \text{BN} \rightarrow \text{ReLU})^2$.	0.0015/0.0025
(9)	$C_1(32) - C_2(32) - C_2(32) - C_2(64) - C_2(128) -$ $-C_2^T(128) - C_2^T(64) - C_2^T(32) - C_2^T(32) - C_1^T(32)$ Notation: $C_1(n) = \text{Conv2D}(n, \text{“same”}) \rightarrow \tanh \rightarrow \text{Dropout}(0.2)$, $C_1^T(n) = \text{Conv2DT}(n, \text{“same”}) \rightarrow \tanh \rightarrow \text{Dropout}(0.2)$, $C_2(n) = (\text{Conv2D}(n, \text{“valid”}) \rightarrow \tanh \rightarrow \text{Dropout}(0.2))^2$, $C_2^T(n) = (\text{Conv2DT}(n, \text{“valid”}) \rightarrow \tanh \rightarrow \text{Dropout}(0.2))^2$.	0.0070/0.0068
(10)	$C_1(32) - C_2(32) - C_2(32) - C_2(64) - C_2(128) -$ $-C_2^T(128) - C_2^T(64) - C_2^T(32) - C_2^T(32) - C_1^T(32)$ Notation: $C_1(n) = \text{Conv2D}(n, \text{“same”}) \rightarrow \text{BN} \rightarrow \tanh \rightarrow \text{Dropout}(0.2)$, $C_1^T(n) = \text{Conv2DT}(n, \text{“same”}) \rightarrow \text{BN} \rightarrow \tanh \rightarrow \text{Dropout}(0.2)$, $C_2(n) = (\text{Conv2D}(n, \text{“valid”}) \rightarrow \text{BN} \rightarrow \tanh \rightarrow \text{Dropout}(0.2))^2$, $C_2^T(n) = (\text{Conv2DT}(n, \text{“valid”}) \rightarrow \text{BN} \rightarrow \tanh \rightarrow \text{Dropout}(0.2))^2$.	0.0019/0.0014

the large number of hyperparameters that can be changed and compared during training. The other limitations are the cost of collecting the data and training the models. It would be reasonable to expect more complex and deep models trained on more data to give more accurate predictions. However, the data collection and training will require more time and computational resources. The selected model has been shown to achieve excellent stress and strain evolution predictions when trained on more data, as shown in the following subsection.

6.3.2 Best Model Training and its Evaluation on the Test Set

The best CNN selected in the previous subsection was used for re-training using a larger dataset, which was split into three parts: training set, validation set, and test set. CPFEM simulations were performed to obtain corresponding labels for the inputs sampled using the Sobol sequence method. This method allows the filling of the samples' space gradually and uniformly, therefore, additional shuffling of the collected dataset is not required. The dataset consisted of $12,000$ samples. The last $1,000$ of all samples in the dataset were used for the test set. The rest of the data was randomly split into training and validation datasets in proportion of 10:1. The number of training samples was selected using the same approach presented in 5.2.4 section: more data was introduced for training until a target validation error was achieved. The validation set was used for evaluation of the CNN accuracy after each training epoch, but the information from this set was not explicitly used for training. The test set was completely unseen during the training. The network was trained for 100 epochs, and the training process took approximately 37 hours. Figure 6.7 shows the schematics of the architecture of the trained convolutional neural network. The black block represents the input to the CNN and shows its dimensions. Then, the input is fed to the series of encoding direct convolutional layers and then to the series of decoding transpose convolutional layers, after which the output is obtained.

The evolution of the training and validation errors is depicted in figure 6.8. The loss curves show that the training and validation errors stopped decreasing significantly at approximately the 60^{th} epoch. At the end of the training, the training error was equal to 6.0×10^{-4}, and the validation error was equal to 5.8×10^{-4}. These errors were calculated for the normalised data and averaged across seven predicted output features, making it nearly impossible to access the network's performance for a human. To understand the predictive performance of the network, the errors for training and test sets were converted to their original scale for each individual predicted variable and assessed using the root mean square error (RMSE):

$$RMSE = \sqrt{\sum_{i=1}^{N} \frac{\left(y_i^{CP} - y_i^{CNN}\right)^2}{N}}, \tag{6.19}$$

where y_i^{CP} is a CPFEM prediction for a given variable, and y_i^{CNN} its corresponding prediction made by the neural network, and $N = 10,000$ is the number of elements in each FE mesh. Two types of RMSE errors were calculated: overall RMSE averaged across all timesteps and RMSE for the final timestep of deformation.

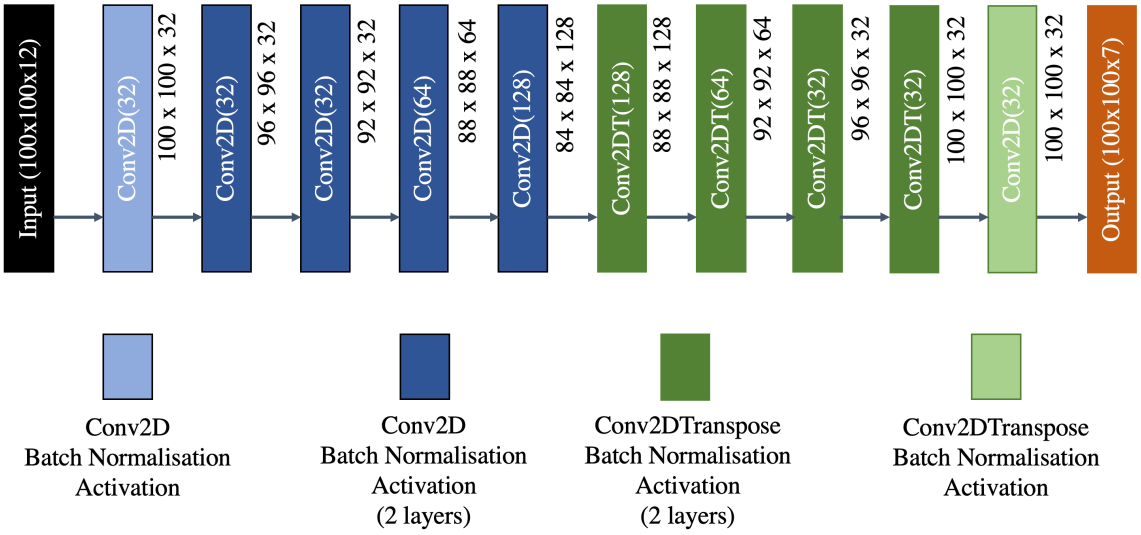

Figure 6.7: The schematics of the CNN architecture. The black block shows the input to the model, and the orange block shows the model's output. The intermediate blocks of the CNN are the convolutional blocks of the CNN with implemented batch normalisation and activation layers. The dimensions next to these blocks are the resulting output dimensions.

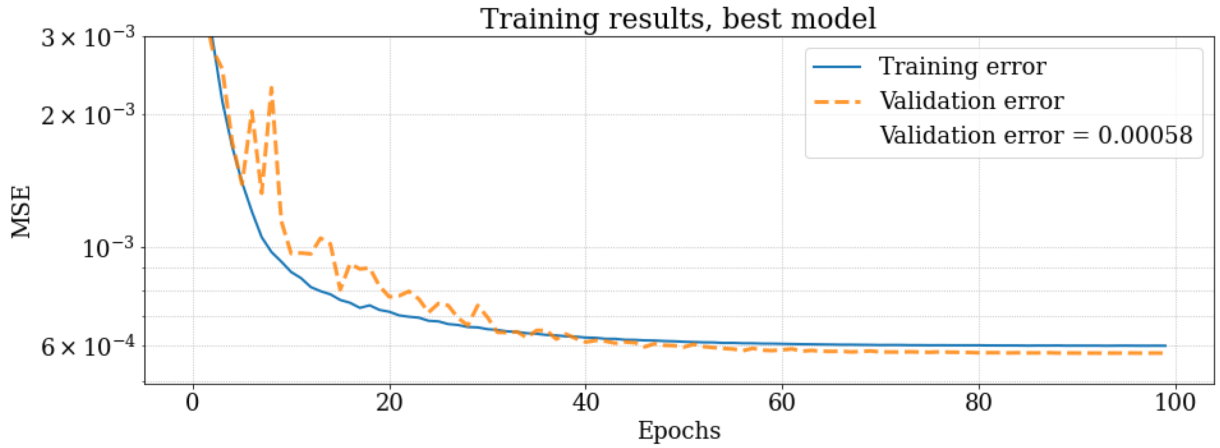

Figure 6.8: The evolution of the training and validation mean squared errors for the selected CNN model.

In order to thoroughly compare the predictions of the CNN and the CPFEM models, the equivalent von Mises stress and strain values were calculated and compared for CNN and CPFEM predictions. The errors for equivalent von Mises stress and strain were obtained by taking the difference between the equivalent von Mises values of the two models. The von Mises strain error can be expressed as $\varepsilon^{VM}_{error} = \varepsilon^{VM}_{CPFEM} - \varepsilon^{VM}_{CNN}$, where ε^{VM}_{CPFEM} is the von Mises strain obtained for CPFEM predictions, and ε^{VM}_{CNN} is the von Mises strain for CNN predictions. The von Mises stress error is calculated similarly. The equivalent stress and strains were assessed for the training and test set sample. Figure 6.9 displays that the equivalent stress error values are primarily situated in the range $[-10, 10]$ MPa, with the median error being close to zero MPa. The error values for the equivalent stress confirm that the model is accurate for the overall stress predictions. The equivalent strain errors mainly were situated within the $[0, 0.2]$ range of strain, with median values close to zero (under 0.01). These results show that the trained neural network model accurately predicted tensorial strain components.

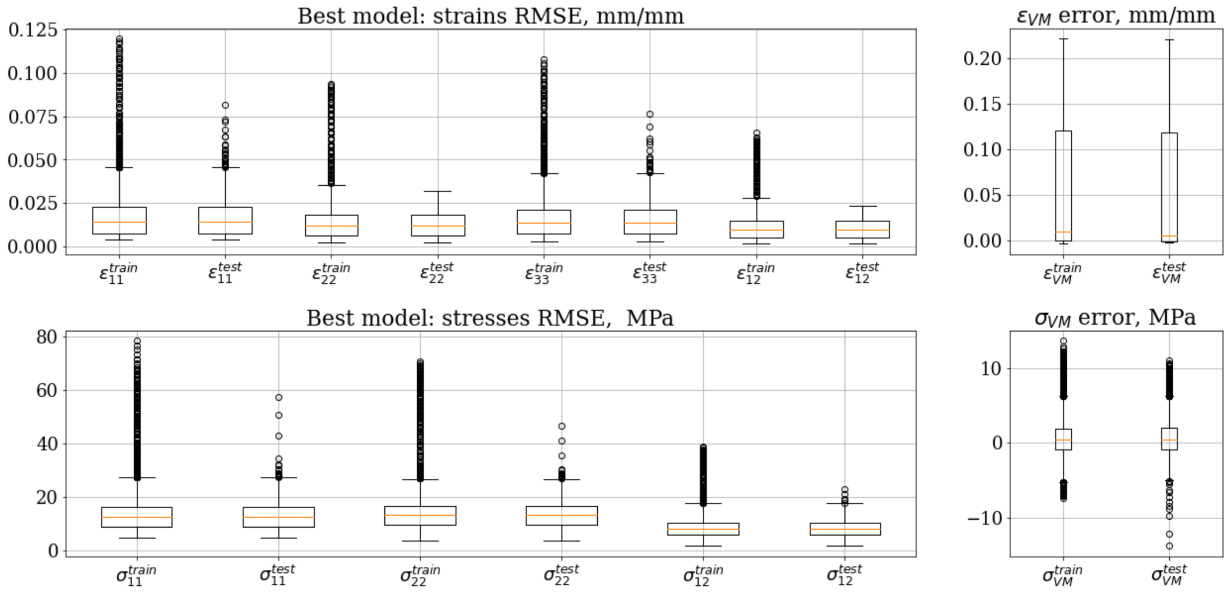

Figure 6.9: Best trained model results: root mean squared errors for all variables predictions (left side of the figure) and von Mises difference error for equivalent stress and strain (right side of the figure).

Further, the median, highest (the "worst" case), and lowest (the "best" case) error cases were visualised for the test set predictions. Figure 6.10 presents the CPFEM simulation and CNN prediction results comparison for the median case. The input microstructure is

shown on the top left of the figure and is coloured using inverse pole figure (IPF) colouring. The bottom left of the figure presents the stress-strain response averaged across all elements in the FEM mesh. The averaged stress-strain response RMSE is 1.93 MPa for the median case, demonstrating the trained network's excellent predictive capabilities. To the right of the input texture and stress-strain curve, the figure displays the evolution of the ε_{11} strain component partitioning. The top row shows the CPFEM predictions compared to the CNN predictions in the bottom row. Strain partitioning of FEM mesh is compared for 6%, 12%, 17%, and 22% of strain. CP and CNN predictions results look nearly identical when compared to each other, for all deformation steps, except for minor deviations. The RMSE errors for all predicted variables are shown in the tables 6.6 and 6.7. Table 6.6 reports the RMSE errors calculated over the whole history of the deformation, and table 6.7 calculated for the last increment of the deformation. The RMSE error for σ_{11} stress component was equal to 16.14 MPa when calculated for all timesteps, which is equal to approximately 5% of the maximum averaged stress, and it was equal to 24.32 MPa for the last timestep which is equal to approximately 8% of the maximum averaged stress. The corresponding errors for ε_{11} were equal to 0.02 and 0.03, which confirm the excellent agreement of CPFEM and CNN results. The results for the median error show that the overall performance of the trained network is very accurate.

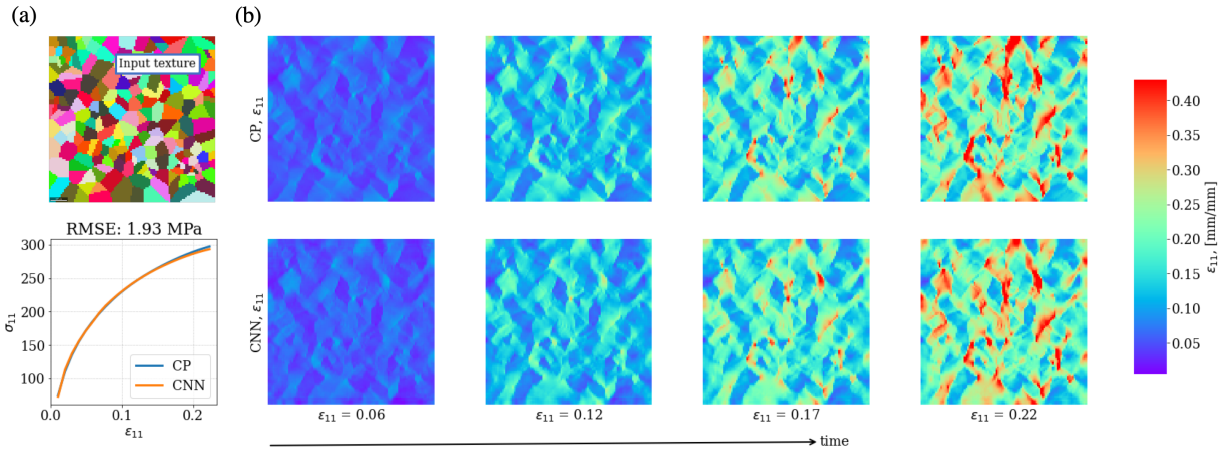

Figure 6.10: Test set prediction with the median error; (a): input microstructure and flow-curve for crystal plasticity and convolutional neural network predictions averaged across all elements of the mesh; (b): ε_{11} strain partitioning evolution for four strain levels.

Secondly, the test set prediction with the highest error was analysed. Similarly to the median case, figure 6.11 shows the input texture, average stress-strain response, and

strain-partitioning evolution for ε_{11} component. The input microstructure deviates from most of the textures in the training set, such that the strain localisation occurs during the deformation. This localisation results in stress unloading, as observed in the average stress-strain response. The average stress-strain response RMSE had a value of 11.55 MPa. This error is higher than the error in the median case, as the predicted macroscopic stress does not capture global softening predicted by crystal plasticity. Even though the network was not able to predict the exact intensity of ε_{11} strains in the localisation region, the CNN was able to identify its location accurately, as identified in figure 6.11. The rest of the strain partitioning is predicted accurately and is in excellent agreement with CPFEM results. For σ_{11}, The RMSE error was equal 22.69 MPa for the values across all timesteps of the deformation (see table 6.6), and the RMSE for the last timestep was equal to 57.29 MPa (see table 6.7). The RMSEs for other variables are also reported in the tables 6.6 and 6.7. Even though this test example is a test set outlier in terms of stress-strain behaviour, the CNN predictions were very accurate up to the point of localisation occurrence. Due to such outliers in the training and test sets, several higher errors can be observed in the figure 6.9 in training and test set errors for all variables. All the test set outliers (can be observed in figure 6.9) were the cases similar to the "worst" case. The microstructures of the outlier samples cause the localisation to occur, which was underpredicted by the CNN.

