PREDICTING MECHANICAL BEHAVIOUR OF AUXETIC LATTICE STRUCTURES USING FINITE ELEMENT ANALYSIS AND MACHINE LEARNING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

YAMAN ARSLANCA

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
MECHANICAL ENGINEERING

DECEMBER 2024

Approval of the thesis:

PREDICTING MECHANICAL BEHAVIOUR OF AUXETIC LATTICE STRUCTURES USING FINITE ELEMENT ANALYSIS AND MACHINE LEARNING

submitted by **YAMAN ARSLANCA** in partial fulfillment of the requirements for the degree of **Master of Science in Mechanical Engineering Department, Middle East Technical University** by,

Prof. Dr. Naci Emre Altun
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Serkan Dağ
Head of Department, **Mechanical Engineering** _____

Assoc. Prof. Dr. Ulaş Yaman
Supervisor, **Mechanical Engineering, METU** _____

Assoc. Prof. Dr. Sezer Özerinç
Co-supervisor, **Mechanical Science and Engineering, UIUC** _____

Examining Committee Members:

Prof. Dr. Melik Dölen
Mechanical Engineering, METU _____

Assoc. Prof. Dr. Ulaş Yaman
Mechanical Engineering, METU _____

Prof. Dr. Özgür Ünver
Mechanical Engineering, TOBB ETU _____

Assoc. Prof. Dr. Elif Sürer
Informatics Institute, METU _____

Assist. Prof. Dr. Hakan Çalışkan
Mechanical Engineering, METU _____

Date: 03.12.2024

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: YAMAN ARSLANCA

Signature :

ABSTRACT

PREDICTING MECHANICAL BEHAVIOUR OF AUXETIC LATTICE STRUCTURES USING FINITE ELEMENT ANALYSIS AND MACHINE LEARNING

ARSLANCA, YAMAN
M.S., Department of Mechanical Engineering
Supervisor: Assoc. Prof. Dr. Ulaş Yaman
Co-Supervisor: Assoc. Prof. Dr. Sezer Özerinç

December 2024, 103 pages

Mechanical properties of auxetic lattice structures have been extensively researched in both academia and industry. Recent advancements in artificial intelligence, particularly in machine learning, have also been attracting significant attention. This work aims to utilize machine learning to investigate, analyze and predict the mechanical behaviour of auxetic double arrow-head lattice structures. A total of 1401 double arrow-head lattice structures were generated using finite element analysis in an automated manner. The analysis results, along with the input features, were used to train three different machine learning models: Neural Network, Random Forest, and Extreme Gradient Boosting. Prediction results from this training for eight output variables are presented, and optimization studies using the Pareto set and a genetic algorithm are conducted to identify the optimal design parameters for the structure.

Keywords: Auxetic Double Arrow-Head Lattice Structures, Finite Element Analysis, Machine Learning, Neural Networks

ÖZ

SONLU ELEMANLAR ANALİZİ VE MAKİNE ÖĞRENİMİ KULLANILARAK ÖKSETİK KAFES YAPILARININ MEKANİK DAVRANIŞININ TAHMİN EDİLMESİ

ARSLANCA, YAMAN
Yüksek Lisans, Makina Mühendisliği Bölümü
Tez Yöneticisi: Doç. Dr. Ulaş Yaman
Ortak Tez Yöneticisi: Doç. Dr. Sezer Özerinç

Aralık 2024 , 103 sayfa

Öksetik kafes yapılarının mekanik özellikleri hem akademide hem de endüstride kapsamlı bir şekilde araştırılmaktadır. Yapay zekadaki, özellikle makine öğrenimindeki son gelişmeler de önemli ilgi görmektedir. Bu çalışma, makine öğrenimini kullanarak öksetik çift ok başlı kafes yapıların mekanik davranışlarını araştırmayı, analiz etmeyi ve tahmin etmeyi amaçlamaktadır. Sonlu elemanlar analizi kullanılarak otomatik bir şekilde toplam 1401 çift ok uçlu kafes yapısı oluşturulmuştur. Analiz sonuçları, girdi parametreleriyle birlikte, üç farklı makine öğrenimi modelini eğitmek için kullanılmıştır: Sinir Ağları, Rastgele Orman ve Aşırı Gradyan Artırma. Bu eğitimden elde edilen tahmin sonuçları sekiz çıktı değişkeni için sunulmuş ve yapı için optimum tasarım parametrelerini belirlemek amacıyla Pareto kümesi ve genetik algoritma kullanılarak optimizasyon çalışmaları yürütülmüştür.

Anahtar Kelimeler: Öksetik Çift Ok Başlı Kafes Yapıları, Sonlu Elemanlar Analizi, Makine Öğrenmesi, Sinir Ağları

To my mother, to Prof. Dr. Refaat Alareer and to all who walk on the straight path,

ACKNOWLEDGMENTS

I would like to express my heartfelt gratitude to my mother, Zeliha Arslanca, for her unwavering support. I also extend my thanks to my supervisors, Assoc. Prof. Dr. Sezer Özerinç and Assoc. Prof. Dr. Ulaş Yaman, for their contribution and guidance throughout this study.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vi

ACKNOWLEDGMENTS viii

TABLE OF CONTENTS ix

LIST OF TABLES xii

LIST OF FIGURES xiii

LIST OF ABBREVIATIONS xvii

CHAPTERS

1 INTRODUCTION 1

1.1 Background 2

1.1.1 Additive Manufacturing 2

1.1.2 Lattice Structures 3

1.1.3 Finite Element Method 6

1.1.4 Machine Learning 7

1.2 Literature Review 10

1.2.1 Double Arrowhead Auxetic Lattice Structures 11

1.2.2 Numerical Simulations of Lattice Structures 12

1.2.3 Related Machine Learning Studies 14

1.3 Problem Statement 18
1.4 Objectives 21
1.5 Scope of the Study 23
2 DESIGN AND FINITE ELEMENT MODELING 25
2.1 Geometry of the Unit Cell 25
2.2 Formation of Lattice Structures 28
2.3 Finite Element Analysis 30
2.3.1 Material Selection 31
2.3.2 Discretization of the Structures 32
2.3.3 Boundary Conditions 33
2.3.4 Analysis Type 35
2.3.5 Comparison with a Literature Work 35
2.4 Analysis Results and Discussion 38
3 MACHINE LEARNING METHODOLOGY AND MODEL TRAINING . . 51
3.1 Pre-processing the Data 51
3.2 Hyperparameter Optimization 55
3.3 Training of the Models 59
3.3.1 Neural Network 59
3.3.2 Extreme Gradient Boosting 61
3.3.3 Random Forest 62
3.4 Predictions and Discussion 63
3.5 Optimization with Genetic Algorithm 71
4 CONCLUSION 77

4.1 Summary of the Key Findings 77

4.2 Suggestions for Future Works 79

REFERENCES 80

APPENDICES

A THE CODE USED IN THIS STUDY 93

LIST OF TABLES

TABLES

Table 2.1 Design parameters 28

Table 2.2 Properties of the Al 7075-T6 material which are used in the Johnson-Cook formulation, taken from [75]. 32

Table 2.3 FEA parameters used in the comparative study. 36

Table 2.4 Equations used to calculate the mechanical properties of lattices structures 43

Table 3.1 Hyperparameter optimization results 59

Table 3.2 Prediction results of the machine learning models 66

Table 3.3 Comparison of different approaches in terms of tasks and the estimated time required to perform them. 76

LIST OF FIGURES

FIGURES

Figure 1.1 Representation of the stress-strain curve of a lattice structure under compression load, adapted from ref. [15]. 3

Figure 1.2 Types of unit cells, adapted from ref. [21]. 5

Figure 1.3 Representative workflow of FEA, adapted from ref. [32] 6

Figure 1.4 Supervised learning: inputs are grouped into known classes, adapted from ref. [45]. 8

Figure 1.5 Unsupervised learning: inputs are grouped into clusters based on their similarity, adapted from ref. [45]. 9

Figure 1.6 Reinforcement learning: the agent perform actions and observe the environment state to maximize reward, adapted from ref. [45]. . . . 9

Figure 1.7 Illustration of various auxetic lattice structures, adapted from ref. [25, 49, 50]. 12

Figure 1.8 A general overview of ML applications in the additive manufacturing community, taken from ref. [62], where X represents the features and Y stands for the label or target value. 15

Figure 1.9 The ideal cell used for mechanical property calculations in the work of Ashby, adapted from ref. [15] 19

Figure 1.10 Unconstrained BCC unit-cell and deformation of its strut, adapted from ref. [72]. 20

Figure 1.11 Summary of the study 24

Figure 2.1 (a) Schematic description of the unit cell. (b) Node locations selected for the generation of the double-arrowhead lattices. Two instances are shown here. 27

Figure 2.2 Six examples of generated unit-cells 28

Figure 2.3 Representation of one lattice structure, (px=7.1, y=7.6, t=1.6) . . 29

Figure 2.4 Twelve examples of the generated lattice structures, different line widths represents different strut thicknesses. 30

Figure 2.5 Comparison of mesh size and depth for a lattice with parameters (px=10, py=5, t=1) 33

Figure 2.6 Load and boundary conditions applied to the models 34

Figure 2.7 Internal energy vs Kinetic energy comparison 35

Figure 2.8 Side by side view of (a) the experimental setup used in the work of Eren et al, adapted from ref. [9], (b) the virtual comparison model generated in this study 37

Figure 2.9 Force-displacement curves comparison with a literature work. . . 38

Figure 2.10 Force-displacement plot of 1401 samples 39

Figure 2.11 Deformation and result plots of one of the FE models, (px = 7.5, py = 7.1, t = 1.2) 40

Figure 2.12 Displacement vectors in lateral direction of a lattice structure (px = 7.5, py = 7.1, t = 1.2) for different strain levels. At the bottom, the contraction of the structure can be clearly seen, indicating auxetic or NPR behavior, with the gray transparent model representing the undeformed structure and the green model representing the deformed structure. 41

Figure 2.13 Ideal absorber properties, adapted from ref. [83]. 44

Figure 2.14 Trends in the data: (a) illustrates the relationship between thickness and the three output results, (b) depicts the trends between *px* and these outputs, and (c) highlights the correspondence between *py* and the same results. 45

Figure 2.15 Relation between four parameters: Plateau Stress, Energy Absorption, Energy Absorption Efficiency and *py*. 47

Figure 2.16 Correlation matrix 48

Figure 2.17 Pareto plots: a) Pareto set obtained in this work for the following objectives in the data: maximize specific useful work, minimize the yield stress, minimize mass of the structure, b) Pareto set from the work of Wang et al. [85] with the objectives of: maximize specific energy absorption, minimize the yield strength) 49

Figure 3.1 IQR box representation and outliers detection, adapted from ref. [87]. 52

Figure 3.2 Detecting and removing the outliers 53

Figure 3.3 Standardization of target values is illustrated, with the y-axis representing the density which is the smoothed approximation of the probability distribution of a dataset at a specific point. 54

Figure 3.4 The training and evaluation workflow includes cross-validation and hyperparameter tuning, adapted from ref. [88]. 56

Figure 3.5 Neural network representation used in this study, adapted from [94]. 60

Figure 3.6 XGBoost illustration, adapted from ref. [98]. 62

Figure 3.7 Random Forest illustration, adapted from ref. [98]. 63

Figure 3.8 Scatter plot of model predictions versus simulation results (ground truth). 64

Figure 3.9 Skewness in the data distribution of two outputs 65

Figure 3.10 Performance comparison of XGB with two traditional linear regression methods 67

Figure 3.11 Learning curves of XGB 69

Figure 3.12 Feature importances for different properties provided by XGB . . 70

Figure 3.13 Objectives history of GA and ML integration include maximizing energy absorption efficiency (n) and having plateau stress close to 40 MPa. Both values are predicted by using the trained ML model, XGB. 73

Figure 3.14 Optimum shape of the unit-cell given by GA and ML integration which is $px = 6.04$, $py = 8.66$, and $t = 1.63$. The blue color tones in the figure indicate the accumulation of design points at specific locations throughout the generations. 74

Figure 3.15 FEA model of the optimum result ($px = 6.04$, $py = 8.66$, $t = 1.63$) given by the GA and ML. 75

Figure 3.16 Plots of the same optimum structure ($px = 6.04$, $py = 8.66$, $t = 1.63$), (a) stress-strain plot (b) energy absorption efficiency plot . . . 75

LIST OF ABBREVIATIONS

ABBREVIATIONS	
AM	Additive Manufacturing
FEM	Finite Element Method
FEA	Finite Element Analysis
CAD	Computer-Aided Design
CAE	Computer-Aided Engineering
DAH	Double Arrow-Head
ML	Machine Learning
RF	Random Forest
XGB	Extreme Gradient Boosting
NN	Neural Networks
EDA	Exploratory Data Analysis
API	Application-Program Interface
SVM	Support Vector Machines
LSTM	Long-Short Term Memory
R^2	Correlation Coefficient
RMSE	Root Mean Squared Error
MAPE	Mean Absolute Percentage Error
GA	Genetic Algorithm

Nomenclature

$\epsilon_{densification}$	densification strain
$\rho_{effective}$	effective density
$\sigma_{densification}$	densification stress
$\sigma_{plateau}$	plateau stress
σ_{yield}	yield stress
EA	energy absorption
n	energy absorption efficiency
n	specific energy absorption
px	variable in two dimensional space
py	variable in two dimensional space
sea	specific energy absorption
Sr_{avg}	average slenderness ratio
svf	solid volume fraction
SW_{useful}	specific useful work
t	variable which represents thickness in milimeters
W_{useful}	useful work

CHAPTER 1

INTRODUCTION

Lattice structures are widely valued in both academia and industry due to their high specific stiffness, specific strength, low weight, and high energy absorption capacity. The advent of additive manufacturing has expanded the range of producible lattice structures, while state-of-the-art, data-driven approaches, such as machine learning, have enhanced the design process by facilitating pattern recognition and predictive capabilities for optimizing structural performance.

In this work, a total of 1401 double arrow-head lattice structures were generated using finite element analysis in an automated manner. The analysis results, along with the input features, were pre-processed through outlier removal and scaling before being used to train three different machine learning models: neural networks, random forest, and extreme gradient boosting. Before evaluating the performance of the models, the hyperparameters of all three machine learning estimators were optimized using 10-fold cross-validation. The prediction results of these optimized models for eight output targets are then presented, and optimization studies using the Pareto set and a genetic algorithm are conducted to identify the optimal design parameters for the structure. Finally, the predicted optimal structure for maximizing energy absorption efficiency, as determined by the genetic algorithm, is simulated, and this numerical result is compared with the machine prediction.

1.1 Background

Lattice structures and machine learning constitute the foundation of this study. By enabling more possible solutions and improving the design of lattice structures, the details of additive manufacturing are also worth mentioning. Moreover, since machine learning requires data for training and prediction, it is important to understand the data extraction method employed in this thesis through finite element analysis. This section provides insights into these fields by offering technical information on additive manufacturing, lattice structures, the finite element method, and machine learning.

1.1.1 Additive Manufacturing

Additive manufacturing (AM), also known as rapid prototyping (RP) or 3D printing, dates back to the 1980s and is a cutting-edge technology for producing complex parts [1]. In this manufacturing technique, parts are produced in a layer-by-layer fashion through the addition or deposition of material onto a base [2]. This method has brought about a significant transformation in many industries, including automotive, aerospace, and medical. Despite certain downsides, such as reduced mechanical properties and longer production times, it offers significant advantages, such as the ability to fabricate complex geometries, reduce waste, and customize components [3]. AM encompasses various methods, including stereolithography (SLA), fused deposition modeling (FDM), selective laser sintering (SLS), laminated engineered net shaping (LENS), electron beam melting (EBM), direct energy deposition (DED), laminated object manufacturing (LOM), and PolyJet [4, 5].

Although this work does not involve the physical production of parts using AM, and manufacturing technique is not considered as a parameter, it is essential to recognize AM as an enabling technology that has significantly enhanced the production of lattice structures. AM has expanded the possibilities of unit cell geometries, which were previously constrained by conventional methods, while also offering precise fabrication and high quality in lattice structure production [6].

1.1.2 Lattice Structures

A lattice structure is a composition formed by spatially periodic or repeated unit cells with their own geometrical features, such as faces and edges [7, 8]. These structures possess highly desirable properties, such as being lightweight [9], having energy absorption capability [10, 11], and exhibiting high specific strength and stiffness [12], as well as vibration damping [13]. Such properties make them attractive for use in aviation, bio-engineering, automation, and other industrial fields [14]. Additionally, advancements in additive manufacturing, a technique where parts are produced in a layer-by-layer fashion through the addition or deposition of material [2], have enabled the production of lattice structures that were not achievable with conventional manufacturing methods [7].

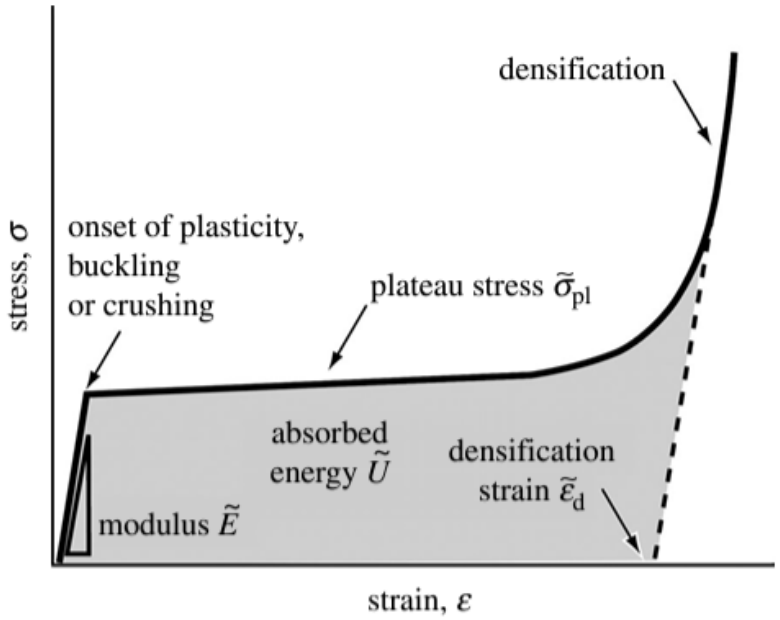


Figure 1.1: Representation of the stress-strain curve of a lattice structure under compression load, adapted from ref. [15].

The energy absorption characteristics of lattice structures have been extensively studied and have found applications in engineering [16, 17, 18]. These structures can absorb a significant amount of energy while undergoing large deformations under nearly constant stress, as shown in Figure 1.1, thereby resisting the transfer of high stress levels to the protected surface [19]. It can also be observed from the same figure that after a certain point, the structure becomes densified, behaves like a regular solid, and loses its significant energy absorption capability.

The energy absorption capacity of lattice structures is highly dependent on the geometry of their unit cells. The literature has explored various unit cell geometries, including conventional designs such as centered honeycomb [20], body-centered cubic (BCC), octahedron, and diamond face-centered cubic (FCC) [21], with some examples illustrated in Figure 1.2. In addition to these, auxetic unit cells have also been investigated, including designs such as the auxetic re-entrant circular [22, 23], auxetic re-entrant honeycomb (NREH) [24], star auxetic honeycomb [25], auxetic chiral [26], and auxetic double arrowhead [27, 28]. The negative Poisson's ratio (NPR) property of auxetic unit cells and lattice structures distinguishes them from conventional ones. For instance, under compression, auxetic materials tend to shrink rather than expand, and vice versa. This unique characteristic imparts resistance to indentation, increased stiffness, and superior energy absorption properties, making auxetic materials a widely preferred choice [27, 29, 30].

This study focused on using the double arrow-head unit cell to form a lattice structure. This unit cell, noted for its impact resistance and substantial energy absorption performance, is a strong contender [27, 28].

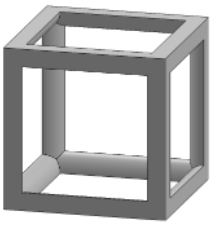
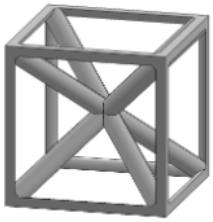
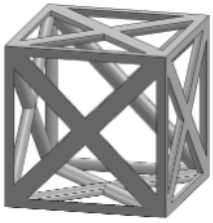
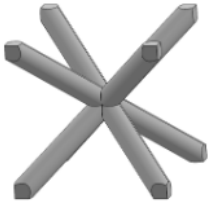
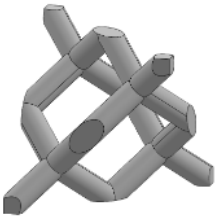
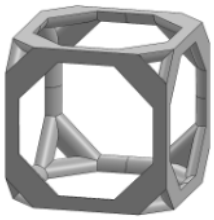
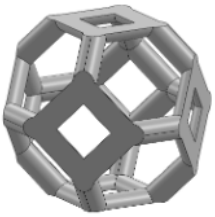
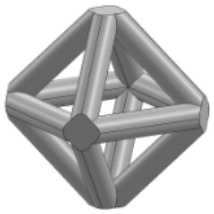
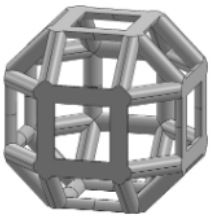
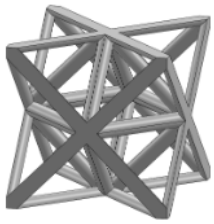
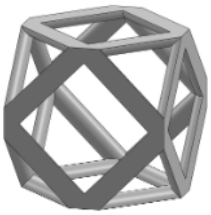
Unit Cell Topology	Image	Unit Cell Topology	Image
Simple cubic		Body-centered cubic	
Face-centered cubic		Body center	
Diamond		Truncated cube	
Truncated octahedron (a.k.a. Kelvin)		Octahedron	
Rhombicuboctahedron		Octet-cross (a.k.a. octet-truss)	
Cuboctahedron			

Figure 1.2: Types of unit cells, adapted from ref. [21].