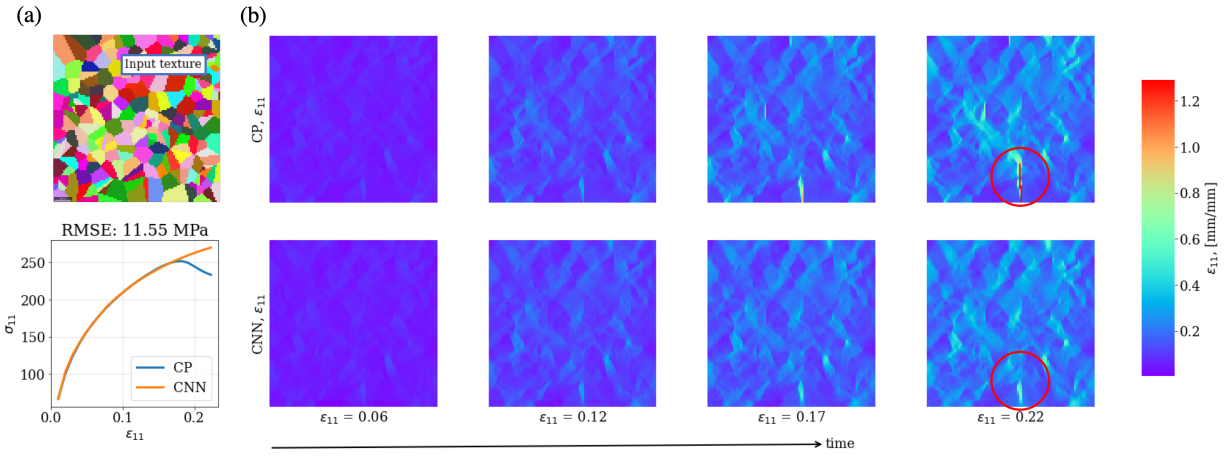

Figure 6.11: Test set prediction with the highest error; (a): input microstructure and flow-curve for crystal plasticity and convolutional neural network predictions averaged across all elements of the mesh; (b): ε_{11} strain partitioning evolution for four strain levels.

Finally, the test case prediction with the lowest error (i.e. most accurate prediction)

was visually assessed and the RMSE for all the variables were evaluated for this case. Similarly to previous cases, figure 6.12 displays the initial microstructure in IPF colouring scheme. Average stress predictions obtained with CNN show exceptional agreement with the averaged crystal plasticity response. The RMSE error for the averaged stress response over the whole deformation was equal to 1.35 MPa. The strain partitioning evolution predicted by CNN are in an excellent agreement with CPFEM simulation results. All the strain hot-spots are predicted correctly by the neural network model with exception of slight intensity underprediction that can be observed in the final timestep of the deformation. The RMSE errors for σ_{11} stress component across all timesteps was equal to 15.37 MPa (see table 6.6) which is comparable to the median case. For the last timestep the σ_{11} RMSE between CNN and CPFEM predictions was equal to 23.58 MPa (see table 6.7) which is also comparable to the median case error. The errors for ε_{11} are identical for best and median cases (tables 6.6 and 6.7) confirm excellent predictive capabilities of the CNN model.

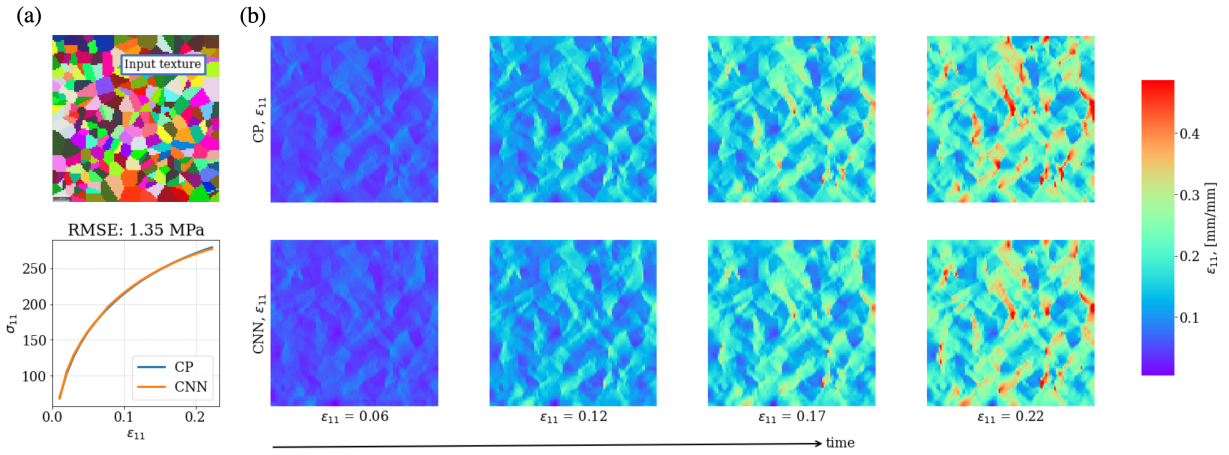

Figure 6.12: Test set prediction with the lowest error; (a): input microstructure and flow-curve for crystal plasticity and convolutional neural network predictions averaged across all elements of the mesh; (b): ε_{11} strain partitioning evolution for four strain levels.

To conclude, the results for the test set has demonstrated a good accuracy of the trained neural network. The test set's microstructures were synthetically generated using the same methodology as for training set microstructures. The test set was entirely new for the CNN model, and the neural network demonstrated excellent accuracy. The following section validates flexibility of the trained CNN by using actual material microstructures.

Table 6.6: Element-wise RMSE errors calculated across all timesteps for all predicted variables.

Case	σ_{11}, **MPa**	σ_{22}, **MPa**	σ_{12}, **MPa**	ε_{11}	ε_{22}	ε_{33}	ε_{12}
Median	16.14	16.14	10.23	0.02	0.02	0.02	0.01
Worst	22.69	20.39	11.42	0.04	0.03	0.03	0.01
Best	15.37	15.61	10.28	0.02	0.01	0.02	0.01

Table 6.7: Element-wise RMSE errors calculated for the last timesteps for all predicted variables.

Case	σ_{11}, **MPa**	σ_{22}, **MPa**	σ_{12}, **MPa**	ε_{11}	ε_{22}	ε_{33}	ε_{12}
Median	24.32	22.33	15.05	0.03	0.03	0.03	0.02
Worst	57.29	46.46	22.85	0.08	0.03	0.08	0.02
Best	23.58	21.71	14.82	0.03	0.03	0.03	0.02

6.4 Application for Real Materials

This section validates the CNN against CPFEM prediction for two completely new actual material microstructures: AA5754 and AA6061 aluminium alloys. Then, the extension of the model to strain localisation and stress softening response prediction was presented. The limitations and model results were presented and discussed. Finally, the runtime comparison of CNN and CPFEM models was performed.

6.4.1 Application to AA5754 and AA6061

This subsection presents the CPFEM and CNN predictions comparison for microstructures of AA5754 and AA6061 alloys. These results show the full predictive capability of the proposed framework and prove that training the CNN on the synthesised microstructure data can help achieve accurate machine learning predictions for the microstructures of real materials. For each material, the input microstructure is presented and the predictions are visualised for ε_{11} strain and σ_{11} stress components. The strain and stress partitionings evolution were compared for CPFEM simulation result and CNN prediction for 6%, 12%,

17%, and 22% macroscopic strain, ε_{11}^{M}, values. Stress and strain responses averaged across all grains are presented. Two error metrics are used to estimate accuracy of the predictions. The first error metric is an element-wise difference between CPFEM and CNN predictions and it was visualised using the histogram method. This difference error (DE) is calculated using the equation:

$$\text{DE} = y^{CP} - y^{CNN}, \tag{6.20}$$

where y^{CP} is the CPFEM prediction and y^{CNN} is the prediction by the neural network. Then, an error map is employed to visualise the error between CPFEM and CNN predictions. The error map displays the absolute value of difference error in the form of a heat map for ε_{11} strain and σ_{11} stress values. The white colour in error maps represents no error, and the black represents the maximum error of a given prediction. The absolute difference error (ADE) was calculated using the following equation:

$$\text{ADE} = \left| y^{CP} - y^{CNN} \right|, \tag{6.21}$$

where y^{CP} is the CPFEM prediction and y^{CNN} is the prediction by the CNN. Finally, the RMSE error metric was employed to calculate the error across all timesteps and for the last timestep. The RMSE were calculated for all tensorial components of strain and stress and are presented in the tables 6.8 and 6.9. The CPFEM and CNN predictions for σ_{22}, σ_{12}, ε_{22}, ε_{33}, and ε_{12} were compared visually for the last timestep, and the RMSE errors for all timesteps and for the last timesteps are reported in the tables 6.8 and 6.9.

First, the AA5754 aluminium alloy microstructure was used to test the flexibility of predictive capabilities of the CNN. Figure 6.13 shows: (a) the initial microstructure of the material, comparison of the averaged ε_{11} strain, histogram of element-wise difference errors at four different deformation levels, and (b) comparison of the strain evolution predicted by CPFEM and CNN for the same different deformation levels and the error map. It is important to notice that the microstructure of AA5754 aluminium alloy is very different from the texture generated for the dataset. The typical examples of the texture in the dataset were presented in figures 6.10, 6.11, 6.12. Despite that the morphology of the grains in the AA5754 microstructure is different from those found in training set, the predictions of the CNN are in excellent agreement with crystal plasticity simulations. The CNN successfully predicted the strain hot spots, except for minor underpredictions of the strain intensity in the late stage of the deformation. The error map in the third row of the figure 6.13, part (b), displays that the prediction errors are overall close to zero. The error histogram also demonstrates that most errors are concentrated around zero and are

mostly less than 0.1 in their value for all deformation steps. Overall RMSE between CNN and CPFEM predictions for ε_{11} is equal to 0.02 (reported in table 6.8) and the RMSE for the last time step is 0.03 (reported in table 6.9) and is less than 5% of the maximum strain value in the mesh. The results for strains are in excellent agreement with CPFEM, which confirm the accuracy of the neural network model and its flexibility to predict deformation for various heterogeneous microstructures.

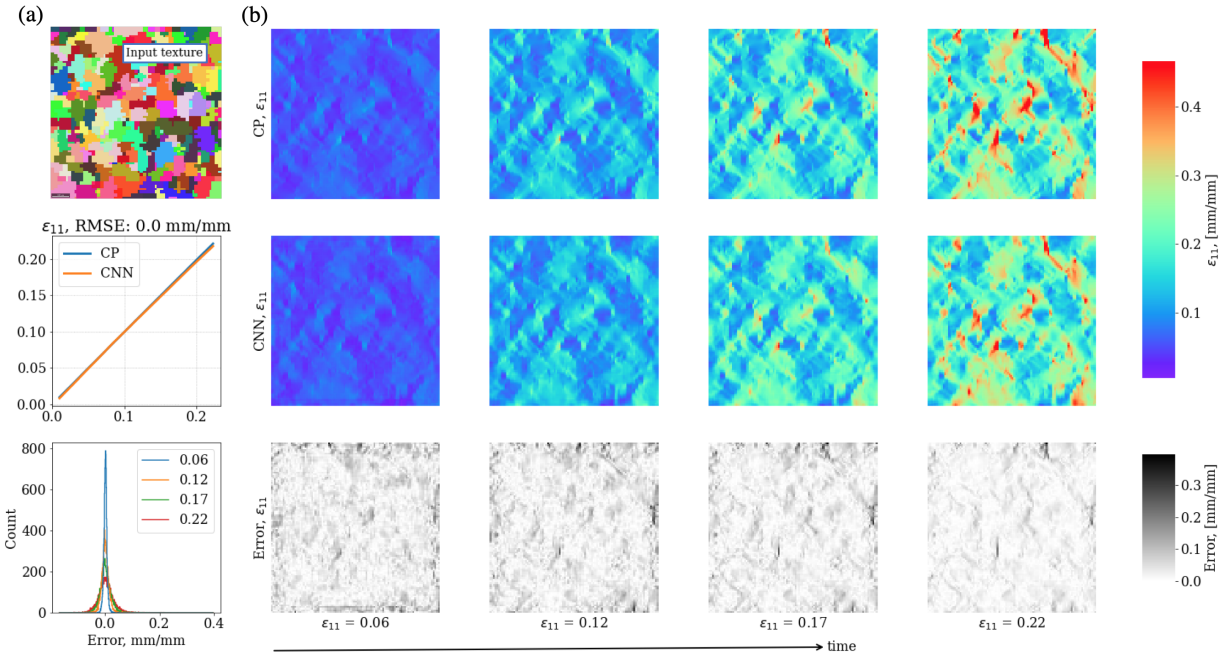

Figure 6.13: Predictions for an AA5754 microstructure; (a): input microstructure, ε_{11} predictions for crystal plasticity and convolutional neural network predictions averaged across all elements of the mesh, and histogram showing the difference error distribution; (b): ε_{11} strain partitioning evolution for four strain levels. The first row displays crystal plasticity predictions, the second row displays convolutional neural network predictions, and the third row shows the error map, which displays the absolute element-wise difference error between the crystal plasticity and convolutional neural network predictions.

Figure 6.14 displays the evolution of σ_{11} stress during deformation process for the same initial microstructure. The second row of the figure's part (a) shows the average stress-strain curve. It was obtained by averaging the stress response across all elements in the mesh for each timestep. The RMSE error for average stress was equal to 3.18 MPa, which confirms the accuracy of overall predictions. For the element-wise comparison of

predictions, figure's first row of the part (b) shows the CPFEM predictions; then, the second row shows the CNN predictions, and the third row shows the element-wise absolute difference error for CPFEM and CNN predictions. The stress hot-spots are predicted for each timestep with excellent accuracy. The ADE heatmap displays that most of the errors are close to zeros, which confirms the accuracy of the CNN model. Histogram plot for the predictions (figure 6.14, (a), third row) also confirm that the CNN predictions are in excellent agreement with CPFEM simulations. Most of the difference errors are close to zero, and the great majority of the errors are less than 100 MPa in their absolute value. Overall RMSE between CNN and CPFEM predictions for σ_{11} is equal to 15.96 MPa (reported in table 6.8) and the RMSE for the last time step is 24.75 MPa (reported in table 6.9), which is less than 8% of the maximum stress value in the mesh. The results for strains are in excellent agreement with CPFEM, which confirm both accuracy and flexibility of the proposed CNN model.

To fully verify the predictive capabilities of the convolutional neural network, another actual material microstructure was used for prediction. The microstructure under consideration is of the AA6061 aluminium alloy [80]. The pole figures of AA5754 and AA6061 microstructures are compared in figure 6.16, demonstrating the significant difference between textures presented in these two materials. The other substantial difference between the materials' microstructures is the grain morphology. The input microstructure of AA6061 can be observed in the first row figure 6.17, (a). The grains in the AA6061 exhibit overall shapes and sizes that are different from grains of AA5754 microstructure (figure 6.13, (a), first row). The grains vary in size, unlike in artificially generated grains in the training data, and unlike in AA5754 microstructure, where grains are similar in their area. Some of the grains in the AA6061 microstructure exhibit different degrees of elongation and grain morphology in this specimen is more heterogeneous than in AA5754. For this microstructure, ε_{11} strain and σ_{11} stress CPFEM and CNN predictions are compared in detail, and all other components are compared visually for the last timestep.

The average strain predictions ε_{11} for CPFEM and CNN are shown on the figure 6.17, (a), second plot. The CNN model displays excellent predictive capabilities, and the error for the averaged predictions is negligible. The third plot in the (a) part of the figure shows the histogram for the difference error between predictions for four strain levels. The errors are majorly situated close to zero and the majorly are less than 0.1 in their absolute value. Strain partitioning comparison is shown on the (b) part of the figure and the error map showing absolute difference error. The red ellipse highlights the only hotspot that the machine learning model did not capture. The rest of the strain hotspots are captured with exceptional accuracy, which is confirmed by the error plot in the third row. The blue arrows highlight some of the successfully predicted strain hotspots within the partitioning.