1.1.3 Finite Element Method

The finite element method (FEM), also referred to as finite element analysis (FEA), is a numerical approach for approximating solutions to partial differential equations. This method involves discretizing the system into smaller, finite elements and imposing boundary conditions to analyze its behavior [31]. The term *discrete parts* refers to the division of a continuum body into relatively small elements with a finite number of degrees of freedom, which can then be represented as an algebraic system of equations. These systems of equations can be solved manually or with the assistance of a computer. Since most industrial applications involve a large number of unknown degrees of freedom, these problems are typically solved using computers. The typical process flow of a finite element analysis is illustrated in Figure 1.3.

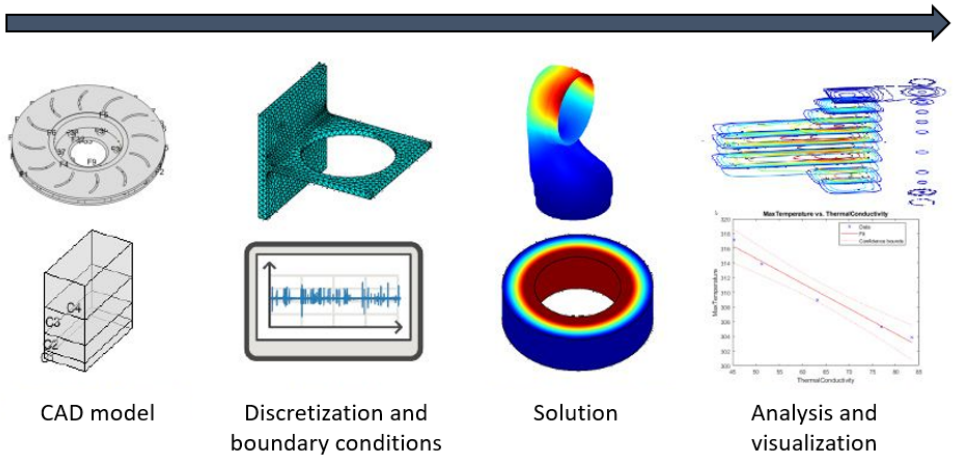


Figure 1.3: Representative workflow of FEA, adapted from ref. [32]

FEM is also part of a software group called CAE (Computer-Aided Engineering), where engineering problems are solved and simulated using the high computational capabilities of computers. The method can solve linear and nonlinear problems involving contact, large deformations, and elasto-plastic material behavior, and is widely used in industries such as aerospace, medical, and automotive [33].

FEA consists of three principal steps [34]:

- **Pre-processing:** In this step, the CAD model is discretized or meshed, and material properties as well as boundary conditions are applied.
- **Solution:** The pre-processed finite element model is solved in this step. Specifically, an *n by n* matrix, derived from the system of algebraic equations, is solved, and the desired results are obtained and stored.
- **Post-processing:** This final step involves visualizing, inspecting, and analyzing the results to draw conclusions.

All of the aforementioned steps can be performed using commercial software such as ABAQUS, NASTRAN, and ANSYS. Alternatively, these steps can also be completed manually through programming. In this work, ABAQUS 2018 with its explicit solver was utilized.

1.1.4 Machine Learning

Machine Learning (ML) is a highly prominent and widely utilized mathematical concept in contemporary times. ML lies at the intersection of computer science and statistics [35]. It is categorized as a branch of Artificial Intelligence (AI) [36, 37, 35]. ML has been applied across various domains, including image recognition, cybersecurity, agriculture, healthcare, natural language processing, manufacturing, education, and e-commerce [38, 39]. ML is defined as the study of algorithms and statistical models that computers use to perform specific tasks without being explicitly programmed [40, 36, 38]. It can also be described as the recognition of patterns in data or generalization from examples [41]. As a subset of AI, ML emphasizes replicating human cognitive processes, including learning and problem-solving [42].

In this context, "machine" refers to an algorithm that processes input into output based on specified calculations [43]. In the ML framework, the goal of this algorithm is to minimize or maximize a goal function, a process referred to as learning. It achieves this by identifying patterns in an abstract manner and then applying these patterns to solve related tasks. Generally, learning from patterns and applying them to associated

problems is similar to human behavior. An expert in a particular field may not need to perform explicit calculations each time they face a problem. Instead, they rely on patterns recognized from past experiences, which have been built through a learning process. This allows them to make decisions automatically and efficiently. Similarly, an ML model trains on historical data to identify patterns and make generalizations. As a result, it can make predictions without requiring extensive, time-consuming calculations.

There are mainly three ML approaches [44, 45]. However, it is important to note that this generalization excludes hybrid techniques, such as semi-supervised learning, due to their nature of combining more than one primary approach. The three main approaches are mentioned below:

- Supervised learning — In this approach, labeled data consisting of both inputs and corresponding outputs are fed into the machine. Figure 1.4 illustrates the concept of labeled data, showing how different shapes are grouped into distinct clusters based on their labels. Supervised learning is also noted as the most common form of ML [39]. It can be further categorized into two based on the objective: regression and classification. Regression aims to model the data, while classification seeks to categorize the data [38]. For example, recognizing images of cats and dogs is a classification task, whereas predicting the height of those animals is a regression task.

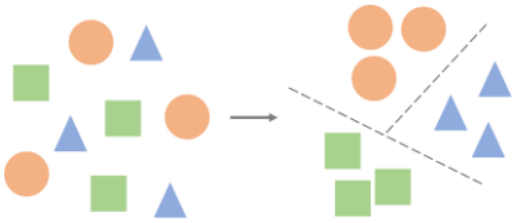


Figure 1.4: Supervised learning: inputs are grouped into known classes, adapted from ref. [45].

- Unsupervised learning — In contrast to supervised learning, this approach does not involve labels; instead, the model clusters the given data based on similar-

ities without prior knowledge of their identities, as illustrated in Figure 1.5, where grayed-out shapes are grouped together based on their proximity. For instance, given images of multiple animals without labels, the machine can group them into species based on similarities. Although the machine may not know the exact names of the species, it can identify that they share similar features and should therefore be classified together.

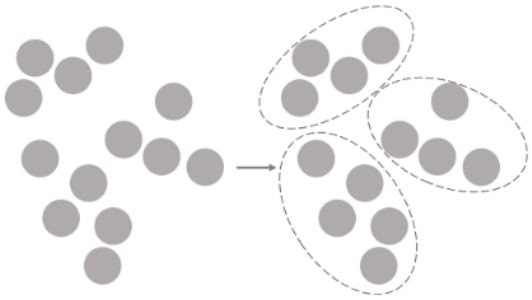


Figure 1.5: Unsupervised learning: inputs are grouped into clusters based on their similarity, adapted from ref. [45].

- Reinforcement learning — In this method, the machine (or agent) interacts with the environment and, based on an action-reward loop, tries to maximize or minimize an objective function over time. This process is demonstrated in Figure 1.6. This learning occurs in an unknown, dynamic, and interactive environment [35]. An example can be given as, an agent who is designed to play better game of pong than humans, by trying to maximize its score.

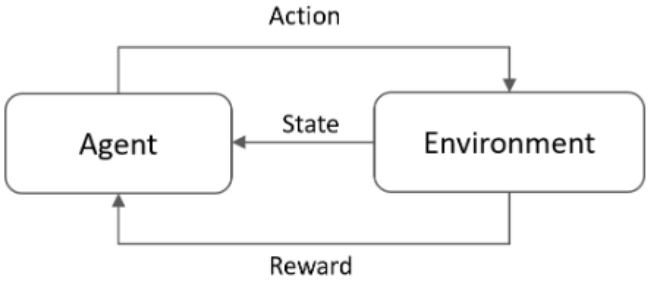


Figure 1.6: Reinforcement learning: the agent perform actions and observe the environment state to maximize reward, adapted from ref. [45].

This work employs ML to predict mechanical properties, specifically the energy absorption characteristics of auxetic double arrow-head (DAH) lattice structures. It enriches the literature on this subject and contributes to the essential application of ML in lattice structure problems. To achieve this, 1401 lattice structures with various unit cell configurations, based on their two arrow-head coordinates and strut thickness, were generated. These models were analyzed using FEA, producing raw data that was subsequently post-processed and used to train three ML models to identify and utilize complex trends in the data for predictions.

Prior to training, the data underwent pre-processing, which included trend observation, Pareto set extraction under specified conditions, outlier removal, and scaling. Three machine learning algorithms were then trained: Random Forest (RF), Extreme Gradient Boosting (XGB), and Artificial Neural Network (ANN), also referred to as Neural Network (NN). To further enhance performance, hyperparameter optimization was applied to each algorithm. The prediction results were compared to simulation results, and a Pareto set identifying the optimal design points within the given data was presented. Additionally, a genetic algorithm was employed to find optimal design points for a given objective beyond the training dataset. This approach demonstrated the successful prediction capability of the ML model when coupled with an optimizer.

1.2 Literature Review

This chapter explores the mechanical characteristics, with a particular focus on the energy absorption properties of auxetic DAH lattice structures from a literature perspective. It discusses common practices, challenges, and key considerations in designing, modeling, and analyzing such structures. The chapter then transitions to ML applications and reviews the contributions made by the research community and practitioners in related areas.

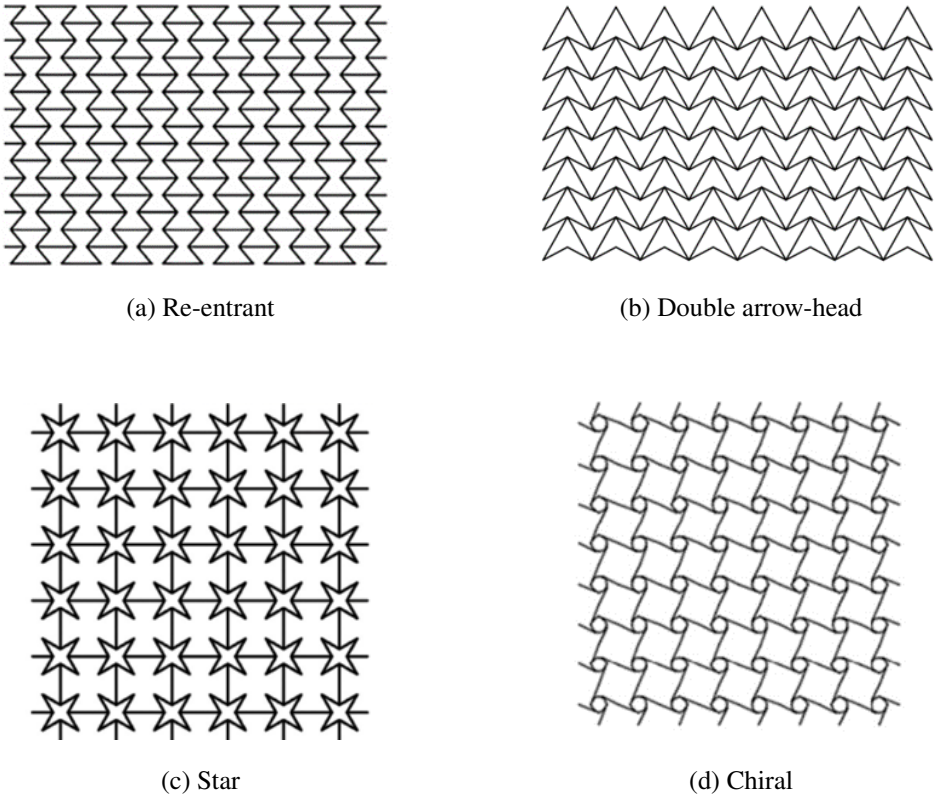
1.2.1 Double Arrowhead Auxetic Lattice Structures

Auxetic lattice structures offer researchers and industry the opportunity to use lighter, noise- and vibration-dampening, indentation-resistant, and efficient energy-absorbing materials [9, 10, 11, 13, 12]. They can be applied in various industries such as aerospace, automotive, biomechanics, and others [14]. One example of such usage is hail or bird strike protection [46] for aircraft radomes or leading edge regions, where auxetic lattice structures are used to provide superior impact resistance with minimal weight penalty.

As with classical honeycomb structures, which are a type of lattice structure, the deformation behavior under impact energy depends on the microstructure arrangement or topology of the lattice. By adjusting these parameters, unique properties can be achieved [47]. The auxetic property, characterized by a *negative Poisson's ratio*, is one such unique characteristic. It enables lattice structures to resist compression forces better by contracting under compression load and to expand under tension, which can be useful for other applications. Typical auxetic lattice structures include re-entrant structures [48, 49], chiral structures [50], star structures [25], and arrowhead structures [51] which can be seen in Figure 1.7.

This study focuses on the DAH unit cell for forming lattice structures. This type of unit cell, especially known for its impact resistance and notable energy absorption performance, can be an important contender [27, 28]. Eren et al. [9] investigated the effects of manufacturing and geometrical parameters on the performance of DAH lattice structures. They performed a parametric optimization using a design of experiments approach with FEA. From the design of experiments table they provided, it is seen that shallow lattices with thick struts and relatively wide angles yielded the highest specific energy absorption value. Another parametric study was conducted by Qiang et al. [52] on DAHs. They noted that the crashworthiness performance of such structures is more sensitive to the thickness of the long struts rather than the short ones. Additionally, they showed that a higher slenderness ratio lowers the specific energy absorption value. In another study, Chen et al. investigated the damping mechanism of 3D DAH structures [53]. They found that large deformations significantly increase the energy dissipation of DAH lattice structures by enabling greater elas-

tic buckling and macroscopic frictional energy consumption. Although these works have made valuable contributions to the literature, they generally lack a mathematical model to predict the mechanical properties of the structure due to the high deformations and the non-linear nature of the problem. The current study addresses this problem by using the latest data-driven problem-solving method in engineering and science, ML, to understand trends in the related data and predict the behavior of double arrow-head lattice structures.



(a) Re-entrant

(b) Double arrow-head

(c) Star

(d) Chiral

Figure 1.7: Illustration of various auxetic lattice structures, adapted from ref. [25, 49, 50].

1.2.2 Numerical Simulations of Lattice Structures

Numerous studies have explored the mechanical properties of lattice structures through both physical experimentation and numerical simulations employing the finite element method. While these studies may differ in their approaches such as using dif-

ferent materials or types of analysis to obtain information about various properties of the structures, they share a common goal: providing valuable insights into performing structural analyses or simulations.

In their study, X. Deng et al. [24] investigated the energy absorption characteristics of novel re-entrant honeycombs. They performed numerical simulations using the Abaqus/Explicit solver and employed two rigid plate models to conduct crushing tests with a friction coefficient of 0.2. They noted that, to perform a quasi-static experiment, the ratio of kinetic energy to internal energy should be kept around 5%. This indicates that the influence of inertial forces should be minimized as much as possible. Similarly, Q. Gao et al. [52] and his team conducted FEA in LS-DYNA to study the crashworthiness of double-arrowhead auxetic structures, using a friction coefficient of 0.2. They also used two rigid walls to load the structure in compression and constrained the bottom wall. Additionally, they introduced a mass scaling factor of 0.9 to speed up the process without exceeding the critical time step for a stable numerical solution and confirmed that the FE model was viable. In another study, Z. Chen et al. [54] examined the energy absorption and stiffness of negative Poisson's ratio lattice structures. They performed quasi-static compression analysis using LS-DYNA and included material non-linearity through the Johnson-Cook plasticity model, applying compression to a lattice structure with a rigid plate and a friction coefficient of 0.2. A similar study conducted by Smith et al. investigated the prediction capability of FEA under quasi-static compression loads on BCC (body-centered cubic) and BCC-Z (body-centered cubic with vertical strut) lattice structures [55]. They compared modeling such structures using 1-D beam and 3-D solid elements and noted the difficulty in accounting for beam element contacts in nodal regions. Additionally, they found that, with the same element type, the inability to define individual contacts between the struts resulted in the densification region of the stress-strain curve remaining unpredictable. They also reported a significant increase in computation time when using 3-D elements. Likewise, Cetin et al. studied the same BCC and BCC-Z type lattice structures as a filling material for thin-walled tubes and investigated their energy absorption characteristics under impact load through FEA [56]. They also noted the simplicity and reduction in computation time by using beam elements. However, they showed the comparison between numerical simulation and

FEA in which beam element model correlated relatively poor with the experiment compared to solid element model. They stated that, the reason behind is that self-contact mechanism can not be simulated for beam elements, thus densification region is not consistent with experimental data for this element type.

In this study, numerical simulation, specifically FEA, has been widely used. Techniques and common practices, such as element categories, load and boundary condition definitions, and analysis methods for building simulation models, have been adapted with application-specific modifications based on the literature and past experiences.

1.2.3 Related Machine Learning Studies

ML has found applications in many fields of science and engineering, additive manufacturing and therefore lattice structures are no exception [57, 58, 59, 60, 61]. This innovative approach introduces new methods for addressing problems in the field and opens up opportunities for further development. It surpasses empirical methods and statistical modeling in accuracy and its capacity to handle a large number of variables and their relationships. Figure 1.8, taken from the review by Meng et al., shows a broad overview of ML applications in additive manufacturing [62]. However, it should be noted that this is a rapidly developing field, and many new works are emerging over time. The learning types depicted in the figure are explained in the following pages of this work.

Research in this field is still ongoing, and researchers are using ML extensively. Dean Grierson et al. [63] highlight five key application domains for ML in AM: computer vision, prediction, semantic analysis, natural language processing (NLP), and information retrieval. Similarly, a review by S. S. Razvi et al. [59] outlines several tasks in additive manufacturing where ML can be utilized, such as build-time prediction, cost estimation, porosity prediction, wear strength prediction, and anomaly or defect detection. They also provide examples of algorithms used for these problems in the literature, including Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), and Neural Networks (NN) for porosity detection, among others. In another study, J. Zhang et al. Application-wise, Zhang [64] demonstrates the prediction of tensile

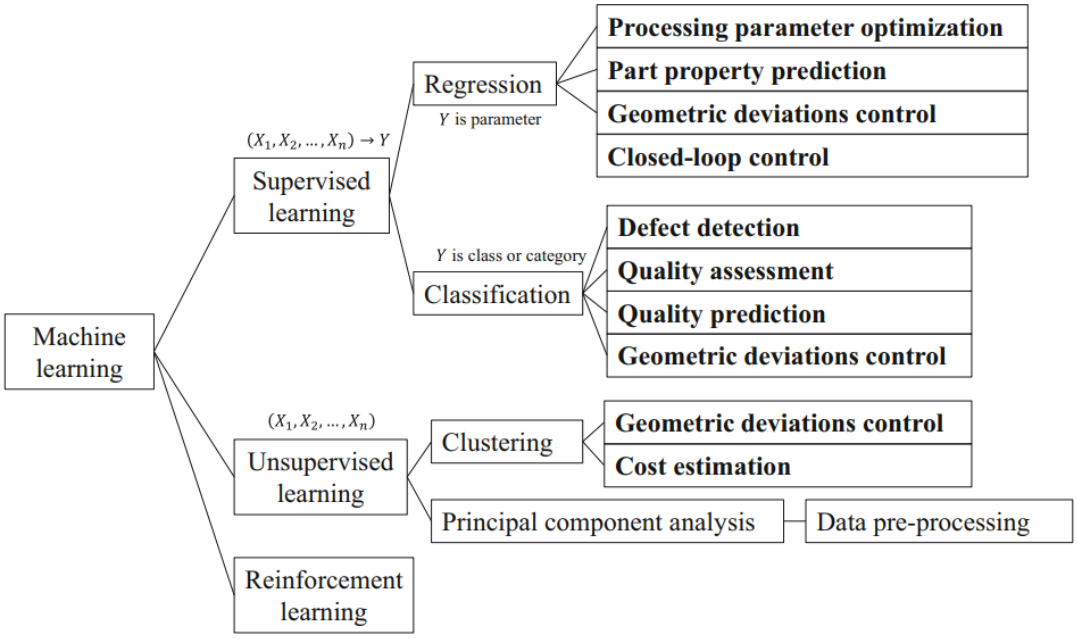


Figure 1.8: A general overview of ML applications in the additive manufacturing community, taken from ref. [62], where X represents the features and Y stands for the label or target value.

strength for basic tensile specimens produced by the FDM method using a specialized deep learning architecture known as Long Short-Term Memory (LSTM). They emphasize that the thermal history of the process is a crucial parameter, and by utilizing LSTM to account for this sequential or historical data, they were able to capture inter-layer interactions and enhance prediction accuracy. For features, they included material properties, extruder temperature, printing speed, layer height, and sequential data from sensors such as deposition temperature, ambient temperature, and printer vibration. As a result, they achieved an R^2 score of 0.899 and a Root Mean Squared Error (RMSE) of 0.58 on the testing data. A similar study by Manoharan et al. [65] predicted the tensile strength of FDM-produced PLA tensile test specimens. The features they used included layer thickness, infill density, printing speed, temperature, and build orientation, with a total of 33 samples. They compared two statistical modeling techniques with artificial neural networks (NN) and found that NN performed the best with a 1.10% error. Additionally, Kim et al. [66] employed a Gaussian Process, an ML algorithm, and trained it with data from 243 elastomer chi spring parts. They used features such as spring width, height, and truss thickness, and labeled them

with elastic stiffness, plateau stress, strain onset, and plateau width via compression tests. Their predictions, based on 97 parts, resulted in a 3% error. They also noted that traditional physics-driven design methods are unsuitable for chi-springs due to their high non-linearity from asymmetrical buckling and self-contact. In another study, Zhang et al. [64] utilized Long Short-Term Memory (LSTM) architecture to leverage sequential data. This type of neural network architecture uses gates to create an artificial memory, allowing it to make predictions about the current state based on previous data. They produced 144 tensile test samples using FDM and trained their LSTM model. They achieved a Root Mean Squared Error (RMSE) of 0.57 with various input parameters. A key finding from their work was that in-process data, such as vibration or layer temperature, can significantly improve the accuracy of ML models.