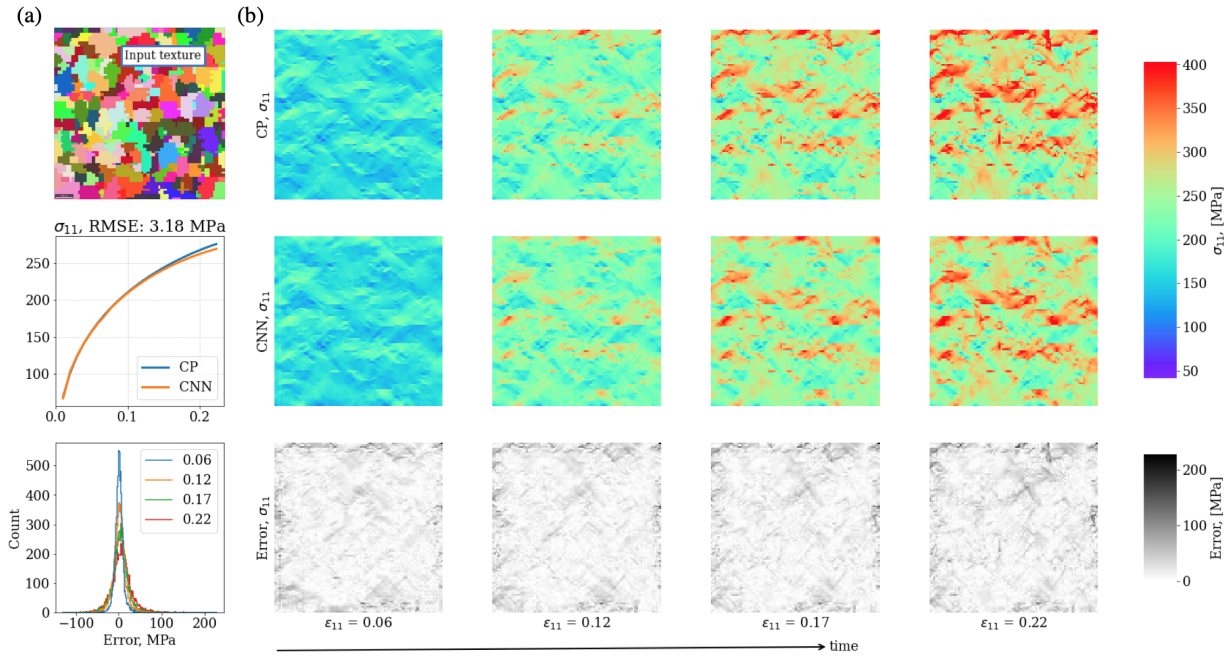

Figure 6.14: Predictions for an AA5754 microstructure; (a): input microstructure, σ_{11} predictions for crystal plasticity and convolutional neural network predictions averaged across all elements of the mesh, and histogram showing the difference error distribution; (b): σ_{11} stress partitioning evolution for four strain levels. The first row displays crystal plasticity predictions, the second row displays convolutional neural network predictions, and the third row shows the error map, which displays the absolute element-wise difference error between the crystal plasticity and convolutional neural network predictions.

Overall RMSE for ε_{11} strain is equal to 0.02 (reported in table 6.8) and the last timestep error for ε_{11} is equal to 0.03 (reported in table 6.9), which confirm the consistency in the accuracies of the predictions. These results have demonstrated the excellent capability of the trained model to predict ε_{11} strain component mesh partitioning across the whole history of deformation.

The average stress predictions σ_{11} for CPFEM and CNN are shown on the figure 6.18, (a), second plot. CNN displays excellent predictive capabilities, and the RMSE for the averaged flow curve is equal to 6.46 MPa. The third plot in the (a) part of the figure shows the histogram for the difference error between predictions for four strain levels. The errors are majorly situated close to zero in their absolute value. Stress partitioning comparison

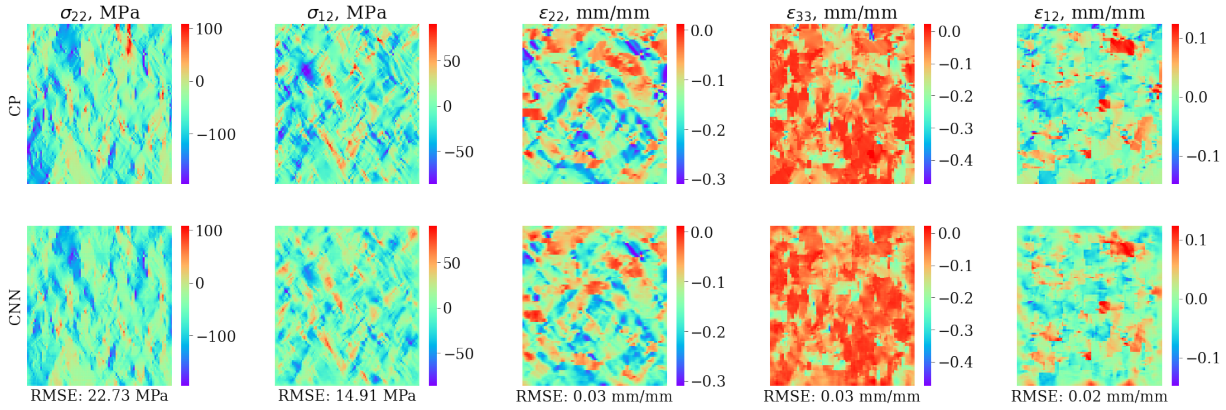

Figure 6.15: Prediction for an AA5754 microstructure; the first row displays the crystal plasticity finite element predictions for the last timestep of the deformation, and the second row the convolutional neural network predictions for the last timestep of the deformation.

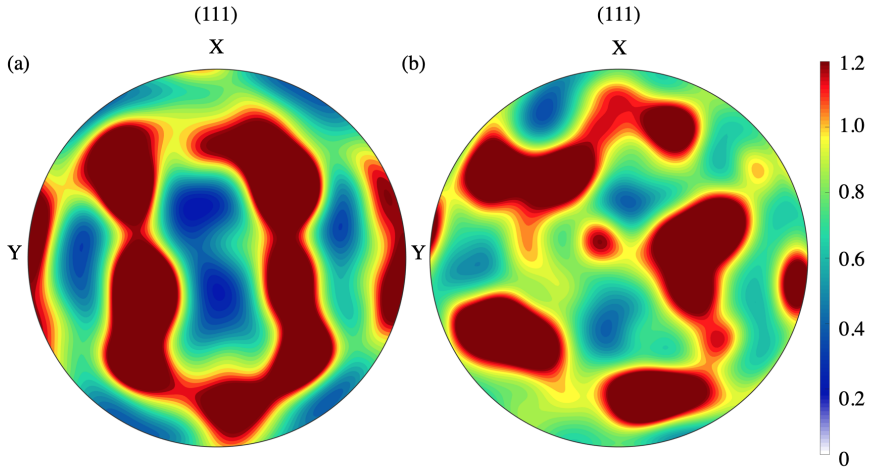

Figure 6.16: (a) Pole figure for AA5754 microstructure and (b) pole figure for AA6061 microstructure

is shown on the (b) part of the figure and the error map showing absolute difference error. The blue ellipses highlight the major stress hotspots that evolved during the deformation, which were captured accurately with the network. This accuracy is also confirmed by the absolute difference error plot in the third row. Overall RMSE for σ_{11} stress is equal to 17.49 MPa (reported in table 6.8) and the last timestep error for σ_{11} is equal to 26.84 MPa

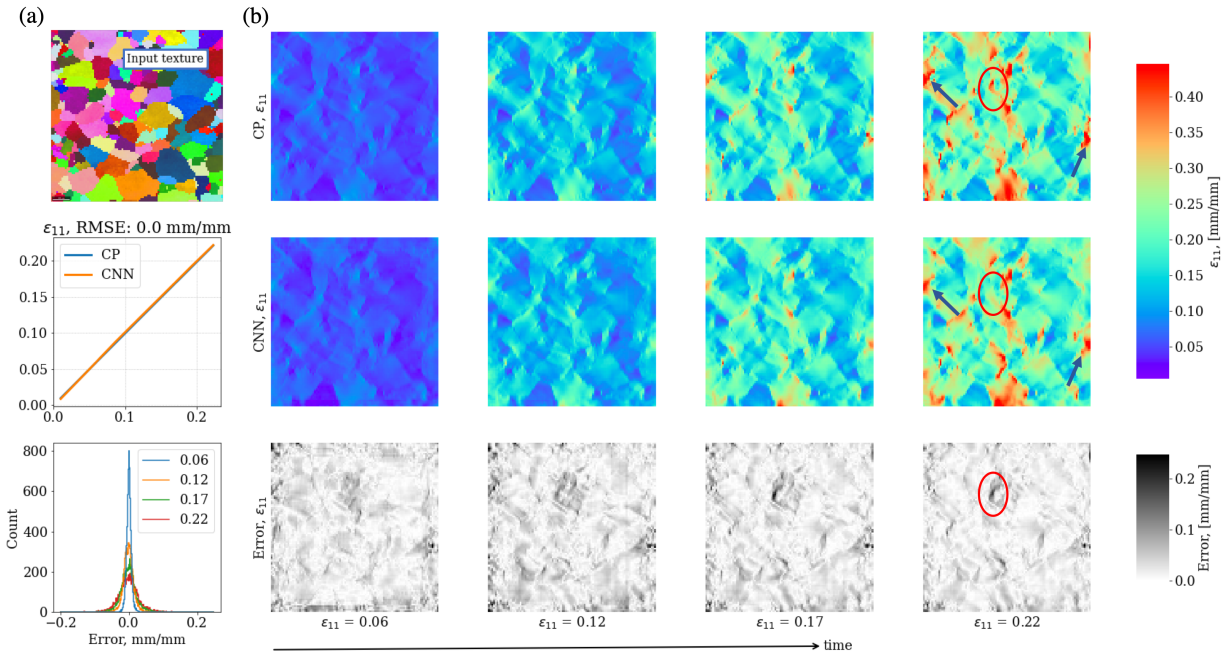

Figure 6.17: Predictions for a AA6061 microstructure; (a): input microstructure, ε_{11} predictions for crystal plasticity and convolutional neural network predictions averaged across all elements of the mesh, and histogram showing the difference error distribution; (b): ε_{11} strain partitioning evolution for four strain levels. The first row displays crystal plasticity predictions, the second row displays convolutional neural network predictions, and the third row shows the error map, which displays the absolute element-wise difference error between the crystal plasticity and convolutional neural network predictions.

(reported in table 6.9), which confirm the consistency in the accuracies of the predictions. These results have demonstrated the excellent capability of the trained model to predict σ_{11} strain component mesh partitioning across the whole history of deformation for AA6061 alloy and validate the framework's capability to predict deformation behaviour for a wide range of materials.

The predictions for the other stress and strain tensor components were evaluated. As previously, only the last time step of the deformation process is visualised (figure 6.19). The overall accuracy of prediction is also clear from the visual representation. The predictions for σ_{22} component has a slight underprediction across the mesh. Overall RMSE for σ_{22} is equal to 18.46 MPa, and the RMSE for the last time step is 25.72 MPa. The prediction is visibly σ_{12} very accurate, and all the stress hotspots are predicted correctly within the

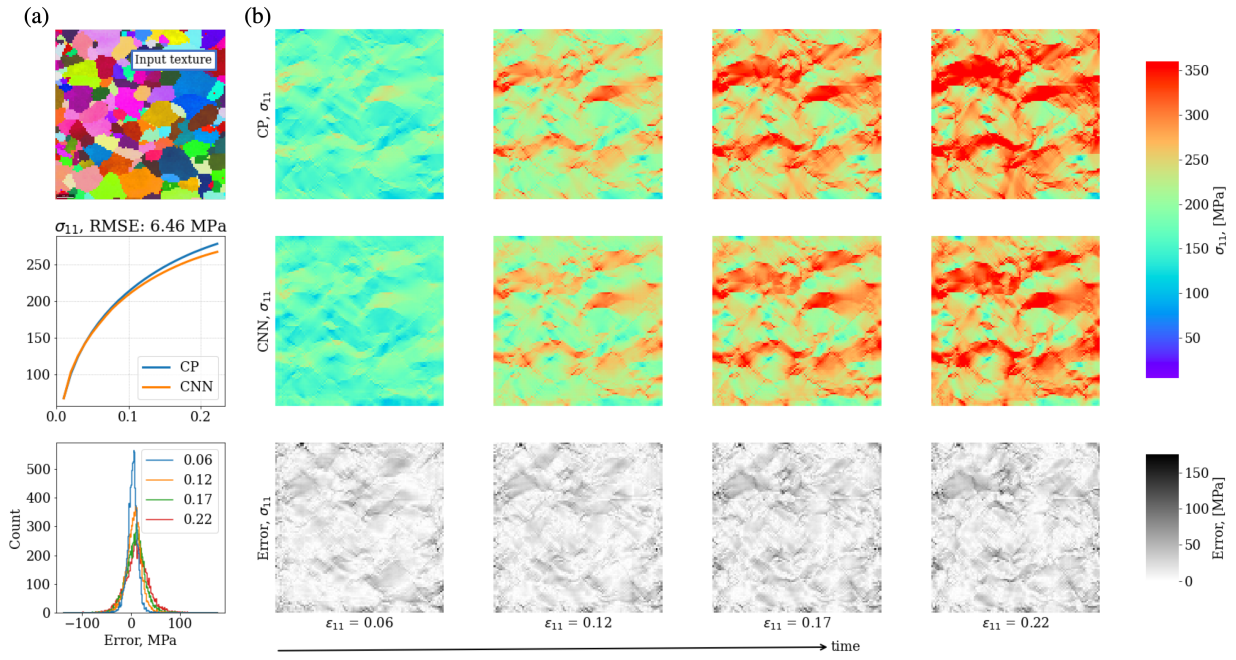

Figure 6.18: Predictions for a AA6061 microstructure; (a): input microstructure, σ_{11} predictions for crystal plasticity and convolutional neural network predictions averaged across all elements of the mesh, and histogram showing the difference error distribution; (b): σ_{11} stress partitioning evolution for four strain levels. The first row displays crystal plasticity predictions, the second row displays convolutional neural network predictions, and the third row shows the error map, which displays the absolute element-wise difference error between the crystal plasticity and convolutional neural network predictions.

partitioning. Overall RMSE for σ_{12} is equal to 9.86 MPa, and the RMSE for the last time step is 14.41 MPa. The rest of the strain partitionings, the predictions are performed exceptionally accurately. The overall RMSE for ε_{22} and ε_{33} is equal to 0.02, and overall RMSE for ε_{12} is as small as 0.01. For the last timestep, the RMSE for ε_{22} and ε_{33} is equal to 0.03, and the last timestep RMSE for ε_{12} is equal to 0.02. The overall RMSE for each of the variables is reported in the table 6.8 and the last timestep RMSE for each variable is reported in the table 6.9. The visualisation of the results and the error values confirm the capability of the convolutional neural network to achieve accurate crystal plasticity predictions for real materials.

To conclude, the CNN was validated for AA5754 and AA6061 microstructures. Despite the fact that the network was never trained for the grain morphology in AA5754, the

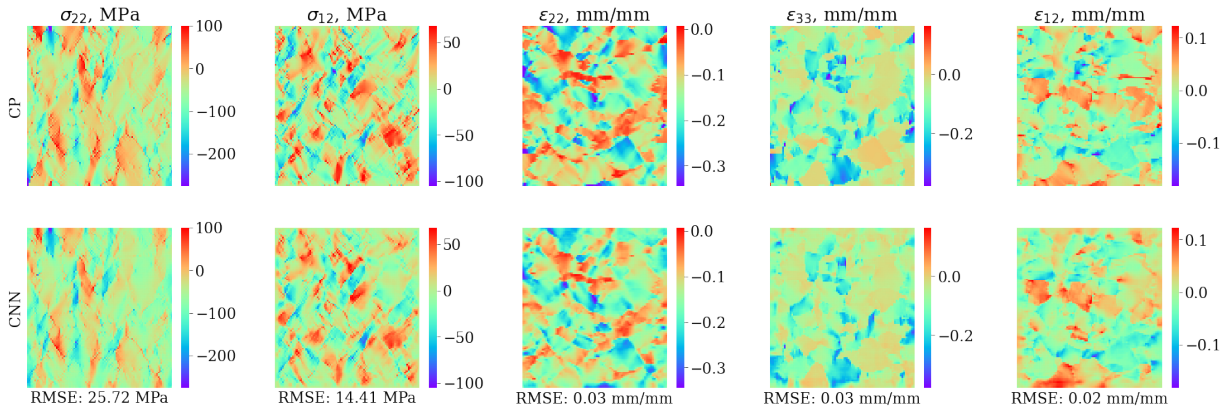

Figure 6.19: Prediction for AA6061 microstructure; the first row displays the crystal plasticity finite element predictions for the last timestep of the deformation, and the second row the convolutional neural network predictions for the last timestep of the deformation.

Table 6.8: Application to AA5754 and AA6061 aluminium alloys: element-wise RMSE errors calculated across all timesteps for all predicted variables.

Case	σ_{11}, **MPa**	σ_{22}, **MPa**	σ_{12}, **MPa**	ε_{11}	ε_{22}	ε_{33}	ε_{12}
AA5754	15.96	15.78	10.06	0.02	0.02	0.02	0.01
AA6061	17.49	18.46	9.86	0.02	0.02	0.02	0.01

CNN predictions were in excellent agreement with the predictions of CPFEM simulations for this microstructure. The microstructure of AA6061 exhibited a completely new grain morphology as well as an overall texture. The CNN predictions were also accurate for AA6061, which demonstrated the flexibility of the proposed CNN for predictions for a wide range of materials.

6.4.2 Extension to Strain Localisation Prediction

In addition to the work presented above, the convolutional neural network's capability to predict strain localisation was investigated. The CPFEM data was simulated up to 44% of macroscopic ε_{11}^{M} strain, and the CNN was retrained on this data. Similarly to the previous training approach, the network was trained on $10,000$ samples, validated on $1,000$ samples, and tested on $1,000$ samples. Final training and validation errors (normalised data) were

Table 6.9: Application to AA5754 and AA6061 aluminium alloys: element-wise RMSE errors calculated only for the last timesteps for all predicted variables.