In the lattice structure research community, ML-driven approaches have also found their place. Reddy et al. examined the prediction of displacement under compression load, surface roughness, and micro-Vickers hardness [67]. For the combination of features such as lattice volume, surface roughness, hardness, post-processing conditions, lattice design (BCC, Honeycomb, Gyroid, etc.), and dimensions, their work achieved 99% accuracy in displacement prediction using the Random Forest regression algorithm. Importantly, they also noted that ML algorithms are more effective for modeling complex, non-linear relationships than classical statistical methods. In another study, Wu et al. predicted the energy absorption effects of different unit cells with R^2 correlation values of 0.9558 and 0.8990 for the training and testing datasets, respectively [68]. Their dataset consisted of 210 samples generated from 33 different unit cells through FEA. They were also able to apply this prediction to three new types of unit cells that were not included in the dataset, achieving a 13.1% relative error compared to experiments. This underscores the generalization capability of ML models across a wide range of problems within the trained domain. Sharing a similar goal to this thesis, Shuai et al. predicted the mechanical properties of lattice structures produced by 3D printing [69]. They used 57 samples with support vector regression (SVR), in other words, support vector machines with a regression task. They noted that predicting the mechanical properties of lattice structures in a fast and accurate manner is a challenge, which is one of the reasons behind the study presented in these papers as well. Aldair et al., in their work, aimed to accelerate lattice structure design

using ML-based approaches [70]. They were also able to observe the impact of each design variable on the result using Shapley additive explanation (SHAP) analysis. They assigned five different triply periodic minimal surface (TPMS) unit cell types and four geometric design parameters—height, width, depth, and thickness of the unit cell as features. They generated 270 samples and, through an FEA pipeline, trained a Gaussian Process (GP) model, reporting excellent prediction of elastic modulus with R^2 accuracy over 0.95. Additionally, they conducted an optimization process using Bayesian optimization to maximize elastic modulus. Moreover, their work importantly emphasized the use of simulations with ML approaches to optimize designs without losing comprehension of the impact of design parameters. Last but not least, Adithya et al. intended to increase the load-carrying capacity of sandwich structures that use lattice structures in their core by employing ML [71]. They built their dataset through FEA combined with manual programming. From a representative volume element (RVE), they generated different unit cells and assessed their buckling resistance under compression using FEA. They reported that ML helped them achieve a significant reduction in human effort and computation time for the task. Additionally, they noted that optimized lattice unit cells achieved 28–67% higher compression and 13–35% higher flexural strength compared to the octet lattice unit cell.

As is apparent, extensive research is being conducted within the AM and lattice structures community using ML techniques, generating the much-needed data through physical tests or numerical simulations, such as FEA. However, the amount of data required to develop well-performing machine learning models remains a significant challenge, especially for deep learning models that use deep neural network (DNN), which particularly benefit from large volumes of data. This may prompt the community to explore new methods or employ hybrid approaches, such as semi-supervised learning, which can at least reduce the volume of labeled data, if not the total amount. Additionally, new neural network architectures, such as transformers, may enhance the performance of existing strategies. Ultimately, ML revolves around two important concepts: data and the processing of data. Therefore, it is logical for related works to focus on these two aspects.

This work, by utilizing ML to predict mechanical properties, specifically the energy absorption characteristics of auxetic DAH lattice structures, enriches the literature on

this topic and contributes significantly to the application of ML in lattice structure problems. To achieve this, 1401 lattice structures were created with varying unit cell configurations based on their two arrow-head coordinates and strut thickness. These models were analyzed using finite element analysis, and the generated raw data was post-processed and input into ML models to leverage the complex trends in the data for predictions.

Three ML algorithms are employed for these predictions: RF, XGB, and NN. The results demonstrate that these ML algorithms effectively capture trends in the data and produce accurate predictions. Additionally, optimization studies were conducted using a Pareto set to identify optimal points within the given data under the applied constraints. A genetic algorithm was also utilized to discover optimal design points that are not samples from the dataset, highlighting the valuable generalization capability of the ML model.

1.3 Problem Statement

Predicting and understanding the behavior and properties of natural systems is one of the key driving forces in science and engineering. For example, engineers design bridges, airplanes, and other structures based on predictions that they will meet their intended objectives. However, acquiring such predictive knowledge relies on statistical, analytical, or numerical models, each of which has its own disadvantages.

Developing analytical formulas can be challenging; they often struggle to manage hundreds or thousands of variables and are frequently simplified representations of the actual phenomena being studied. In his well-known study, which has garnered over 1,800 citations, Ashby investigated lattice structures and proposed methods for calculating their mechanical properties [15]. His calculations were based on an idealized cell, shown in Figure 1.10, where L represents cell length and t denotes strut thickness. Ashby derived proportional relationships rather than exact equations, providing approximations as the result.

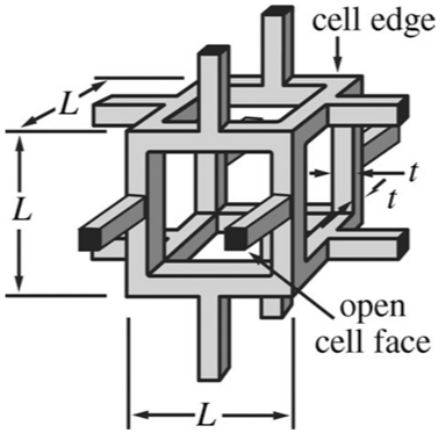


Figure 1.9: The ideal cell used for mechanical property calculations in the work of Ashby, adapted from ref. [15]

$$\left(\frac{\tilde{E}}{E_s}\right) \propto \left(\frac{\tilde{\rho}}{\rho_s}\right)^2 \tag{1.1}$$

Above equation, with constrained to *bending-dominated behaviour* as it is noted by Ashby is made proportional relation for modulus of elasticity (E), where E_s is the elasticity modulus of solid material, $\tilde{\rho}$ is density of the foam and ρ_s is density of the solid by him.

$$\left(\frac{\tilde{\sigma}}{\sigma_{y,s}}\right) \propto \left(\frac{\tilde{\rho}}{\rho_s}\right)^{3/2} \tag{1.2}$$

Another equation made by Ashby, with constrained to *bending-dominated behaviour* again, shows proportional relation for plateau stress ($\tilde{\sigma}$), where $\sigma_{y,s}$ is the yield strength of the solid, $\tilde{\rho}$ is density of the foam and ρ_s is density of the solid by him.

As can be inferred from these pseudo-equations, they establish proportional relationships between mechanical properties, which are applied to an idealized unit cell. It is also important to note that Ashby presents different equations for various loading conditions, such as *bending-dominated*, *stretch-dominated*, and *buckling-dominated* behavior. Therefore, their suitability for auxetic lattice structures subjected to varying loading conditions may be subject to scrutiny. Furthermore, and perhaps more importantly, these equations primarily provide insights into properties within the elastic

region. However, they do not account for properties exhibiting non-linear behavior, such as energy absorption. One possible reason for this omission is that properties influenced by the non-linear region may be difficult, or perhaps impossible, to quantify with high accuracy through analytical formulations.

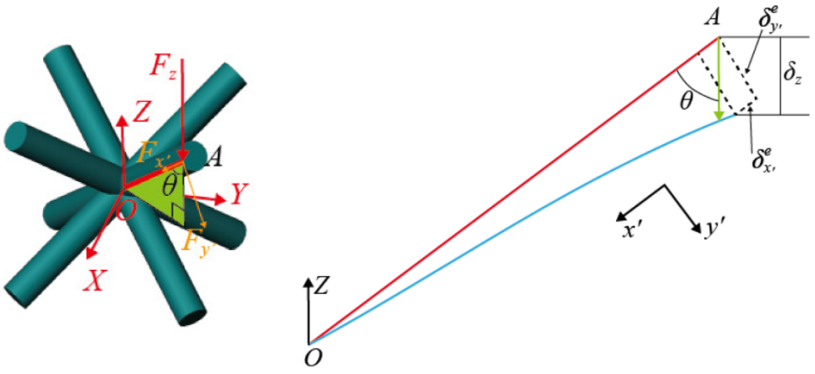


Figure 1.10: Unconstrained BCC unit-cell and deformation of its strut, adapted from ref. [72].

A similar observation is made in the study by Shang et al., where an analytical formulation is presented to predict the elastic modulus of a body-centered cubic (BCC) lattice structure [72]. However, they clearly state that their assumption relies on a linear change in the elastic modulus, highlighting the challenges in predicting non-linear properties. Their formulation for the elastic modulus of an unconstrained BCC lattice is shown in Equation 1.3, where $E^u e$ represents the elastic modulus of the unit cell in the Z-direction, k is the shear coefficient of the Timoshenko beam, νm is the Poisson's ratio of the material, and E_m is the elastic modulus of the material.

$$E_e^u = 9\sqrt{3}\pi^4 \frac{E_m}{12k^2\nu_m + 17k^2 + 2} \tag{1.3}$$

Lastly, regarding the prediction performance of analytical formulations for lattice structures, it is worth considering the work of Silva et al., in which they investigated several analytical models from different sources in the literature for truncated octahedron and cubic diamond lattices, comparing these models with compression test results of samples produced via additive manufacturing (AM) [73]. They noted

that geometry-based analytical models exhibited significant discrepancies from test results, with up to 98% average relative error for plateau stress prediction and 91% for elastic modulus. For the truncated octahedron, one analytical formula cited from another study performed with a 14% average relative error, the lowest reported error among the two different analytical approaches examined. Furthermore, Silva et al. also explored scaling law models based on experimental data, which demonstrated substantial improvement over the analytical models they reviewed. As one of their conclusion, they stated that the nonlinear, anisotropic behavior of the building material resulting from AM production led to a significant divergence from the assumptions underlying the analytical models.

As a conclusion it can be said that statistical models, derived from empirical data, may offer insights into the relationships between parameters. However, they often suffer from low accuracy, which is a crucial requirement. Numerical methods, such as FEM, also have drawbacks. They require extensive pre-processing, which consumes significant time and computational resources. Additionally, these numerical simulations lack generalization, meaning that each problem must be solved individually to achieve accurate results. Therefore, it is logical to seek an approach for predicting the mechanical behavior of lattice structures that is fast, efficient, easy to implement, and accurate. With the increasing computational power and the growing volume of data due to advancements in hardware and digitalization in recent years, ML offers a promising solution. By recognizing patterns in the data, ML can handle billions of variables, generalize across a wide range of similar problems, and provide results that are faster, easier to develop, and often more accurate compared to traditional techniques such as statistical modeling.

1.4 Objectives

The objective of this study is to build a ML model that can be trained with data generated through FEA and subsequently make predictions about the mechanical properties of double-arrowhead lattice structures, particularly their energy absorption characteristics. The model should be robust, fast, and efficient, providing answers about the energy absorption performance of the structure with an acceptable margin of error.

Additionally, the ML model should be compatible with a genetic algorithm to facilitate information exchange and assist users in finding optimal design parameters for a given property within seconds.

To achieve this objective, this work follows the below procedure:

1. Construct a unit cell in the FEA interface using three variables: px and py, which represent spatial coordinate points, and t, which denotes the thickness of the unit cell and, consequently, the lattice structure, as explained in the following pages.
2. Replicate this unit cell to create a structure of 6 cells in height and 8 cells in width. During this process, apply symmetrical features to avoid eccentricity and secondary bending while respecting the given constraints.
3. Apply the necessary boundary conditions to the lattice structure and perform the analysis as a quasi-static type. Save the results in a file along with the corresponding input variables
4. Perform mathematical operations using the raw output data to obtain the mechanical properties of the structure.
5. Automate the steps outlined above using the Python-ABAQUS API to generate the required number of samples.
6. Conduct exploratory data analysis (EDA) on the inputs and outputs of all samples, interpret the results, and present a showcase of the Pareto set.
7. Pre-process the data as needed, then feed it into various ML models. Train the models and compare their results.
8. Select the most effective ML model and perform optimization by integrating ML with a genetic algorithm.

1.5 Scope of the Study

The ML methodology presented in this study can be used to predict the mechanical properties of arrowhead lattice structures with a proper amount of training data. Since one of the hardest problems in training a ML model is acquiring data, this study also demonstrates how such data can be generated artificially through structural simulations. Once the model is trained and deployed, it can be used to predict and provide information about the desired mechanical properties of an arrowhead lattice structure in seconds, almost effortlessly.

For this task, the data generated through finite element analysis is converted into valuable information regarding the mechanical properties of the lattice structures under investigation. Pre-processing techniques applied to this data include outlier removal and standardization. Following these steps, the data is divided into two subsets: a training set and a test set. The test set is reserved until the evaluation phase of the models. The three ML models, RF, XGB, and NN, are trained using the training dataset, with hyperparameter optimization applied to each. Their performance is then evaluated using the test dataset. The results demonstrate that the ML algorithms effectively captured trends in the data and produced accurate predictions. Additionally, a Pareto set was employed to identify optimal points within the dataset under the applied constraints. After selecting the most effective ML model, a genetic algorithm was used to find optimal design points for the structure by exploring design parameters not present in the dataset, thereby showcasing the valuable generalization capability of the ML model. A representative summary of the study is illustrated in Figure 1.11.

An important aspect that lies beyond the scope of this study is the fracture or rupture of lattice structures under loading. This study disregards such damage behaviors and assumes that the structure under investigation behaves in an elasto-plastic manner, without any occurrence of damage.

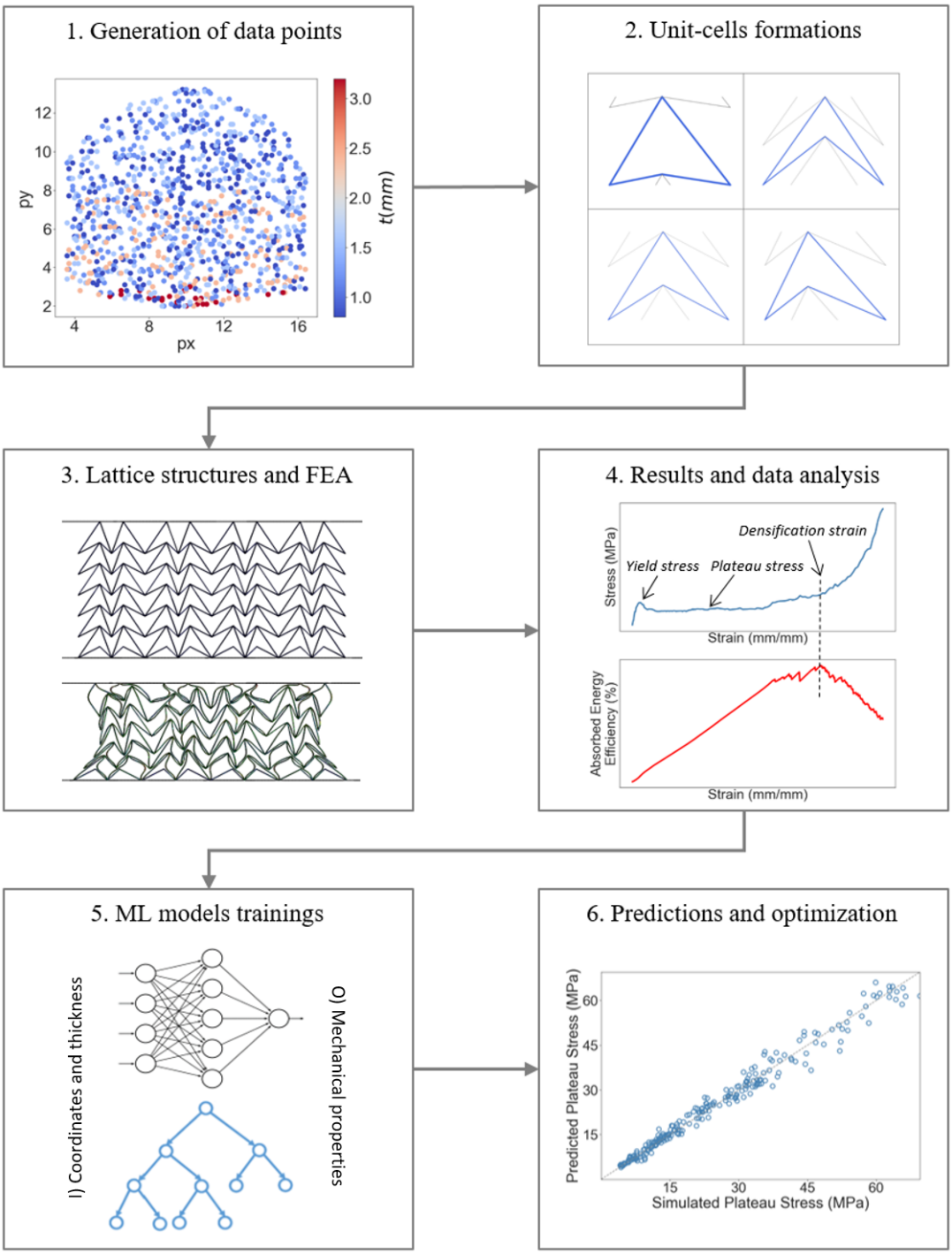


Figure 1.11: Summary of the study

CHAPTER 2

DESIGN AND FINITE ELEMENT MODELING

This chapter examines the design and analysis of DAH lattice structures. The discussion begins with the formation of the unit cell, followed by an explanation of how this unit cell is replicated to construct the overall lattice structure. FEA is then performed on this lattice structure to extract valuable information about its structural behavior. The resulting data is systematically organized to facilitate pre-processing and training of the machine learning models. The design and analysis process is executed within the ABAQUS interface. Rather than manual execution, the process is automated using Python programming language. This automation involves defining points, lines, materials, loads, boundary conditions, and other necessary entities, as well as extracting force-displacement data from the finite element solution. The pseudo-code for the algorithm responsible for this process is presented in Algorithm 1.

2.1 Geometry of the Unit Cell

The unit cell serves as the foundational building block of the lattice structure. While various types of unit cells exist, such as honeycomb, chiral, and re-entrant honeycomb [20, 22, 23, 24, 25, 26], this study focuses specifically on the double arrow-head (DAH) configuration. Figure 2.1a illustrates the construction of a lattice structure by depicting four spatial points (P1, P2, P3, P4). Points P1 and P2 are designated as fixed, whereas P3 and P4 are variable, with P3 being dependent on P4. The Y-coordinate of P3 is fixed at 20 mm, while its X-coordinate is equal to that of P4. This alignment ensures that there is no eccentricity between the two arrow-head points and improves structural integrity. Figure 2.1b demonstrates the generation of unit cells for

Algorithm 1 Data generation

1: **begin**
2: *number of models* $\leftarrow$ 1401
3: *count* $\leftarrow$ 0
4: *thicknesses* $\leftarrow$ [0.8, 1.2, 1.6, 2.4, 3.2]
5: **while** *count* $<$ *number of models* **do**
6: $px \leftarrow$ random between $[0, 20]$
7: $py \leftarrow$ random between $[0, 20]$
8: $t \leftarrow$ random choice from *thicknesses*
9: **if** (px, py) violates geometric constraints **then**
10: **go back** to line 6
11: **else**
12: **generate** a lattice structure
13: **perform** FEA on the lattice structure
14: **extract** force-displacement values and store them
15: *count* $\leftarrow$ *count* + 1
16: **end if**
17: **end while**
18: **end**

different geometries within a 20 mm by 20 mm space around these four points. As P3-X is identical to P4-X, these two variables are defined as *px*, with P4-Y defined as *py*. Additionally, the uniform thickness of the lattice structure is set as another independent variable, denoted as *t*.

Figure 2.1b further highlights that although the point constraints range from 0 to 20, the distribution never reaches these extremes due to applied constraints. These constraints are crucial for maintaining structural integrity by preventing eccentricity or secondary bending effects. Any violation of these constraints results in the termination of the unit cell formation process, prompting the selection of a new set of variables to restart the process.

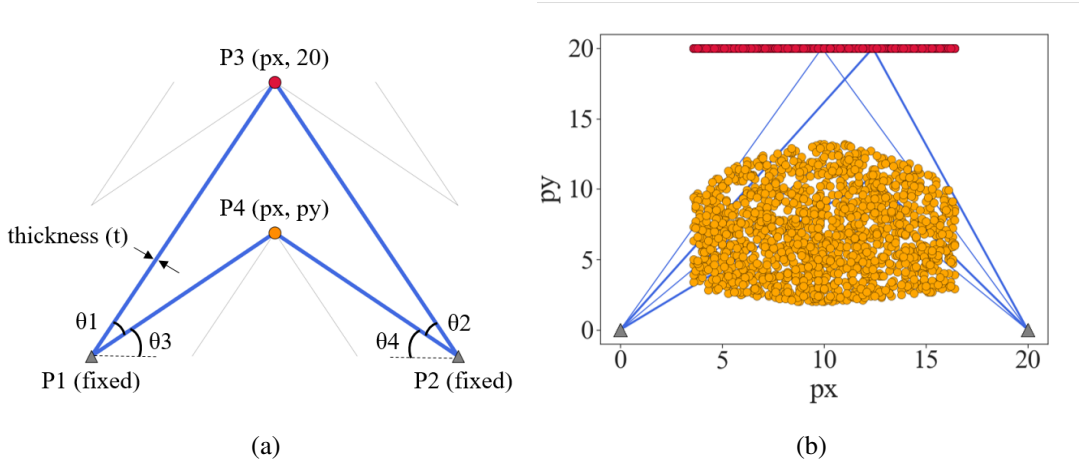


Figure 2.1: (a) Schematic description of the unit cell. (b) Node locations selected for the generation of the double-arrowhead lattices. Two instances are shown here.