Case	σ_{11}, **MPa**	σ_{22}, **MPa**	σ_{12}, **MPa**	ε_{11}	ε_{22}	ε_{33}	ε_{12}
AA5754	24.75	22.73	14.91	0.03	0.03	0.03	0.02
AA6061	26.84	25.72	14.41	0.03	0.03	0.03	0.02

equal to 1.3×10^{-3} after 60 training epochs, and the test error was equal to 1.2×10^{-3}. A low training error confirmed that the CNN learned the local strain distributions. However, CNN could not predict the correct intensity of the strain localisation formations. The average predictions were fairly accurate. For brevity and discussion purposes, only the example of predictions for AA5754 microstructure are presented. In detail, predictions for ε_{11} strain and σ_{11} stress components are analysed visually, and the rest components predictions were assessed with RMSE across all timesteps and for the last timestep of the deformation.

Aluminium alloy AA5754 microstructure was used to assess the predictive capabilities of the CNNs to predict strain localisation. Figure 6.20, (a), shows the input microstructure, averaged strain response, and the difference errors for four strain levels in the form of a histogram. The averaged ε_{11} strain predicted with neural network is predicted exceptionally accurately and has a minor error. The difference error plot also displays that most of the errors are situated in the proximity of zero, with the majority of the errors being less than 0.3 in their absolute value. However, the visualisations of the FE meshes (figure 6.20, (b)) reveal that the neural network, while captures the localisation formation location, does not predict the correct intensity of hot spots, as highlighted by red ellipses on the plot. Outside of the localisation region, local ε_{11} strains are accurately captured by the convolutional neural network. The overall RMSE for ε_{11} component is equal to 0.16, and the last timestep RMSE is 0.26. Both of the errors are reported in the table 6.10.

The analysis of the σ_{11} stress component prediction is demonstrated in figure 6.21. Part (a) of the figure displays the input microstructure, averaged stress error and the difference errors in the form of histogram. The averaged stress is captured well by the neural network, and the RMSE of the averaged predictions is 3.38 MPa. The CNN has successfully captured local stress relaxation, resulting in highly-accurate predictions of the averaged σ_{11} stress. Histogram confirms confirms the accuracy of the CNN by demonstrating that great majority of local stress prediction errors are close to zero. The majority of the difference errors is less than 250 MPa in their absolute value. As in the case with the prediction

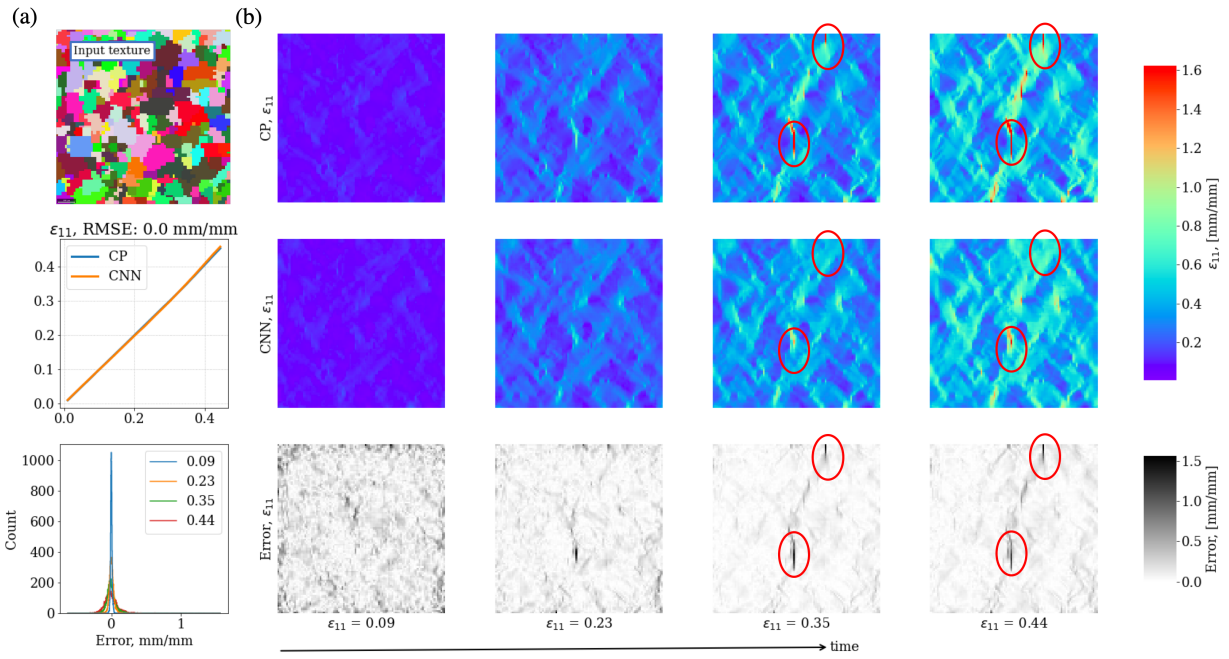

Figure 6.20: Predictions for an AA5754 microstructure (extension to localisation predictions); (a): input microstructure, ε_{11} predictions for crystal plasticity and convolutional neural network predictions averaged across all elements of the mesh, and histogram showing the difference error distribution; (b): ε_{11} strain partitioning evolution for four strain levels. The first row displays crystal plasticity predictions, the second row displays convolutional neural network predictions, and the third row shows the error map, which displays the absolute element-wise difference error between the crystal plasticity and convolutional neural network predictions. Strain localisation intensity is underpredicted by the CNN, as highlighted by red ellipses. The predictions outside the localisation region and the averaged ε_{11} strain are captured accurately.

of the ε_{11} strain component, the predictions of the stress hotspot location is correct, but the intensity of the predicted value is not captured accurately, as highlighted with red ellipses on the figure. The other areas of the FE mesh are in excellent agreement with crystal plasticity simulations. Notably, that the local stress relaxation is predicted quite well by the CNN, as highlighted with blue arrows on the figure. This is also reflected on the averaged flow curve: the stress decreases after the material's maximum resistance to loading was achieved. The error map validates the observations: most of the errors are close to zero, apart from the localisation spots. The overall RMSE for the σ_{11} predictions

was equal to 80.8 MPa, and the last timestep RMSE was equal to 166.39 MPa. The errors are reported in the table 6.10.

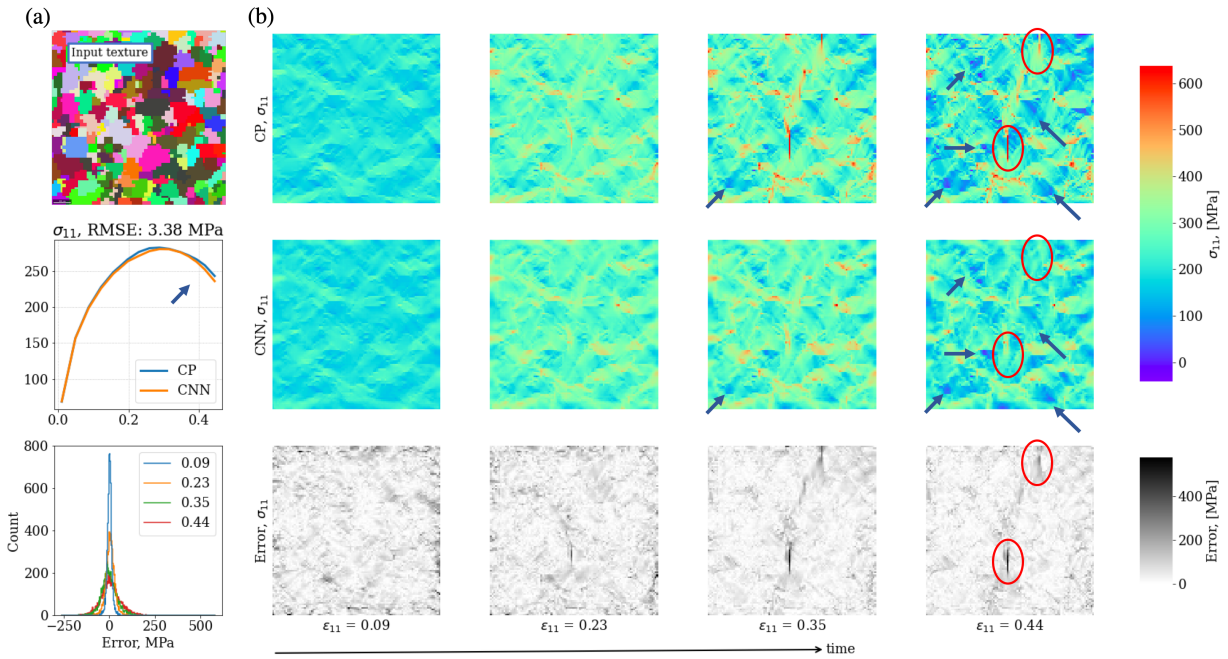

Figure 6.21: Predictions for an AA5754 microstructure (extension to localisation predictions); (a): input microstructure, σ_{11} predictions for crystal plasticity and convolutional neural network predictions averaged across all elements of the mesh, and histogram showing the difference error distribution; (b): σ_{11} strain partitioning evolution for four strain levels. The first row displays crystal plasticity predictions, the second row displays convolutional neural network predictions, and the third row shows the error map, which displays the absolute element-wise difference error between the crystal plasticity and convolutional neural network predictions. The intensity of stress localisation is underpredicted, as highlighted with red ellipses. Local stress relaxation is captured well by the CNN, as well as the averaged stress response, as demonstrated by the arrows.

The errors for the other variables are reported in the table 6.10. Similarly to the results for ε_{11} and σ_{11}, the RMSEs for σ_{22}, σ_{12}, ε_{22}, ε_{33}, and ε_{12} are higher than in the cases where deformation is predicted up to 22%. The authors attempted to train deeper networks, adding more data to the training set, which did not help achieve better localisation predictions. The other types of networks that can be considered for application of the localisation predictions could be convolutional long short-term memory networks

(CNN-LSTM), a type of recurrent neural network. They could be beneficial for predictions of spatial-sequential data such as deformation evolution of finite element mesh. These networks were not considered in this research as they are out of the scope of this work.

Table 6.10: Element-wise RMSE errors calculated across all timesteps for all predicted variables (extension to localisation predictions).

Case	σ_{11}, **MPa**	σ_{22}, **MPa**	σ_{12}, **MPa**	ε_{11}	ε_{22}	ε_{33}	ε_{12}
All timesteps	80.8	60.36	37.51	0.16	0.12	0.11	0.07
Last timestep	116.39	77.71	53.77	0.26	0.19	0.16	0.12

6.4.3 Runtime Comparison

The results have demonstrated the predictive capabilities of the proposed CNN to predict stress and strain partitioning of FE mesh. Besides the predictive accuracy, CNNs have another beneficial quality: computational efficiency. Crystal plasticity simulations are computationally demanding [206], especially when it comes to finite elements applications. Crystal plasticity finite element models computational time increases exponentially with the increase of elements in the FE mesh [137]. The exponential increase in computational time would not be the case for neural networks due to the relative simplicity of the performed computations in the networks. CNN computations include operations such as matrix multiplication, addition, and application of non-linearity to the weights in a layer, while crystal plasticity methods require solving highly non-linear computational equations. Therefore the usability of the networks becomes evident for such problems as material design, as multiple microstructures' deformation behaviour can be evaluated by neural networks simultaneously in a matter of seconds or minutes. Crystal plasticity simulations would require computationally demanding simulations, usually performed on high-performance computing clusters. If the number of microstructures requiring deformation behaviour evaluation is large, crystal plasticity simulation may not be feasible due to computational limitations, while machine learning methods are fast and becoming more and more accessible nowadays.

A number of studies compared the runtime of crystal plasticity compared to machine learning models [134, 135, 137, 140, 245] and highlighted the computational time superiority

of machine learning models to crystal plasticity models. The computational time comparison has also been conducted in this study and confirmed the computational efficiency of the neural networks compared to crystal plasticity methods. Figure 6.22 demonstrated runtime comparison for CPFEM and CNN models. The compassion in seconds for 1 and 10 simulations is demonstrated for predictions up to 22% and 44% of ε_{11}^{M} strain. A logarithmic scale was used to highlight the magnitudes of difference in the runtime. For example, to predict local stress and strain values up to 22% strain, the CPFEM model takes 30 minutes for one simulation and 300 minutes for 10 simulations, while the CNN model takes 56.3 ms and 470 ms correspondingly. For 1 and 10 simulations for localisation predictions, the CPFEM model requires 71 minute and 710 minutes correspondingly, while CNN requires 71 ms and 650 ms correspondingly. Therefore, machine learning enables enormous time savings and results in over 99.9% faster predictions.

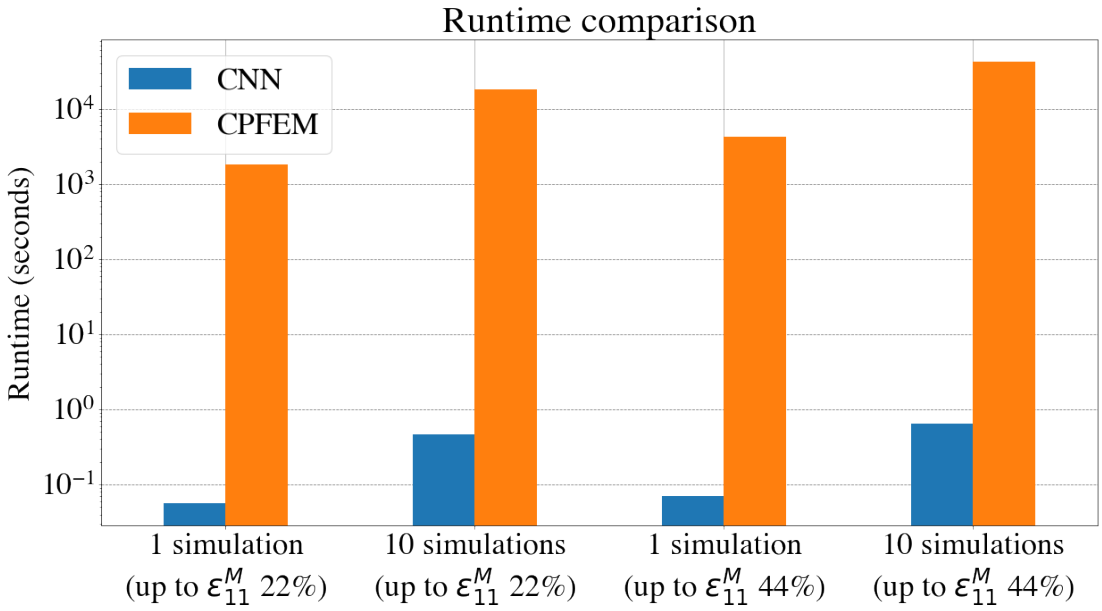

Figure 6.22: Runtime comparison between CPFEM and CNN. Vertical axis is displayed using logarithmic scale to emphasize the time difference between the models.

6.5 Chapter Conclusions

In this paper, a CNN-based framework to achieve high-fidelity predictions of local stress and strain evolution is proposed. The proposed framework resulted in training an optimal CNN

on a CPFEM dataset and included: feature engineering, synthesis of microstructures, data preprocessing, dataset design, and optimal architecture search. The important summary points, observations, and conclusions are as follows:

- The result of the framework was a deep convolutional neural networks which enabled accurate crystal plasticity finite element predictions for materials.
- The training dataset consisted of synthetically generated microstructures and two material parameters: initial hardness and initial hardening modulus. The microstructures were generated using Voronoi tessellations; each microstructure was unique and contained varying number of grains. The material hardening parameters were sampled using Sobol sequences. The orientations for generated microstructures were randomly sampled from the dataset of orientations. The orientation dataset was generated using AA5754 EBSD scan, and contained $8,750$ orientations.
- The CNN architecture was selected based on the ten different architecture trial training on partial dataset ($5,000$ samples) for 15 epochs. Then, the best performing neural network was trained on the full dataset for 100 training epochs. The training consisted of $10,000$ samples, the validation and test sets consisted of $1,000$ samples each.
- The CNN's predictive capabilities were validated using a test set containing completely new synthetically generated microstructures. Then the CNN was validated for two actual material microstructures (AA5754 and AA6061). The grain morphology within the AA5754 microstructure was different from the typical synthetically generated microstructure. The texture and the grain morphology of the AA6061 microstructure were also completely new to the CNN.
- The network has demonstrated excellent agreement with crystal plasticity simulations for test sets and for both real material samples.
- Test set median prediction error (RMSE across all timesteps) for σ_{11} was 16.14 MPa and for ε_{11} was 0.02. Test set worst prediction error (RMSE across all timesteps) for σ_{11} was 22.69 MPa and for ε_{11} was 0.04. Test set best prediction error (RMSE across all timesteps) for σ_{11} was 15.37 MPa and for ε_{11} was 0.02. The other components errors were also reported. In the worst prediction case, the CNN could not predict the localisation which appeared earlier during the deformation process due to specific texture morphology. The cases with localisation formations were not common within the dataset. Therefore the network did not learn the localisation behaviour, resulting

in higher error. The median and best errors were close in values, which confirm the accuracy of the neural network. Visually, the meshes also confirm the excellent agreement between CPFEM and CNN predictions.