The applied constraints are listed below:

- $\Theta 1 + \Theta 3 < 80^o$ and $\Theta 2 + \Theta 4 < 80^o$
- $\Theta 1 \geq 10^o$ and $\Theta 3 \geq 10^o$
- $\Theta 3 > 10^o$ and $\Theta 4 > 10^o$
- Solid volume fraction < 0.8
- (px, py) must be unique.

A brief summary of the design parameters is provided in Table 2.1. These parameters are matched with the results to establish a supervised learning environment for the machine learning task in the subsequent steps of this work. The parameters *px*, *py*, and *t*, as shown in the table, represent the point coordinate variables and thickness in millimeters, respectively, as previously mentioned. It is worth mentioning that, not including too many features can be a beneficial by reducing the complexity of ML model. Adding more variables increases the model's dimen- sionality, which in turn exponentially increases the amount of data required to main- tain model accuracy. To keep the required training data manageable, it is beneficial to limit the number of variables [41].

Table 2.1: Design parameters

Design Parameter	**Description**
px	coordinate point of point 3 and 4 in the width direction
py	coordinate point of point 4 in the height direction
t	thickness (mm) of all struts used in building the unit-cell

The procedure described above resulted in the creation of 1401 DAH unit cells, which were subsequently transformed into lattice structures. Six examples from this dataset are depicted in Figure 2.2, where line widths denote the strut thickness.

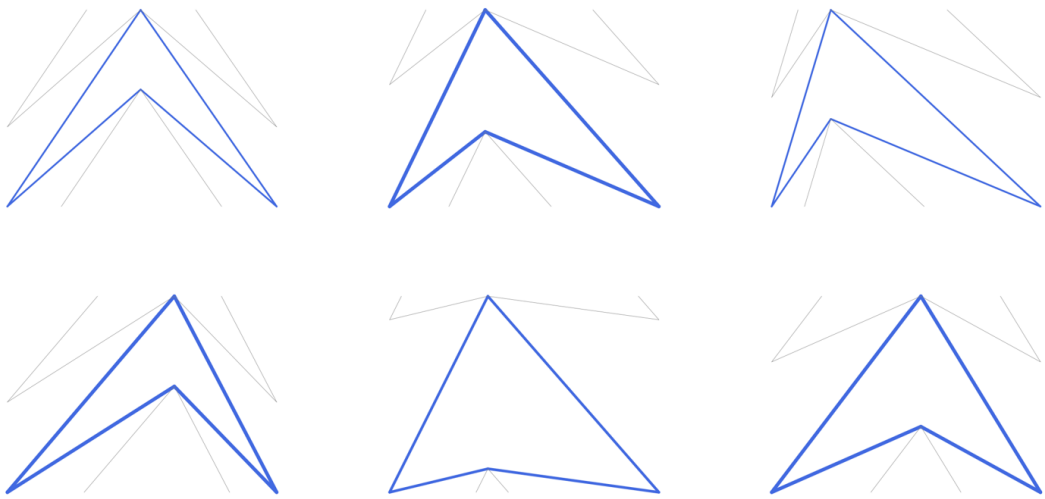


Figure 2.2: Six examples of generated unit-cells

2.2 Formation of Lattice Structures

The lattice structure is the primary focus of this investigation, with its properties being utilized in this work. It is generated through the repetitive arrangement of unit cells. Once the unit-cell geometry is established, it can be replicated within the in-plane spatial space. The unit cell is multiplied eight times along the width (x-direction) and six times along the height, resulting in 6×8 (height $\times$ width) lattice structures for various unit cells. The thickness of the struts or beams is a critical parameter in this study, selected randomly from a set of values for each lattice structure and applied

uniformly throughout the structure. Additionally, the lattice structures are designed to minimize eccentricity and secondary bending effects by incorporating a symmetry line between consecutive unit cells. This approach is illustrated in Figure 2.3, where the symmetry line is also visible.

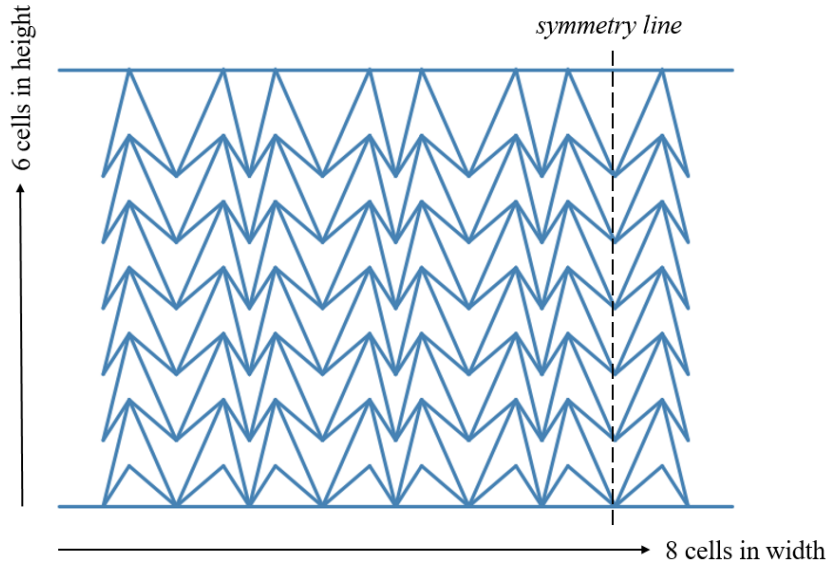


Figure 2.3: Representation of one lattice structure, (px=7.1, y=7.6, t=1.6)

Twelve examples of lattice structures, created using the aforementioned procedure, are provided in Figure 2.4, with line widths adjusted according to their respective thicknesses.

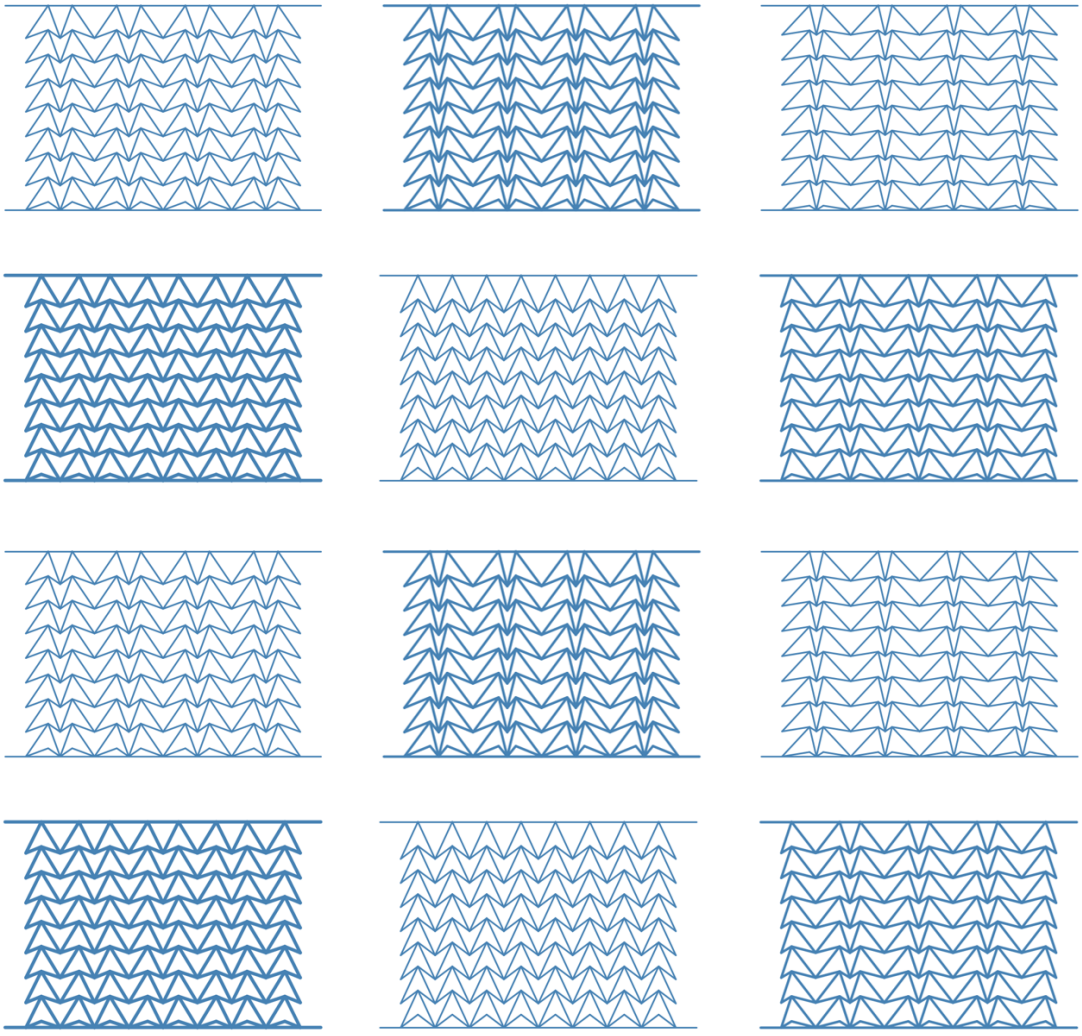


Figure 2.4: Twelve examples of the generated lattice structures, different line widths represents different strut thicknesses.

2.3 Finite Element Analysis

FEA is a widely used numerical modeling technique in both academia and industry. It is especially useful for solving structural problems that are particularly challenging to address with manual calculations. Its primary purpose is to discretize a continuous space, whether a structure or a fluid, and solve for each discretized point under given boundary conditions. In this work, FEA is extensively utilized for each lattice structure, with structural simulations performed to acquire raw data. ABAQUS com-

mercial finite element package, specifically its Explicit solver, has been employed for this task.

2.3.1 Material Selection

Accurately capturing material behavior is a crucial step in finite element analysis (FEA). In this study, the Al-7075 alloy was selected due to its ductility and the availability of data for Johnson-Cook plasticity modeling [54], as detailed in Table 2.2.