- The network demonstrated excellent predictive capabilities for real AA5754 microstructure. The orientation distribution was similar to the orientation distributions in microstructures from the synthesized dataset, but the grain morphology was different. The real AA5754 grain morphology was rather distorted than a smooth and similar size and shape of grains in the synthetically generated material. Despite that microstructural difference, the network predictions demonstrated excellent agreement with crystal plasticity results. The predictions were compared visually for four strain levels, and the agreement between the results was evident. The prediction error (RMSE across all timesteps) for σ_{11} was 15.96 MPa and for ε_{11} was 0.02. The other components for strain and strain were also reported. The error results were comparable to the best test cases predictions, thus confirming the accuracy of the neural network.

- The network also demonstrated excellent predictive capabilities for real Al 6061 microstructure. For this microstructure, the orientation distribution was different compared to the orientation distributions present in the synthesised dataset. The grain morphology differed from both the synthesised dataset and AA5754 microstructure. Despite that microstructural difference, the network predictions demonstrated excellent agreement with crystal plasticity results. The predictions were compared visually for four strain levels, and the agreement between the results was evident. The prediction error (RMSE across all timesteps) for σ_{11} was 17.49 MPa and for ε_{11} was 0.02. The other components for strain and strain were also reported. The error results were comparable to the best test cases predictions, thus confirming the accuracy of the neural network and its applicability outside of the training dataset.

- The network was tested to predict localisation prediction. The proposed CNN successfully captured stress and strain responses averaged across most of elements of a mesh. Overall, except for predictions for localisation intensity, the predictive performance of the CNN showed good agreement with CP simulations, including unloading behaviour predictions.

- The presented approach to CPFEM predictions has been computationally efficient compared to the crystal plasticity finite element model. The trained convolutional neural network has shown up to 99.9% computational speedup: up to 65 ms versus up to 11 hours and 50 minutes for ten simulations. As a direction to further study, such

machine learning models can be applied to the microstructural design problems and save a considerable amount of time on crystal plasticity simulations, as conventional crystal plasticity models are very computationally demanding.

Chapter 7

Conclusions and Future Work

7.1 Summary & Key Conclusions

The goal of this research was to develop a machine learning-based framework to enable high-fidelity predictions of material deformation. The work was divided into two major parts. The first goal was to develop a machine learning framework that can serve as a single-crystal deformation model to accurately predict stress-strain behaviour and texture evolution for complex strain paths. The constituents of the framework included an ensemble of artificial neural networks and a CP-update algorithm to accommodate for complex strain-paths deformation behaviour predictions while having ANNs trained only on monotonic loading scenarios. The framework was successfully implemented and validated against the Taylor-type CP model for monotonic and non-monotonic strain paths for single crystal and polycrystalline aggregates. The second goal was to develop a machine learning framework to achieve accurate local stress and strain evolution predictions for a wide range of material microstructures. A single convolutional neural network was implemented and trained to predict stress and strain partitionings of an aluminium microstructure under a proportional loading condition similar to uniaxial tension. The CNN was trained on crystal plasticity finite element simulations for synthetically generated microstructures with defined material parameters. The resultant CNN was successfully implemented and validated against CPFEM for a new set of synthetic microstructures. The flexibility of the CNN was demonstrated by its validation for two microstructures of actual materials (AA5754 and AA6061). Both frameworks demonstrated substantial reductions in computational time compared to CP models without significant sacrifice in accuracies. The thesis key conclusions are as follows:

7.1.1 A new ANN based crystal plasticity model for FCC materials and its application to non-monotonic strain paths

- A machine learning- and crystal plasticity-based framework is proposed to model stress-strain and texture evolution for a wide variety of the FCC family crystals under non-monotonic strain path.
- The proposed framework consists of an ensemble of artificial neural networks trained only on monotonic loading examples. The implemented crystal plasticity-based algorithm makes it possible to predict the complex cases of non-monotonic loadings using those networks. The presented framework allows accurate material behaviour predictions on any, monotonic or non-monotonic, strain path. The predictions include full stress-strain predictions and texture evolution.
- The proposed framework was validated for a wide variety of strain paths: uniaxial tension, compression, simple shear, equibiaxial tension, tension-compression-tension, compression- tension-compression, cyclic shear, and arbitrary non-monotonic loading.
- The framework demonstrated excellent predictive capabilities for polycrystal simulations. The two cases were considered in this research: shear followed by tension, and arbitrary non-monotonic loading.
- The framework demonstrated excellent ability to predict texture evolution under plane-strain compression which is the major strain path during rolling of sheet metals.
- One of the most important outcomes of the current work is the proof of feasibility: it is possible to utilize machine learning to capture the crystal plasticity predictions of stress-strain properties and texture evolution. The presented results suggest that machine learning methods can be successfully applied to perform crystal plasticity prediction of material deformation behaviour.
- The presented approach is computationally efficient and shows up to 99.9 % computational speedup compared to Taylor-type crystal plasticity model.

7.1.2 Convolutional neural networks applications to crystal plasticity finite element predictions

- In this work, the CNN-based framework which enables rapid full-field stress and strain evolution prediction is proposed.

- The training dataset consisted of synthetically generated microstructures and two material parameters: initial hardness and initial hardening modulus.
- The microstructures were generated using Voronoi tessellations; each microstructure was unique and contained varying number of grains. The material hardening parameters were sampled using Sobol sequences. The orientations for generated microstructures were randomly sampled from the dataset of orientations. The orientation dataset was generated using AA5754 EBSD scan, and contained $8,750$ orientations.
- The CNN's predictive capabilities were successfully validated using a test set containing completely new synthetically generated microstructures.
- The CNN was successfully validated for two actual material microstructures (AA5754 and AA6061). The grain morphology within the AA5754 microstructure was different from the typical synthetically generated microstructure. The texture and the grain morphology of the AA6061 microstructure were also completely new to the CNN.
- The network was tested to predict localisation prediction. The proposed CNN successfully captured stress and strain responses averaged across most of elements of a mesh. Overall, except for predictions for localisation intensity, the predictive performance of the CNN showed good agreement with CP simulations, including unloading behaviour predictions.
- The presented approach is computationally efficient and shows up to 99.9 % computational speedup compared to crystal plasticity finite element method model.

7.2 Future Work

The frameworks presented in this research provide an excellent tool for rapid and accurate high-fidelity predictions of the deformation behaviour of polycrystalline solids. The framework presented in chapter 5 can be extended for using more material parameters, e.g., strain rate, and other parameters related to chosen hardening models.

The framework presented in chapter 6 had a limitation in predictions of strain localisation. Logically, this limitation should be addressed. The ways to resolve this limitation include implementing time-dependent neural networks, such as recurrent convolutional neural networks, or creating other strategies for localisation prediction. Once this limitation

is addressed, the next step would be to extend the model to other strain paths, monotonic and non-monotonic, and incorporate texture evolution prediction into a model.

Finally, the logical extension of both presented frameworks is the design and incorporation of a machine learning model into finite element software to produce rapid predictions of material deformations.

References

[1] W. Muhammad, U. Ali, A. P. Brahme, J. Kang, R. K. Mishra, and K. Inal, "Experimental analyses and numerical modeling of texture evolution and the development of surface roughness during bending of an extruded aluminum alloy using a multiscale modeling framework," *International Journal of Plasticity*, vol. 117, pp. 93–121, 2017.

[2] T. O. Mason, "Ceramic composition and properties | britannica." `https://www.britannica.com/topic/ceramic-composition-and-properties-103137`. (Accessed on 07/15/2019).

[3] D. R. Askeland, P. P. Fulay, and W. J. Wright, *The science and engineering of materials*. Nelson Education, 2011.

[4] W. Hutchinson, *An introduction to textures in metals*. Institution of Metallurgists, 1979.

[5] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*, vol. 112. Springer, 2013.

[6] "The neural network zoo - the asimov institute." `https://www.asimovinstitute.org/neural-network-zoo/`, September 2016. (Accessed on 07/15/2019).

[7] "Neural network zoo prequel: Cells and layers - the asimov institute." `https://web.archive.org/web/20190330183748/http://www.asimovinstitute.org/neural-network-zoo-prequel-cells-layers/`, March 2017. (Accessed on 07/15/2019).

[8] S. Ruder, "An overview of gradient descent optimization algorithms," pp. 1–14, 2016.

[9] A. Karpathy *et al.*, "Cs231n convolutional neural networks for visual recognition," *Neural networks*, vol. 1, 2016.

[10] M. Mozaffar, R. Bostanabad, W. Chen, K. Ehmann, J. Cao, and M. Bessa, "Deep learning predicts path-dependent plasticity," *Proceedings of the National Academy of Sciences*, vol. 116, no. 52, pp. 26414–26420, 2019.

[11] A. Beniwal, R. Dadhich, and A. Alankar, "Deep learning based predictive modeling for structure-property linkages," *Materialia*, vol. 8, p. 100435, 2019.

[12] A. Frankel, K. Tachida, and R. Jones, "Prediction of the evolution of the stress field of polycrystals undergoing elastic-plastic deformation with a hybrid neural network model," *Machine Learning: Science and Technology*, vol. 1, no. 3, p. 035005, 2020.

[13] P. Rajpurkar, J. Irvin, A. Bagul, D. Ding, T. Duan, H. Mehta, B. Yang, K. Zhu, D. Laird, R. L. Ball, *et al.*, "Mura: Large dataset for abnormality detection in musculoskeletal radiographs," *arXiv preprint arXiv:1712.06957*, 2017.

[14] P. K. Mall, P. K. Singh, and D. Yadav, "Glcm based feature extraction and medical x-ray image classification using machine learning techniques," in *2019 IEEE Conference on Information and Communication Technology*, pp. 1–6, IEEE, 2019.

[15] M. Roberts, D. Driggs, M. Thorpe, J. Gilbey, M. Yeung, S. Ursprung, A. I. Aviles-Rivero, C. Etmann, C. McCague, L. Beer, *et al.*, "Common pitfalls and recommendations for using machine learning to detect and prognosticate for covid-19 using chest radiographs and ct scans," *Nature Machine Intelligence*, vol. 3, no. 3, pp. 199–217, 2021.

[16] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[17] M. Liang and X. Hu, "Recurrent convolutional neural network for object recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3367–3375, 2015.

[18] A. Holzinger, "Trends in interactive knowledge discovery for personalized medicine: cognitive science meets machine learning," *The IEEE intelligent informatics bulletin*, vol. 15, no. 1, pp. 6–14, 2014.

[19] M. E. Ozer, P. O. Sarica, and K. Y. Arga, "New machine learning applications to accelerate personalized medicine in breast cancer: rise of the support vector machines," *Omics: a journal of integrative biology*, vol. 24, no. 5, pp. 241–246, 2020.

[20] L. Deng and X. Li, "Machine learning paradigms for speech recognition: An overview," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 5, pp. 1060–1089, 2013.

[21] P. K. Chan, M. V. Mahoney, and M. H. Arshad, "A machine learning approach to anomaly detection," tech. rep., 2003.

[22] N. Subrahmanya, Y. C. Shin, and P. H. Meckl, "A bayesian machine learning method for sensor selection and fusion with application to on-board fault diagnostics," *Mechanical systems and signal processing*, vol. 24, no. 1, pp. 182–192, 2010.

[23] G. Harshvardhan, M. K. Gourisaria, M. Pandey, and S. S. Rautaray, "A comprehensive survey and analysis of generative models in machine learning," *Computer Science Review*, vol. 38, p. 100285, 2020.

[24] Y. Kang, H. Yin, and C. Berger, "Test your self-driving algorithm: An overview of publicly available driving datasets and virtual testing environments," *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 2, pp. 171–185, 2019.

[25] S. Park and J.-Y. Choi, "Malware detection in self-driving vehicles using machine learning algorithms," *Journal of advanced transportation*, vol. 2020, 2020.

[26] S. Inc., "Global car sales 2010-2021." `https://www.statista.com/statistics/200002/international-car-sales-since-1990/`, Nov 2021. (Accessed on 01/1/2022).

[27] P. Baguley, R. Roy, and J. Watson, "Cost of physical vehicle crash testing," in *Collaborative Product and Service Life Cycle Management for a Sustainable World*, pp. 113–121, Springer, 2008.

[28] G. I. Taylor, "The mechanism of plastic deformation of crystals. part i.—theoretical," *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, vol. 145, no. 855, pp. 362–387, 1934.

[29] E. Schmid, "Yield point of crystals, critical shear stress law," in *Proceedings of the First International Congress for Applied Mechanics, Delft*, 1924.

[30] U. F. Kocks, C. N. Tomé, and H.-R. Wenk, *Texture and anisotropy: preferred orientations in polycrystals and their effect on materials properties*. Cambridge university press, 1998.

[31] O. Engler and V. Randle, *Introduction to texture analysis: macrotexture, microtexture, and orientation mapping*. CRC press, 2009.

[32] O. Instruments, "Ebsd explained: from data acquisition to advanced analysis," *Abingdon: Oxford Instruments*, pp. 1–23, 2015.

[33] A. Kochendorfer, "Reine und angewandte metallkunde," 1941.

[34] J. Bishop and R. Hill, "Xlvi. a theory of the plastic distortion of a polycrystalline aggregate under combined stresses.," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 42, no. 327, pp. 414–427, 1951.

[35] J. Bishop and R. Hill, "Cxxviii. a theoretical derivation of the plastic properties of a polycrystalline face-centred metal," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 42, no. 334, pp. 1298–1307, 1951.

[36] R. J. Asaro and A. Needleman, "Overview no. 42 texture development and strain hardening in rate dependent polycrystals," *Acta metallurgica*, vol. 33, no. 6, pp. 923–953, 1985.

[37] G. I. Taylor, "Analysis of plastic strain in a cubic crystal," *Stephen Timoshenko 60th Anniversary Volume*, pp. 218–224, 1938.

[38] G. I. Taylor and C. F. Elam, "Bakerian lecture: the distortion of an aluminium crystal during a tensile test," *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, vol. 102, no. 719, pp. 643–667, 1923.

[39] K. Inal, K. Neale, P. Wu, and S. MacEwen, "Numerical simulation of large deformation polycrystalline plasticity," in *Mathematical Modeling in Metal Processing and Manufacturing (COM2000)*, 2000.

[40] K. Inal, P. Wu, and K. Neale, "Finite element analysis of localization in fcc polycrystalline sheets under plane stress tension," *International Journal of Solids and Structures*, vol. 39, no. 13-14, pp. 3469–3486, 2002.

[41] A. P. Brahme, K. Inal, R. K. Mishra, and S. Saimoto, "The backstress effect of evolving deformation boundaries in fcc polycrystals," *International journal of plasticity*, vol. 27, no. 8, pp. 1252–1266, 2011.

[42] P. Wu, S. MacEwen, D. Lloyd, and K. Neale, "A mesoscopic approach for predicting sheet metal formability," *Modelling and Simulation in Materials Science and Engineering*, vol. 12, no. 3, p. 511, 2004.

[43] J. Moor, "The dartmouth college artificial intelligence conference: The next fifty years," *AI Magazine*, vol. 27, pp. 87–91, Winter 2006. Copyright - Copyright Association for the Advancement of Artificial Intelligence Winter 2006; Document feature - Photographs; ; Last updated - 2017-10-27.

[44] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[45] F. Chollet, *Deep Learning mit Python und Keras: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek*. MITP-Verlags GmbH & Co. KG, 2018.

[46] T. Glasmachers, "Limits of End-to-End Learning," pp. 1–15, 2017.

[47] L. Muda, M. Begam, and I. Elamvazuthi, "Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and dynamic time warping (dtw) techniques," *arXiv preprint arXiv:1003.4083*, 2010.

[48] H. Lee, P. Pham, Y. Largman, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Advances in neural information processing systems*, pp. 1096–1104, 2009.

[49] D. A. Winkler and T. C. Le, "Performance of deep and shallow neural networks, the universal approximation theorem, activity cliffs, and qsar," *Molecular informatics*, vol. 36, no. 1-2, p. 1600118, 2017.

[50] H. Robbins and S. Monro, "A stochastic approximation method," *The annals of mathematical statistics*, pp. 400–407, 1951.

[51] R. Sutton, "Two problems with back propagation and other steepest descent learning procedures for networks," in *Proceedings of the Eighth Annual Conference of the Cognitive Science Society, 1986*, pp. 823–832, 1986.

[52] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural networks*, vol. 12, no. 1, pp. 145–151, 1999.

[53] J. S. Hunter, "The exponentially weighted moving average," *Journal of quality technology*, vol. 18, no. 4, pp. 203–210, 1986.

[54] Y. E. Nesterov, "A method for solving the convex programming problem with convergence rate o (1/kˆ 2)," in *Dokl. akad. nauk Sssr*, vol. 269, pp. 543–547, 1983.

[55] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[56] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[57] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[58] T. Dozat, "Incorporating nesterov momentum into adam," 2016.

[59] J. Burgess, B. Gallagher, D. D. Jensen, B. N. Levine, *et al.*, "Maxprop: Routing for vehicle-based disruption-tolerant networks.," in *Infocom*, vol. 6, Barcelona, Spain, 2006.

[60] R. Kohavi *et al.*, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Ijcai*, vol. 14-2, pp. 1137–1145, Montreal, Canada, 1995.

[61] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[62] A. Y. Ng, "Feature selection, l 1 vs. l 2 regularization, and rotational invariance," in *Proceedings of the twenty-first international conference on Machine learning*, p. 78, ACM, 2004.

[63] A. Krogh and J. A. Hertz, "A simple weight decay can improve generalization," in *Advances in neural information processing systems*, pp. 950–957, 1992.

[64] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[65] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, pp. 448–456, PMLR, 2015.

[66] S. Santurkar, D. Tsipras, A. Ilyas, and A. Mądry, "How does batch normalization help optimization?," in *Proceedings of the 32nd international conference on neural information processing systems*, pp. 2488–2498, 2018.

[67] F. Chollet *et al.*, "Keras." `https://keras.io`, 2015.

[68] S. L. Smith, P.-J. Kindermans, C. Ying, and Q. V. Le, "Don't decay the learning rate, increase the batch size," *arXiv preprint arXiv:1711.00489*, 2017.

[69] M. Folk, G. Heber, Q. Koziol, E. Pourmal, and D. Robinson, "An overview of the hdf5 technology suite and its applications," in *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases*, pp. 36–47, ACM, 2011.

[70] R. v. Mises, "Mechanik der festen körper im plastisch- deformablen zustand," *Nachrichten von der Gesellschaft der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse*, vol. 1913, pp. 582–592, 1913.

[71] A. Hershey, "The elasticity of an isotropic aggregate of anisotropic cubic crystals," *Journal of Applied mechanics-transactions of the ASME*, vol. 21, no. 3, pp. 236–240, 1954.

[72] Y. Lou and J. W. Yoon, "Anisotropic yield function based on stress invariants for bcc and fcc metals and its extension to ductile fracture criterion," *International Journal of Plasticity*, vol. 101, pp. 125–155, 2018.

[73] D. C. Drucker and W. Prager, "Soil mechanics and plastic analysis or limit design," *Quarterly of applied mathematics*, vol. 10, no. 2, pp. 157–165, 1952.

[74] S. Isavand and A. Assempour, "Strain localization and deformation behavior in ferrite-pearlite steel unraveled by high-resolution in-situ testing integrated with crystal plasticity simulations," *International Journal of Mechanical Sciences*, vol. 200, p. 106441, 2021.

[75] M. Jafari, S. Ziaei-Rad, N. Saeidi, and M. Jamshidian, "Micromechanical analysis of martensite distribution on strain localization in dual phase steels by scanning electron microscopy and crystal plasticity simulation," *Materials Science and Engineering: A*, vol. 670, pp. 57–67, 2016.

[76] X. Liao, Y. Zhao, Y. Zhu, R. Valiev, and D. Gunderov, "Grain-size effect on the deformation mechanisms of nanostructured copper processed by high-pressure torsion," *Journal of applied physics*, vol. 96, no. 1, pp. 636–640, 2004.

[77] J. Sun, P. Trimby, F. Yan, X. Liao, N. Tao, and J. Wang, "Grain size effect on deformation twinning propensity in ultrafine-grained hexagonal close-packed titanium," *Scripta Materialia*, vol. 69, no. 5, pp. 428–431, 2013.

[78] W. Muhammad, R. K. Sabat, A. P. Brahme, J. Kang, R. K. Mishra, B. Shalchi-Amirkhiz, J. Hirsch, and K. Inal, "Deformation banding in a precipitation hardened aluminum alloy during simple shear deformation," *Scripta Materialia*, vol. 162, pp. 300–305, 2019.

[79] Y. L. Li, C. P. Kohar, R. K. Mishra, and K. Inal, "A new crystal plasticity constitutive model for simulating precipitation-hardenable aluminum alloys," *International Journal of Plasticity*, p. 102759, 2020.

[80] Y. L. Li, C. P. Kohar, W. Muhammad, and K. Inal, "Precipitation kinetics and crystal plasticity modeling of artificially aged aa6061," *International Journal of Plasticity*, p. 103241, 2022.

[81] C. C. Tasan, M. Diehl, D. Yan, C. Zambaldi, P. Shanthraj, F. Roters, and D. Raabe, "Integrated experimental–simulation analysis of stress and strain partitioning in multiphase alloys," *Acta Materialia*, vol. 81, pp. 386–400, 2014.

[82] D. S. Connolly, C. P. Kohar, W. Muhammad, L. G. Hector Jr, R. K. Mishra, and K. Inal, "A coupled thermomechanical crystal plasticity model applied to quenched and partitioned steel," *International Journal of Plasticity*, vol. 133, p. 102757, 2020.

[83] D. Connolly, C. Kohar, and K. Inal, "A novel crystal plasticity model incorporating transformation induced plasticity for a wide range of strain rates and temperatures," *International Journal of Plasticity*, p. 103188, 2022.

[84] G. Sachs, "Zur ableitung einer fliessbedingung," in *Mitteilungen der deutschen Materialprüfungsanstalten*, pp. 94–97, Springer, 1929.

[85] G. I. Taylor, "Plastic strain in metals," *J. Inst. Metals*, vol. 62, pp. 307–324, 1938.

[86] J. Rossiter, A. Brahme, M. Simha, K. Inal, and R. Mishra, "A new crystal plasticity scheme for explicit time integration codes to simulate deformation in 3d microstructures: effects of strain path, strain rate and thermal softening on localized deformation in the aluminum alloy 5754 during simple shear," *International Journal of Plasticity*, vol. 26, no. 12, pp. 1702–1725, 2010.

[87] H. Bunge and C. Esling, "Texture development by plastic deformation," *Scripta metallurgica*, vol. 18, no. 3, pp. 191–195, 1984.

[88] A. Clément and P. Coulomb, "Eulerian simulation of deformation textures," *Scripta Metallurgica*, vol. 13, no. 9, pp. 899–901, 1979.

[89] D. S. Li, H. Garmestani, and S. Schoenfeld, "Evolution of crystal orientation distribution coefficients during plastic deformation," *Scripta Materialia*, vol. 49, no. 9, pp. 867–872, 2003.

[90] S. R. Kalidindi, H. K. Duvvuru, and M. Knezevic, "Spectral calibration of crystal plasticity models," *Acta Materialia*, vol. 54, no. 7, pp. 1795–1804, 2006.

[91] M. Knezevic, S. R. Kalidindi, and D. Fullwood, "Computationally efficient database and spectral interpolation for fully plastic Taylor-type crystal plasticity calculations of face-centered cubic polycrystals," *International Journal of Plasticity*, vol. 24, no. 7, pp. 1264–1276, 2008.

[92] S. R. Kalidindi, S. R. Niezgoda, G. Landi, S. Vachhani, and T. Fast, "A novel framework for building materials knowledge systems," *Computers, Materials, & Continua*, vol. 17, no. 2, pp. 103–125, 2010.

[93] T. Fast and S. R. Kalidindi, "Formulation and calibration of higher-order elastic localization relationships using the MKS approach," *Acta Materialia*, vol. 59, no. 11, pp. 4595–4605, 2011.

[94] M. I. Latypov, L. S. Toth, and S. R. Kalidindi, "Materials knowledge system for nonlinear composites," *Computer Methods in Applied Mechanics and Engineering*, vol. 346, pp. 180–196, 2019.

[95] A. Gupta, A. Cecen, S. Goyal, A. K. Singh, and S. R. Kalidindi, "Structure-property linkages using a data science approach: Application to a non-metallic inclusion/steel composite system," *Acta Materialia*, vol. 91, pp. 239–254, 2015.

[96] T. Fast, S. R. Niezgoda, and S. R. Kalidindi, "A new framework for computationally efficient structure–structure evolution linkages to facilitate high-fidelity scale bridging in multi-scale materials models," *Acta Materialia*, vol. 59, no. 2, pp. 699–707, 2011.

[97] Y. C. Yabansu, D. K. Patel, and S. R. Kalidindi, "Calibrated localization relationships for elastic response of polycrystalline aggregates," *Acta Materialia*, vol. 81, pp. 151–160, 2014.

[98] S. R. Kalidindi, S. R. Niezgoda, and A. A. Salem, "Microstructure informatics using higher-order statistics and efficient data-mining protocols," *Jom*, vol. 63, no. 4, pp. 34–41, 2011.

[99] Y. C. Yabansu and S. R. Kalidindi, "Representation and calibration of elastic localization kernels for a broad class of cubic polycrystals," *Acta Materialia*, vol. 94, pp. 26–35, 2015.

[100] H. F. Al-Harbi, G. Landi, and S. R. Kalidindi, "Multi-scale modeling of the elastic response of a structural component made from a composite material using the materials knowledge system," *Modelling and Simulation in Materials Science and Engineering*, vol. 20, no. 5, p. 055001, 2012.

[101] D. B. Brough, D. Wheeler, J. A. Warren, and S. R. Kalidindi, "Microstructure-based knowledge systems for capturing process-structure evolution linkages," *Current Opinion in Solid State and Materials Science*, vol. 21, no. 3, pp. 129–140, 2017.

[102] M. W. Priddy, N. H. Paulson, S. R. Kalidindi, and D. L. McDowell, "Strategies for rapid parametric assessment of microstructure-sensitive fatigue for hcp polycrystals," *International Journal of Fatigue*, vol. 104, pp. 231–242, 2017.

[103] D. B. Brough, D. Wheeler, and S. R. Kalidindi, "Materials knowledge systems in python—a data science framework for accelerated development of hierarchical materials," *Integrating materials and manufacturing innovation*, vol. 6, no. 1, pp. 36–53, 2017.

[104] D. B. Brough, A. Kannan, B. Haaland, D. G. Bucknall, and S. R. Kalidindi, "Extraction of process-structure evolution linkages from x-ray scattering measurements using dimensionality reduction and time series analysis," *Integrating Materials and Manufacturing Innovation*, vol. 6, no. 2, pp. 147–159, 2017.

[105] S. R. Kalidindi, "Computationally efficient, fully coupled multiscale modeling of materials phenomena using calibrated localization linkages," *ISRN Materials Science*, vol. 2012, 2012.

[106] S. R. Kalidindi, "Microstructure informatics," in *Informatics for Materials Science and Engineering*, pp. 443–466, Elsevier, 2013.

[107] D. M. de Oca Zapiain, E. Popova, and S. R. Kalidindi, "Prediction of microscale plastic strain rate fields in two-phase composites subjected to an arbitrary macroscale

strain rate using the materials knowledge system framework," *Acta Materialia*, vol. 141, pp. 230–240, 2017.

[108] N. H. Paulson, M. W. Priddy, D. L. McDowell, and S. R. Kalidindi, "Reduced-order structure-property linkages for polycrystalline microstructures based on 2-point statistics," *Acta Materialia*, vol. 129, pp. 428–438, 2017.

[109] J. Ghaboussi, J. Garrett Jr, and X. Wu, "Knowledge-based modeling of material behavior with neural networks," *Journal of engineering mechanics*, vol. 117, no. 1, pp. 132–153, 1991.

[110] X. Li, C. C. Roth, and D. Mohr, "Machine-learning based temperature-and rate-dependent plasticity model: application to analysis of fracture experiments on dp steel," *International Journal of Plasticity*, vol. 118, pp. 320–344, 2019.

[111] M. B. Gorji, M. Mozaffar, J. N. Heidenreich, J. Cao, and D. Mohr, "On the potential of recurrent neural networks for modeling path dependent plasticity," *Journal of the Mechanics and Physics of Solids*, p. 103972, 2020.

[112] Z. Liu, C. Wu, and M. Koishi, "A deep material network for multiscale topology learning and accelerated nonlinear modeling of heterogeneous materials," *Computer Methods in Applied Mechanics and Engineering*, 2018.

[113] A. Brahme, M. Winning, and D. Raabe, "Prediction of cold rolling texture of steels using an artificial neural network," *Computational Materials Science*, vol. 46, no. 4, pp. 800–804, 2009.

[114] K. Hansen, G. Montavon, F. Biegler, S. Fazli, M. Rupp, M. Scheffler, O. A. Von Lilienfeld, A. Tkatchenko, and K. R. Müller, "Assessment and validation of machine learning methods for predicting molecular atomization energies," *Journal of Chemical Theory and Computation*, vol. 9, no. 8, pp. 3404–3419, 2013.

[115] W. Muhammad, A. P. Brahme, O. Ibragimova, J. Kang, and K. Inal, "A machine learning framework to predict local strain distribution and the evolution of plastic anisotropy & fracture in additively manufactured alloys," *International Journal of Plasticity*, vol. 136, p. 102867, 2020.

[116] S. G. Javed, A. Khan, A. Majid, A. M. Mirza, and J. Bashir, "Lattice constant prediction of orthorhombic ABO3 perovskites using support vector machines," *Computational Materials Science*, vol. 39, no. 3, pp. 627–634, 2007.

[117] A. Majid, A. Khan, G. Javed, and A. M. Mirza, "Lattice constant prediction of cubic and monoclinic perovskites using neural networks and support vector regression," *Computational Materials Science*, vol. 50, no. 2, pp. 363–372, 2010.

[118] A. Majid, A. Khan, and T. S. Choi, "Predicting lattice constant of complex cubic perovskites using computational intelligence," *Computational Materials Science*, vol. 50, no. 6, pp. 1879–1888, 2011.

[119] R. Liu, Y. C. Yabansu, A. Agrawal, S. R. Kalidindi, and A. N. Choudhary, "Machine learning approaches for elastic localization linkages in high-contrast composite materials," *Integrating Materials and Manufacturing Innovation*, vol. 4, no. 1, p. 13, 2015.

[120] R. Liu, Y. C. Yabansu, Z. Yang, A. N. Choudhary, S. R. Kalidindi, and A. Agrawal, "Context Aware Machine Learning Approaches for Modeling Elastic Localization in Three-Dimensional Composite Microstructures," *Integrating Materials and Manufacturing Innovation*, vol. 6, no. 2, pp. 160–171, 2017.

[121] M. Bessa, R. Bostanabad, Z. Liu, A. Hu, D. W. Apley, C. Brinson, W. Chen, and W. K. Liu, "A framework for data-driven analysis of materials under uncertainty: Countering the curse of dimensionality," *Computer Methods in Applied Mechanics and Engineering*, vol. 320, pp. 633–667, 2017.

[122] V. N. Garla and C. Brandt, "Ontology-guided feature engineering for clinical text classification," *Journal of biomedical informatics*, vol. 45, no. 5, pp. 992–998, 2012.

[123] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*, pp. 24–29, IEEE, 2011.

[124] G. V. Cormack, J. M. G. Hidalgo, and E. P. Sánz, "Feature engineering for mobile (sms) spam filtering," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 871–872, ACM, 2007.

[125] A. Severyn and A. Moschitti, "Automatic feature engineering for answer selection and extraction," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 458–467, 2013.

[126] I. Sobol, "The distribution of points in a cube and the approximate evaluation of integrals, zh. vychisl. mat. i mat. fiz. 7 (1967), 784–802," 1967.

[127] I. M. Sobol, "Uniformly distributed sequences with an additional uniform property," *USSR Computational Mathematics and Mathematical Physics*, vol. 16, no. 5, pp. 236–242, 1976.

[128] Y. Zhang and C. Ling, "A strategy to apply machine learning to small datasets in materials science," *npj Computational Materials*, vol. 4, no. 1, pp. 28–33, 2018.

[129] J. Lee, A. Seko, K. Shitara, K. Nakayama, and I. Tanaka, "Prediction model of band gap for inorganic compounds by combination of density functional theory calculations and machine learning techniques," *Physical Review B*, vol. 93, no. 11, p. 115104, 2016.

[130] J. P. Perdew, K. Burke, and M. Ernzerhof, "Generalized gradient approximation made simple," *Physical review letters*, vol. 77, no. 18, p. 3865, 1996.

[131] Z. Yang, S. Papanikolaou, A. C. Reid, W.-k. Liao, A. N. Choudhary, C. Campbell, and A. Agrawal, "Learning to predict crystal plasticity at the nanoscale: Deep residual networks and size effects in uniaxial compression discrete dislocation simulations," *Scientific reports*, vol. 10, no. 1, pp. 1–14, 2020.

[132] Z. Liu, C. Wu, and M. Koishi, "Transfer learning of deep material network for seamless structure–property predictions," *Computational Mechanics*, vol. 64, no. 2, pp. 451–465, 2019.

[133] Y. Kim, Y. Kim, C. Yang, K. Park, G. X. Gu, and S. Ryu, "Deep learning framework for material design space exploration using active transfer learning and data augmentation," *npj Computational Materials*, vol. 7, no. 1, pp. 1–7, 2021.

[134] A. Mangal and E. A. Holm, "Applied machine learning to predict stress hotspots i: Face centered cubic materials," *International Journal of Plasticity*, vol. 111, pp. 122–134, 2018.

[135] A. Mangal and E. A. Holm, "Applied machine learning to predict stress hotspots ii: Hexagonal close packed materials," *International Journal of Plasticity*, vol. 114, pp. 1–14, 2019.

[136] A. Mangal and E. A. Holm, "A dataset of synthetic face centered cubic 3D polycrystalline microstructures, grain-wise microstructural descriptors and grain averaged stress fields under uniaxial tensile deformation," *Data in Brief*, vol. 19, pp. 2029–2036, 2018.

[137] U. Ali, W. Muhammad, A. Brahme, O. Skiba, and K. Inal, "Application of artificial neural networks in micromechanics for polycrystalline metals," *International Journal of Plasticity*, vol. 120, pp. 205–219, 2019.

[138] Y. Miyazawa, F. Briffod, T. Shiraiwa, and M. Enoki, "Prediction of cyclic stress–strain property of steels by crystal plasticity simulations and machine learning," *Materials*, vol. 12, no. 22, p. 3668, 2019.

[139] A. Pandey and R. Pokharel, "Machine learning enabled surrogate crystal plasticity model for spatially resolved 3d orientation evolution under uniaxial tension," *arXiv preprint arXiv:2005.00951*, 2020.

[140] P. Acar, "Machine learning reinforced crystal plasticity modeling under experimental uncertainty," in *AIAA Scitech 2020 Forum*, p. 1152, 2020.

[141] S. Papanikolaou, M. Tzimas, A. C. Reid, and S. A. Langer, "Spatial strain correlations, machine learning, and deformation history in crystal plasticity," *Physical Review E*, vol. 99, no. 5, p. 053003, 2019.

[142] M. Yuan, S. Paradiso, B. Meredig, and S. R. Niezgoda, "Machine learning–based reduce order crystal plasticity modeling for icme applications," *Integrating Materials and Manufacturing Innovation*, vol. 7, no. 4, pp. 214–230, 2018.

[143] A. Cecen, H. Dai, Y. C. Yabansu, S. R. Kalidindi, and L. Song, "Material structure-property linkages using three-dimensional convolutional neural networks," *Acta Materialia*, vol. 146, pp. 76–84, 2018.

[144] G. Zhao, B. Zhou, K. Wang, R. Jiang, and M. Xu, "Respond-cam: Analyzing deep models for 3d imaging data by visualizations," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 485–492, Springer, 2018.

[145] A. L. Frankel, R. E. Jones, C. Alleman, and J. A. Templeton, "Predicting the mechanical response of oligocrystals with deep learning," *Computational Materials Science*, vol. 169, p. 109099, 2019.

[146] M. F. Horstemeyer and D. J. Bammann, "Historical review of internal state variable theory for inelasticity," *International Journal of Plasticity*, vol. 26, no. 9, pp. 1310–1334, 2010.

[147] P. Follansbee and U. Kocks, "A constitutive description of the deformation of copper based on the use of the mechanical threshold stress as an internal state variable," *Acta Metallurgica*, vol. 36, no. 1, pp. 81–93, 1988.

[148] P. Perzyna, "Internal state variable description of dynamic fracture of ductile solids," *International Journal of Solids and Structures*, vol. 22, no. 7, pp. 797–818, 1986.

[149] F. Roters, D. Raabe, and G. Gottstein, "Work hardening in heterogeneous alloys—a microstructural approach based on three internal state variables," *Acta materialia*, vol. 48, no. 17, pp. 4181–4189, 2000.

[150] X. Fan and H. Yang, "Internal-state-variable based self-consistent constitutive modeling for hot working of two-phase titanium alloys coupling microstructure evolution," *International Journal of Plasticity*, vol. 27, no. 11, pp. 1833–1852, 2011.

[151] F. Roters, P. Eisenlohr, L. Hantcherli, D. D. Tjahjanto, T. R. Bieler, and D. Raabe, "Overview of constitutive laws, kinematics, homogenization and multiscale methods in crystal plasticity finite-element modeling: Theory, experiments, applications," *Acta Materialia*, vol. 58, no. 4, pp. 1152–1211, 2010.

[152] M. Zecevic, Y. P. Korkolis, T. Kuwabara, and M. Knezevic, "Dual-phase steel sheets under cyclic tension–compression to large strains: experiments and crystal plasticity modeling," *Journal of the Mechanics and Physics of Solids*, vol. 96, pp. 65–87, 2016.

[153] A. Izadbakhsh, K. Inal, R. K. Mishra, and M. Niewczas, "New crystal plasticity constitutive model for large strain deformation in single crystals of magnesium," *Computational materials science*, vol. 50, no. 7, pp. 2185–2202, 2011.

[154] A. Alankar, P. Eisenlohr, and D. Raabe, "A dislocation density-based crystal plasticity constitutive model for prismatic slip in α-titanium," *Acta Materialia*, vol. 59, no. 18, pp. 7003–7009, 2011.

[155] A. Arsenlis and D. M. Parks, "Modeling the evolution of crystallographic dislocation density in crystal plasticity," *Journal of the Mechanics and Physics of Solids*, vol. 50, no. 9, pp. 1979–2009, 2002.

[156] L. Evers, W. Brekelmans, and M. Geers, "Scale dependent crystal plasticity framework with dislocation density and grain boundary effects," *International Journal of solids and structures*, vol. 41, no. 18-19, pp. 5209–5230, 2004.

[157] E. Popova, Y. Staraselski, A. Brahme, R. Mishra, and K. Inal, "Coupled crystal plasticity–probabilistic cellular automata approach to model dynamic recrystallization in magnesium alloys," *International Journal of Plasticity*, vol. 66, pp. 85–102, 2015.

[158] E. Cyr, M. Mohammadi, A. Brahme, R. K. Mishra, and K. Inal, "Modeling the formability of aluminum alloys at elevated temperatures using a new thermo-elasto-viscoplastic crystal plasticity framework," *International Journal of Mechanical Sciences*, vol. 128, pp. 312–325, 2017.

[159] W. G. Feather, S. Ghorbanpour, D. J. Savage, M. Ardeljan, M. Jahedi, B. A. McWilliams, N. Gupta, C. Xiang, S. C. Vogel, and M. Knezevic, "Mechanical response, twinning, and texture evolution of we43 magnesium-rare earth alloy as a function of strain rate: experiments and multi-level crystal plasticity modeling," *International Journal of Plasticity*, vol. 120, pp. 180–204, 2019.

[160] H. Farooq, G. Cailletaud, S. Forest, and D. Ryckelynck, "Crystal plasticity modeling of the cyclic behavior of polycrystalline aggregates under non-symmetric uniaxial loading: Global and local analyses," *International Journal of Plasticity*, vol. 126, p. 102619, 2020.

[161] K. Inal, "Numerical simulation of sheet metal forming processes and localized deformation phenomena for fcc polycrystals.," *PhD Thesis. Université de Sherbrooke, Sherbrooke, QC*, 2003.

[162] D. S. Joseph, P. Chakraborty, and S. Ghosh, "Wavelet transformation based multi-time scaling method for crystal plasticity fe simulations under cyclic loading," *Computer methods in applied mechanics and engineering*, vol. 199, no. 33-36, pp. 2177–2194, 2010.

[163] P. Chakraborty, D. S. Joseph, and S. Ghosh, "Wavelet transformation based multi-time scale crystal plasticity fem for cyclic deformation in titanium alloys under dwell load," *Finite elements in analysis and design*, vol. 47, no. 6, pp. 610–618, 2011.

[164] M. Knezevic and S. R. Kalidindi, "Crystal plasticity modeling of microstructure evolution and mechanical fields during processing of metals using spectral databases," *JOM*, vol. 69, no. 5, pp. 830–838, 2017.

[165] A. Gupta, M. B. Bettaieb, F. Abed-Meraim, and S. R. Kalidindi, "Computationally efficient predictions of crystal plasticity based forming limit diagrams using a spectral database," *International Journal of Plasticity*, vol. 103, pp. 168–187, 2018.

[166] A. Eghtesad, M. Zecevic, R. A. Lebensohn, R. J. McCabe, and M. Knezevic, "Spectral database constitutive representation within a spectral micromechanical solver for computationally efficient polycrystal plasticity modelling," *Computational Mechanics*, vol. 61, no. 1-2, pp. 89–104, 2018.

[167] S. R. Kalidindi, A. Gupta, and E. Popova, "Computationally efficient crystal plasticity simulations using spectral databases," *Handbook of Materials Modeling: Methods: Theory and Modeling*, pp. 1685–1710, 2020.

[168] S. R. Kalidindi, *Hierarchical materials informatics: novel analytics for materials data*. Elsevier, 2015.

[169] S. R. Kalidindi, "Data science and cyberinfrastructure: critical enablers for accelerated development of hierarchical materials," *International Materials Reviews*, vol. 60, no. 3, pp. 150–168, 2015.

[170] J. Ghaboussi and D. Sidarta, "New nested adaptive neural networks (nann) for constitutive modeling," *Computers and Geotechnics*, vol. 22, no. 1, pp. 29–52, 1998.

[171] M. Lefik and B. A. Schrefler, "Artificial neural network as an incremental non-linear constitutive model for a finite element code," *Computer methods in applied mechanics and engineering*, vol. 192, no. 28-30, pp. 3265–3283, 2003.

[172] O. Sabokpa, A. Zarei-Hanzaki, H. Abedi, and N. Haghdadi, "Artificial neural network modeling to predict the high temperature flow behavior of an az81 magnesium alloy," *Materials & Design*, vol. 39, pp. 390–396, 2012.

[173] N. Haghdadi, A. Zarei-Hanzaki, A. Khalesian, and H. Abedi, "Artificial neural network modeling to predict the hot deformation behavior of an a356 aluminum alloy," *Materials & Design*, vol. 49, pp. 386–391, 2013.

[174] B. Plunkett, O. Cazacu, and F. Barlat, "Orthotropic yield criteria for description of the anisotropy in tension and compression of sheet metals," *International Journal of Plasticity*, vol. 24, no. 5, pp. 847–866, 2008.

[175] A. Abedini, C. Butcher, and M. J. Worswick, "Experimental fracture characterisation of an anisotropic magnesium alloy sheet in proportional and non-proportional loading conditions," *International Journal of Solids and Structures*, vol. 144, pp. 1–19, 2018.

[176] X. Xiao, L. Chen, L. Yu, and H. Duan, "Modelling nano-indentation of ion-irradiated fcc single crystals by strain-gradient crystal plasticity theory," *International Journal of Plasticity*, vol. 116, pp. 216–231, 2019.

[177] K. Inal, K. Neale, and A. Aboutajeddine, "Forming limit comparisons for fcc and bcc sheets," *International Journal of Plasticity*, vol. 21, no. 6, pp. 1255–1266, 2005.

[178] A. E. Tallman, L. P. Swiler, Y. Wang, and D. L. McDowell, "Reconciled top-down and bottom-up hierarchical multiscale calibration of bcc fe crystal plasticity," *International Journal for Multiscale Computational Engineering*, vol. 15, no. 6, 2017.

[179] P. Zhang, C. P. Kohar, A. P. Brahme, S.-H. Choi, R. K. Mishra, and K. Inal, "A crystal plasticity formulation for simulating the formability of a transformation induced plasticity steel," *Journal of Materials Processing Technology*, p. 116493, 2019.

[180] H. Wang, P. Wu, J. Wang, and C. Tomé, "A crystal plasticity model for hexagonal close packed (hcp) crystals including twinning and de-twinning mechanisms," *International Journal of Plasticity*, vol. 49, pp. 36–52, 2013.

[181] C. Paramatmuni and A. K. Kanjarla, "A crystal plasticity fft based study of deformation twinning, anisotropy and micromechanics in hcp materials: Application to az31 alloy," *International Journal of Plasticity*, vol. 113, pp. 269–290, 2019.

[182] Y. W. Chang and R. J. Asaro, "An experimental study of shear localization in aluminum-copper single crystals," *Acta Metallurgica*, vol. 29, no. 1, pp. 241–257, 1981.

[183] Y. Zhang and C. Ling, "A strategy to apply machine learning to small datasets in materials science," *Npj Computational Materials*, vol. 4, no. 1, pp. 1–8, 2018.

[184] A. K. Jain and B. Chandrasekaran, "39 dimensionality and sample size considerations in pattern recognition practice," *Handbook of statistics*, vol. 2, pp. 835–855, 1982.

[185] S. J. Raudys, A. K. Jain, *et al.*, "Small sample size effects in statistical pattern recognition: Recommendations for practitioners," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 13, no. 3, pp. 252–264, 1991.

[186] P. Bratley and B. L. Fox, "Algorithm 659: Implementing sobol's quasirandom sequence generator," *ACM Transactions on Mathematical Software (TOMS)*, vol. 14, no. 1, pp. 88–100, 1988.

[187] S. Burhenne, D. Jacob, and G. P. Henze, "Sampling based on sobol'sequences for monte carlo techniques applied to building simulations," in *Proc. Int. Conf. Build. Simulat*, pp. 1816–1823, 2011.

[188] J. Santiago, M. Claeys-Bruno, and M. Sergent, "Construction of space-filling designs using wsp algorithm for high dimensional spaces," *Chemometrics and Intelligent Laboratory Systems*, vol. 113, pp. 26–31, 2012.

[189] C. P. Kohar, J. L. Bassani, A. Brahme, W. Muhammad, R. K. Mishra, and K. Inal, "A new multi-scale framework to incorporate microstructure evolution in phenomenological plasticity: theory, explicit finite element formulation, implementation and validation," *International Journal of Plasticity*, vol. 117, pp. 122–156, 2019.

[190] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*, vol. 1. Springer series in statistics New York, 2001.

[191] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[192] C. M. Bishop, "Neural networks and their applications," *Review of scientific instruments*, vol. 65, no. 6, pp. 1803–1832, 1994.

[193] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, no. 11, p. e00938, 2018.

[194] J. Koutnik, K. Greff, F. Gomez, and J. Schmidhuber, "A clockwork rnn," in *International Conference on Machine Learning*, pp. 1863–1871, PMLR, 2014.

[195] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[196] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[197] G. Keren and B. Schuller, "Convolutional rnn: an enhanced model for extracting features from sequential data," in *2016 International Joint Conference on Neural Networks (IJCNN)*, pp. 3412–3419, IEEE, 2016.

[198] A. Graves, N. Jaitly, and A.-r. Mohamed, "Hybrid speech recognition with deep bidirectional lstm," in *2013 IEEE workshop on automatic speech recognition and understanding*, pp. 273–278, IEEE, 2013.

[199] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[200] S. Sharma, "Activation functions in neural networks," *Towards Data Science*, vol. 6, 2017.

[201] M. R. Zadeh, S. Amin, D. Khalili, and V. P. Singh, "Daily outflow prediction by multi layer perceptron with logistic sigmoid and tangent sigmoid activation functions," *Water resources management*, vol. 24, no. 11, pp. 2673–2688, 2010.

[202] A. Pandey, A. S. Khan, E.-Y. Kim, S.-H. Choi, and T. Gnäupel-Herold, "Experimental and numerical investigations of yield surface, texture, and deformation mechanisms in aa5754 over low to high temperatures and strain rates," *International Journal of Plasticity*, vol. 41, pp. 165–188, 2013.

[203] U. Ali, D. Odoh, W. Muhammad, A. Brahme, R. K. Mishra, M. Wells, and K. Inal, "Experimental investigation and through process crystal plasticity-static recrystallization modeling of temperature and strain rate effects during hot compression of aa6063," *Materials Science and Engineering: A*, vol. 700, pp. 374–386, 2017.

[204] W. Muhammad, A. P. Brahme, J. Kang, R. K. Mishra, and K. Inal, "Experimental and numerical investigation of texture evolution and the effects of intragranular backstresses in aluminum alloys subjected to large strain cyclic deformation," *International Journal of Plasticity*, vol. 93, pp. 137–163, 2017.

[205] S. Banovic, M. Iadicola, and T. Foecke, "Textural development of aa 5754 sheet deformed under in-plane biaxial tension," *Metallurgical and Materials Transactions A*, vol. 39, no. 9, p. 2246, 2008.

[206] A. P. Brahme, K. Inal, R. K. Mishra, and S. Saimoto, "A new strain hardening model for rate-dependent crystal plasticity," *Computational materials science*, vol. 50, no. 10, pp. 2898–2908, 2011.

[207] A. Brahme, Y. Staraselski, K. Inal, and R. K. Mishra, "Determination of the minimum scan size to obtain representative textures by electron backscatter diffraction," *Metallurgical and materials transactions A*, vol. 43, no. 13, pp. 5298–5307, 2012.

[208] A. Shamsipur, A. Anvari, and A. Keyvani, "Improvement of microstructure and corrosion properties of friction stir welded aa5754 by adding zn interlayer," *International Journal of Minerals, Metallurgy, and Materials*, vol. 25, no. 8, pp. 967–973, 2018.

[209] J. S. Nagra, A. Brahme, J. Levesque, R. Mishra, R. A. Lebensohn, and K. Inal, "A new micromechanics based full field numerical framework to simulate the effects of dynamic recrystallization on the formability of hcp metals," *International Journal of Plasticity*, vol. 125, pp. 210–234, 2020.

[210] R. K. Sabat, W. Muhammad, R. K. Mishra, and K. Inal, "Effect of microstructure on fracture in age hardenable al alloys," *Philosophical Magazine*, vol. 100, no. 11, pp. 1476–1498, 2020.

[211] P. Agrawal, S. Shukla, S. Thapliyal, P. Agrawal, S. S. Nene, R. S. Mishra, B. A. McWilliams, and K. C. Cho, "Microstructure–property correlation in a laser powder bed fusion processed high-strength af-9628 steel," *Advanced Engineering Materials*, vol. 23, no. 1, p. 2000845, 2021.

[212] D. Kuhlmann-Wilsdorf, "The theory of dislocation-based crystal plasticity," *Philosophical Magazine A*, vol. 79, no. 4, pp. 955–1008, 1999.

[213] B. Hansen, I. Beyerlein, C. Bronkhorst, E. Cerreta, and D. Dennis-Koller, "A dislocation-based multi-rate single crystal plasticity model," *International Journal of Plasticity*, vol. 44, pp. 129–146, 2013.

[214] R. Kondo, Y. Tadano, and K. Shizawa, "A phase-field model of twinning and detwinning coupled with dislocation-based crystal plasticity for hcp metals," *Computational materials science*, vol. 95, pp. 672–683, 2014.

[215] G. Liu, H. Mo, J. Wang, and Y. Shen, "Coupled crystal plasticity finite element-phase field model with kinetics-controlled twinning mechanism for hexagonal metals," *Acta Materialia*, vol. 202, pp. 399–416, 2021.

[216] N. Jia, F. Roters, P. Eisenlohr, C. Kords, and D. Raabe, "Non-crystallographic shear banding in crystal plasticity fem simulations: Example of texture evolution in α-brass," *Acta Materialia*, vol. 60, no. 3, pp. 1099–1115, 2012.

[217] S. L. Wong, M. Madivala, U. Prahl, F. Roters, and D. Raabe, "A crystal plasticity model for twinning-and transformation-induced plasticity," *Acta Materialia*, vol. 118, pp. 140–151, 2016.

[218] P. Zhang, C. P. Kohar, A. P. Brahme, S.-H. Choi, R. K. Mishra, and K. Inal, "A crystal plasticity formulation for simulating the formability of a transformation induced plasticity steel," *Journal of Materials Processing Technology*, vol. 287, p. 116493, 2021.

[219] A. Izadbakhsh, K. Inal, and R. K. Mishra, "Crystal plasticity based finite element modelling of large strain deformation in am30 magnesium alloy," *Modelling and Simulation in Materials Science and Engineering*, vol. 20, no. 3, p. 035016, 2012.

[220] F. Han, B. Tang, H. Kou, L. Cheng, J. Li, and Y. Feng, "Static recrystallization simulations by coupling cellular automata and crystal plasticity finite element method using a physically based model for nucleation," *Journal of materials science*, vol. 49, no. 8, pp. 3253–3267, 2014.

[221] O. Diard, S. Leclercq, G. Rousselier, and G. Cailletaud, "Evaluation of finite element based analysis of 3d multicrystalline aggregates plasticity: Application to crystal plasticity model identification and the study of stress and strain fields near grain boundaries," *International Journal of Plasticity*, vol. 21, no. 4, pp. 691–722, 2005.

[222] M. Knezevic, B. Drach, M. Ardeljan, and I. J. Beyerlein, "Three dimensional predictions of grain scale plasticity and grain boundaries using crystal plasticity finite element models," *Computer Methods in Applied Mechanics and Engineering*, vol. 277, pp. 239–259, 2014.

[223] F. Qayyum, A. A. Chaudhry, S. Guk, M. Schmidtchen, R. Kawalla, and U. Prahl, "Effect of 3d representative volume element (rve) thickness on stress and strain partitioning in crystal plasticity simulations of multi-phase materials," *Crystals*, vol. 10, no. 10, p. 944, 2020.

[224] W. Liu, B. K. Chen, Y. Pang, and A. Najafzadeh, "A 3d phenomenological yield function with both in and out-of-plane mechanical anisotropy using full-field crystal plasticity spectral method for modelling sheet metal forming of strong textured aluminum alloy," *International Journal of Solids and Structures*, vol. 193, pp. 117–133, 2020.

[225] C. M. A. Iftikhar, Y. L. Li, C. P. Kohar, K. Inal, and A. S. Khan, "Evolution of subsequent yield surfaces with plastic deformation along proportional and non-proportional loading paths on annealed aa6061 alloy: Experiments and crystal plasticity finite element modeling," *International Journal of Plasticity*, p. 102956, 2021.

[226] Z.-h. Li, G.-w. Zhou, D.-y. Li, H.-m. Wang, W.-q. Tang, Y.-h. Peng, H. S. Zurob, and P.-d. Wu, "Crystal plasticity based modeling of grain boundary sliding in magnesium alloy az31b sheet," *Transactions of Nonferrous Metals Society of China*, vol. 31, no. 1, pp. 138–155, 2021.

[227] B. Ravaji and S. P. Joshi, "A crystal plasticity investigation of grain size-texture interaction in magnesium alloys," *Acta Materialia*, vol. 208, p. 116743, 2021.

[228] X. Fang, K. Ding, S. Janocha, C. Minnert, W. Rheinheimer, T. Frömling, K. Durst, A. Nakamura, and J. Rödel, "Nanoscale to microscale reversal in room-temperature

plasticity in srtio3 by tuning defect concentration," *Scripta Materialia*, vol. 188, pp. 228–232, 2020.

[229] D. Zhang, J. He, and J. Liang, "Creep rupture mechanism and microstructure evolution around film-cooling holes in nickel-based single crystal superalloy specimen," *Engineering Fracture Mechanics*, vol. 235, p. 107187, 2020.

[230] M. Wroński, M. A. Kumar, R. McCabe, K. Wierzbanowski, and C. Tomé, "Deformation behavior of cp-titanium under strain path changes: Experiment and crystal plasticity modeling," *International Journal of Plasticity*, p. 103129, 2021.

[231] Y. Mellbin, H. Hallberg, and M. Ristinmaa, "Accelerating crystal plasticity simulations using gpu multiprocessors," *International journal for numerical methods in engineering*, vol. 100, no. 2, pp. 111–135, 2014.

[232] A. Eghtesad, K. Germaschewski, R. A. Lebensohn, and M. Knezevic, "A multi-gpu implementation of a full-field crystal plasticity solver for efficient modeling of high-resolution microstructures," *Computer Physics Communications*, vol. 254, p. 107231, 2020.

[233] H. F. Alharbi and S. R. Kalidindi, "Crystal plasticity finite element simulations using a database of discrete fourier transforms," *International Journal of Plasticity*, vol. 66, pp. 71–84, 2015.

[234] M. Knezevic, H. F. Al-Harbi, and S. R. Kalidindi, "Crystal plasticity simulations using discrete fourier transforms," *Acta materialia*, vol. 57, no. 6, pp. 1777–1784, 2009.

[235] M. I. Latypov and S. R. Kalidindi, "Data-driven reduced order models for effective yield strength and partitioning of strain in multiphase materials," *Journal of Computational Physics*, vol. 346, pp. 242–261, 2017.

[236] A. E. Tallman, K. S. Stopka, L. P. Swiler, Y. Wang, S. R. Kalidindi, and D. L. McDowell, "Gaussian-process-driven adaptive sampling for reduced-order modeling of texture effects in polycrystalline alpha-ti," *JOM*, vol. 71, no. 8, pp. 2646–2656, 2019.

[237] R. Pidaparti and M. Palakal, "Material model for composites using neural networks," *AIAA journal*, vol. 31, no. 8, pp. 1533–1535, 1993.

[238] K. Rao and Y. Prasad, "Neural network approach to flow stress evaluation in hot deformation," *Journal of Materials Processing Technology*, vol. 53, no. 3-4, pp. 552–566, 1995.

[239] A. Mukherjee, S. Schmauder, M. Ru, *et al.*, "Artificial neural networks for the prediction of mechanical behavior of metal matrix composites," *Acta metallurgica et materialia*, vol. 43, no. 11, pp. 4083–4091, 1995.

[240] M.-S. Chun, J.-J. Yi, B. Jalal, and J. G. Lenard, "A comparative study of material flow stress modeling by artificial neural networks and statistical methods," *Transactions of the Korean Society of Mechanical Engineers A*, vol. 21, no. 5, pp. 828–834, 1997.

[241] D. Kim, D. Kim, and B. Kim, "The application of neural networks and statistical methods to process design in metal forming processes," *The International Journal of Advanced Manufacturing Technology*, vol. 15, no. 12, pp. 886–894, 1999.

[242] P. Cavaliere, "Flow curve prediction of an al-mmc under hot working conditions using neural networks," *Computational materials science*, vol. 38, no. 4, pp. 722–726, 2007.

[243] A. Tran and T. Wildey, "Solving stochastic inverse problems for property–structure linkages using data-consistent inversion and machine learning," *JOM*, vol. 73, no. 1, pp. 72–89, 2021.

[244] A. Pandey and R. Pokharel, "Machine learning based surrogate modeling approach for mapping crystal deformation in three dimensions," *Scripta Materialia*, vol. 193, pp. 1–5, 2021.

[245] O. Ibragimova, A. Brahme, W. Muhammad, J. Lévesque, and K. Inal, "A new ann based crystal plasticity model for fcc materials and its application to non-monotonic strain paths," *International Journal of Plasticity*, vol. 144, p. 103059, 2021.

[246] J. Raphanel, G. Ravichandran, and Y. Leroy, "Three-dimensional rate-dependent crystal plasticity based on runge–kutta algorithms for update and consistent linearization," *International Journal of Solids and Structures*, vol. 41, no. 22-23, pp. 5995–6021, 2004.

[247] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *2016 fourth international conference on 3D vision (3DV)*, pp. 565–571, IEEE, 2016.

[248] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[249] M. Sun, T. X. Han, M.-C. Liu, and A. Khodayari-Rostamabad, "Multiple instance learning convolutional neural networks for object recognition," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 3270–3275, IEEE, 2016.

[250] S. Khan, M. H. Javed, E. Ahmed, S. A. Shah, and S. U. Ali, "Facial recognition using convolutional neural networks and implementation on smart glasses," in *2019 International Conference on Information Science and Communication Technology (ICISCT)*, pp. 1–6, IEEE, 2019.

[251] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[252] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv preprint arXiv:1603.07285*, 2016.

[253] J. Bjorck, C. Gomes, B. Selman, and K. Q. Weinberger, "Understanding batch normalization," *arXiv preprint arXiv:1806.02375*, 2018.

[254] M. Simon, E. Rodner, and J. Denzler, "Imagenet pre-trained models with batch normalization," *arXiv preprint arXiv:1612.01452*, 2016.

[255] A. Ismail, S. A. Ahmad, A. C. Soh, K. Hassan, and H. H. Harith, "Improving convolutional neural network (cnn) architecture (minivggnet) with batch normalization and learning rate decay factor for image classification," *International Journal of Integrated Engineering*, vol. 11, no. 4, 2019.

[256] S. Ben-David, P. Hrubeš, S. Moran, A. Shpilka, and A. Yehudayoff, "Learnability can be undecidable," *Nature Machine Intelligence*, vol. 1, no. 1, p. 44, 2019.

[257] A. L. Samuel, "Some studies in machine learning using the game of checkers," *IBM Journal on Research and Development*, vol. 3, no. 3, p. 211, 1959.

[258] A. Abolhassani, J. Harner, M. Jaridi, and B. Gopalakrishnan, "Productivity enhancement strategies in north american automotive industry," *International Journal of Production Research*, vol. 56, no. 4, pp. 1414–1431, 2018.

[259] F. Roters, P. Eisenlohr, L. Hantcherli, D. D. Tjahjanto, T. R. Bieler, and D. Raabe, "Overview of constitutive laws, kinematics, homogenization and multiscale methods in crystal plasticity finite-element modeling: Theory, experiments, applications," *Acta Materialia*, vol. 58, no. 4, pp. 1152–1211, 2010.

[260] G. Tucker, "Texture and earing in deep drawing of aluminium," *Acta Metallurgica*, vol. 9, no. 4, pp. 275–286, 1961.

[261] F. Laves, "Plastische eigenschaften von kristallen und metallischen werkstoffen. von a. kochendörfer. (reine und angewandte metallkunde in einzeldarstellungen. herausgeg. von w. köster. bd. vii.) 312 s., 91 abb. j. springer, berlin 1941. pr. geh. rm. 27,—, geb. rm. 28,50," *Angewandte Chemie*, vol. 55, no. 1-2, pp. 18–18, 1942.

[262] K. Arulkumaran, A. Cully, and J. Togelius, "Alphastar: An evolutionary computation perspective," *arXiv preprint arXiv:1902.01724*, 2019.

[263] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2017.

[264] E. Lejon, P. Kyösti, and J. Lindström, "Machine learning for detection of anomalies in press-hardening: Selection of efficient methods," *Procedia CIRP*, vol. 72, pp. 1079–1083, 2018.

[265] J. Lindström, H. Larsson, M. Jonsson, and E. Lejon, "Towards intelligent and sustainable production: combining and integrating online predictive maintenance and continuous quality control," *Procedia CIRP*, vol. 63, pp. 443–448, 2017.

[266] R. Ramprasad, R. Batra, G. Pilania, A. Mannodi-Kanakkithodi, and C. Kim, "Machine learning in materials informatics: recent applications and prospects," *npj Computational Materials*, vol. 3, no. 1, p. 54, 2017.

[267] M. Yadav, P. Malhotra, L. Vig, K. Sriram, and G. Shroff, "Ode-augmented training improves anomaly detection in sensor data from machines," *arXiv preprint arXiv:1605.01534*, 2016.

[268] A. Rovinelli, M. D. Sangid, H. Proudhon, and W. Ludwig, "Using machine learning and a data-driven approach to identify the small fatigue crack driving force in polycrystalline materials," *npj Computational Materials*, vol. 4, no. 1, p. 35, 2018.

[269] J. Schmidhuber, "Deep Learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[270] P. M. Domingos, "A few useful things to know about machine learning.," *Commun. acm*, vol. 55, no. 10, pp. 78–87, 2012.

[271] N. Carlini and D. Wagner, "Audio adversarial examples: Targeted attacks on speech-to-text," in *2018 IEEE Security and Privacy Workshops (SPW)*, pp. 1–7, IEEE, 2018.

[272] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[273] The HDF Group, "Hierarchical Data Format, version 5," 1997-NNNN. http://www.hdfgroup.org/HDF5/.

[274] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[275] A. Ng, "Machine learning yearning: Technical strategy for ai engineers in the era of deep learning," 2019.

[276] E. F. Grant, "It | the new yorker." `https://www.newyorker.com/magazine/1952/08/02/it`, July 1952. (Accessed on 07/15/2019).

[277] S. R. Sree, S. Vyshnavi, and N. Jayapandian, "Real-world application of machine learning and deep learning," in *2019 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pp. 1069–1073, IEEE, 2019.

[278] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, "Zero-shot text-to-image generation," *arXiv preprint arXiv:2102.12092*, 2021.

[279] J. Vamathevan, D. Clark, P. Czodrowski, I. Dunham, E. Ferran, G. Lee, B. Li, A. Madabhushi, P. Shah, M. Spitzer, *et al.*, "Applications of machine learning in

drug discovery and development," *Nature Reviews Drug Discovery*, vol. 18, no. 6, pp. 463–477, 2019.

[280] O. Rudovic, J. Lee, M. Dai, B. Schuller, and R. W. Picard, "Personalized machine learning for robot perception of affect and engagement in autism therapy," *Science Robotics*, vol. 3, no. 19, 2018.

[281] M. Leo, M. Del Coco, P. Carcagni, C. Distante, M. Bernava, G. Pioggia, and G. Palestra, "Automatic emotion recognition in robot-children interaction for asd treatment," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 145–153, 2015.

[282] S. Malik, N. Kanwal, M. N. Asghar, M. A. A. Sadiq, I. Karamat, and M. Fleury, "Data driven approach for eye disease classification with machine learning," *Applied Sciences*, vol. 9, no. 14, p. 2789, 2019.

[283] C. Y. Cheung, F. Tang, D. S. W. Ting, G. S. W. Tan, and T. Y. Wong, "Artificial intelligence in diabetic eye disease screening," *The Asia-Pacific Journal of Ophthalmology*, vol. 8, no. 2, pp. 158–164, 2019.

[284] Q. Abbas, "Glaucoma-deep: detection of glaucoma eye disease on retinal fundus images using deep learning," *Int J Adv Comput Sci Appl*, vol. 8, no. 6, pp. 41–5, 2017.

[285] D. M. de Oca Zapiain and S. R. Kalidindi, "Localization models for the plastic response of polycrystalline materials using the material knowledge systems framework," *Modelling and Simulation in Materials Science and Engineering*, vol. 27, no. 7, p. 074008, 2019.

[286] H. Li and M. Fu, *Deformation-based processing of materials: Behavior, performance, modeling, and control*. Elsevier, 2019.

[287] H. Lim, J. Carroll, C. C. Battaile, T. Buchheit, B. Boyce, and C. Weinberger, "Grain-scale experimental validation of crystal plasticity finite element simulations of tantalum oligocrystals," *International Journal of Plasticity*, vol. 60, pp. 1–18, 2014.

[288] W. Muhammad, A. P. Brahme, R. K. Sabat, R. K. Mishra, and K. Inal, "A criterion for ductile failure in age-hardenable aluminum alloys," *Materials Science and Engineering: A*, vol. 759, pp. 613–623, 2019.

[289] J. R. Rice and D. M. Tracey, "On the ductile enlargement of voids in triaxial stress fields," *Journal of the Mechanics and Physics of Solids*, vol. 17, no. 3, pp. 201–217, 1969.

[290] H.-J. Bunge, *Texture analysis in materials science: mathematical methods*. Elsevier, 2013.