In this hardening model, static yield stress σ^0 has the following form [74],

$$\sigma^0 = \left[A + B(\varepsilon^{pl})^n\right]\left(1 - \hat{\theta}^m\right) \tag{2.1}$$

where ϵ^{pl} is equivalent plastic strain, A is initial yield stress, B is hardening constant, n is hardening exponent and m is thermal softening exponent [75]. These are the material parameters at or below transition temperature [74]. $\hat{\theta}$ is the non-dimensional temperature which is 273 K as default value in ABAQUS and this makes the result of the below equation 0,

$$\hat{\theta} \equiv \begin{cases} 0 & \text{for } \theta < \theta_{\text{transition}} \\ (\theta - \theta_{\text{transition}})/(\theta_{\text{melt}} - \theta_{\text{transition}}) & \text{for } \theta_{\text{transition}} \leq \theta \leq \theta_{\text{melt}} \\ 1 & \text{for } \theta > \theta_{\text{melt}} \end{cases} \tag{2.2}$$

As can be seen from Equation 2.1, the Johnson-Cook hardening model does not necessarily require the inclusion of strain rate. A strain-rate-independent variant of the model is also valid and was employed in this study to reduce computation time by simplifying the analysis and to prevent potential convergence issues during the data generation process. This approach has also been observed in the literature [76, 52]. However, it should be noted that such neglect can result in higher elastic modulus values and plateau stresses compared to reality. Additionally, the omission of such a variable constrains ML models from capturing different patterns arising from strain rate changes.

Table 2.2: Properties of the Al 7075-T6 material which are used in the Johnson-Cook formulation, taken from [75].

Parameter	Unit	Value
Density	kg/mm^3	2.7×10^{-6}
Young's Modulus	GPa	72
Poisson's Ratio	GPa	0.32
A	-	520
B	-	477
n	-	0.52
m	-	1
θ_{melt}	$Kelvin$	893
$\theta_{transition}$	$Kelvin$	403

It is important to note that the same material model was maintained for all lattice structures generated in this investigation. However, although this base material is consistent, the overall mechanical behavior of the lattice structures, such as elastic stiffness, can differ, as presented later. While some literature utilizes an elastic-perfectly plastic model [24, 77, 23, 78], a plastic material model incorporating strain hardening provides a closer approximation to physical reality. Additionally, since the material is an aluminum alloy, the lattice structures can also be produced using metal additive manufacturing (MAM), an advanced manufacturing technique [79].

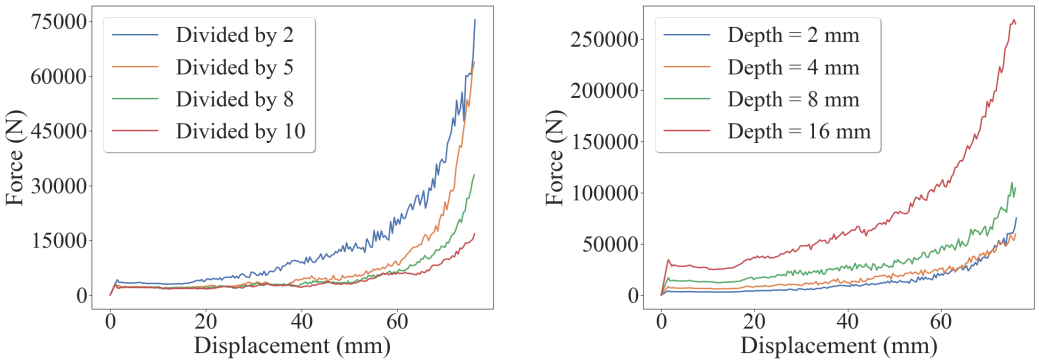
2.3.2 Discretization of the Structures

Similar to material selection, the proper element size is critical for accurately capturing the behavior of the modeled structure. In this study, the lattice structure is discretized, or meshed, with an element size determined by dividing the minimum strut length by 5. As each lattice structure has different geometrical dimensions, the element size varies accordingly. This dynamic scaling of the mesh size is designed to capture adequate buckling behavior while minimizing computational power and time requirements. It is known that, an excessively coarse mesh can result in excessive stiffness and misrepresentation of the structural behavior, whereas an overly fine

mesh can lead to increased computational time.

For instance, Figure 2.5a illustrates a lattice structure with $(px = 10, py = 5, t = 1)$, where the strut length is divided by factors of 5, 8, and 10. The results reveal similar force-displacement curves, leading to the selection of 5 as the optimal mesh size for effectively capturing structural behavior while reducing computation time.

Additionally, 1-D beam elements are inadequate for accurately representing the contact behavior of lattice struts [55, 56] whereas 3-D elements has relatively high cost of computation which is an important aspect of this study. Moreover, 3-D elements also introduce the difficulty names as *shear locking* in which, inadequate element number of elements through thickness may introduce artificial shear stress and stiffness. Therefore, to address these challenges, this study utilizes 2-D shell elements with four nodes and a reduced integration formulation (S4R) to balance computational efficiency, mitigate shear locking effects and capture frictional dissipations.



(a) Strut length divided by different numbers. (b) Lattice structure extruded with different depths.

Figure 2.5: Comparison of mesh size and depth for a lattice with parameters (px=10, py=5, t=1)

2.3.3 Boundary Conditions

Boundary conditions are applied to the top and bottom rigid plate components [80, 24]. The top plate is displaced downward to deform the initial height by 80% [52], as

described in Equation 2.3, where V represents velocity, $h_{lattice}$ denotes lattice structure height, and t_{total} is the total simulation time of 0.1 seconds. The bottom plate is fixed in all spatial degrees of freedom (Figure 2.6). Additionally, all nodes are constrained in the out-of-plane direction to impose a plane-strain condition on the model [81], which significantly reduces computational time.

To evaluate the effect of the plane-strain condition on the structure's depth, the same model with parameters px = 10, py = 5, and t = 1 is analyzed with depths of 2, 4, 8, and 16 mm, as shown in Figure 2.5b. The force-displacement plots converge at lower depth values; thus, a depth of 4 mm is selected for the analysis.

$$V = \frac{h_{lattice} \times 0.8}{t_{total}} \tag{2.3}$$

Lastly, contact interactions are incorporated into all models with a friction coefficient of 0.2 [54, 24, 47]. This friction value ensures the symmetry of the stiffness matrix, thus contributing to reduced computation time [82]. By including these interactions, all models address three types of non-linearity: elasto-plastic material behavior, contact interaction, and geometrical non-linearity, which accounts for changes in geometrical stiffness due to large deformations. Although these considerations increase computational demands, they enable a more accurate representation of structural behavior, as observed in real-life scenarios.

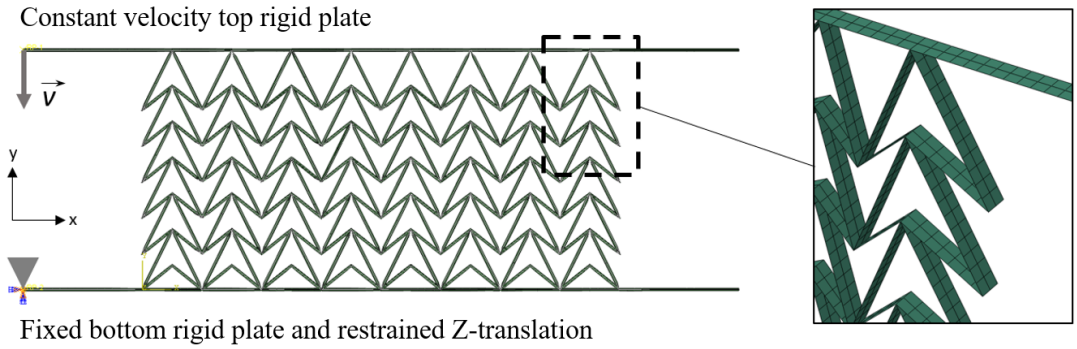


Figure 2.6: Load and boundary conditions applied to the models

2.3.4 Analysis Type

Finally, for all lattice structures considered in this study, the *Dynamic/Explicit* type of analysis was selected within the ABAQUS environment, with the compression load applied as a velocity, as previously described. Considering the extensive amount of data to be generated, this type of analysis was deemed appropriate due to its ability to reduce computation time. Additionally, it was ensured that the analysis remained within the quasi-static region, where inertial effects are negligible, a common approach for analyzing the energy absorption characteristics of lattice structures [23, 24, 52, 54, 55]. To satisfy this requirement, the ratio of kinetic energy to internal energy was kept below 5% [80]. As demonstrated in Figure 2.7, this condition was met, with kinetic energy remaining well below internal energy, thereby confirming the quasi-static nature of the analysis.

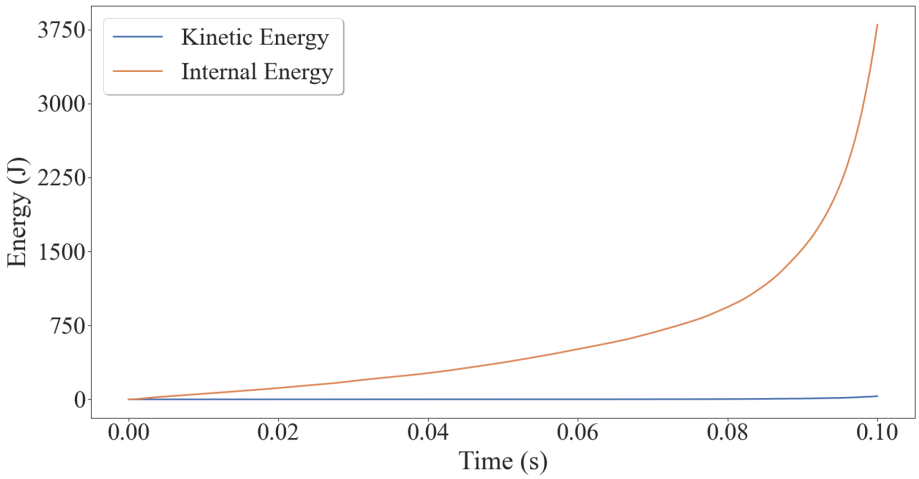


Figure 2.7: Internal energy vs Kinetic energy comparison

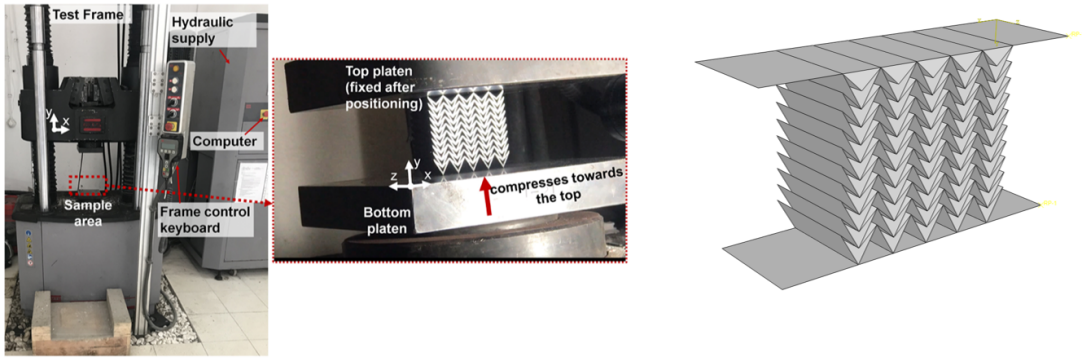
2.3.5 Comparison with a Literature Work

Although the core idea of this study is to demonstrate a methodology for obtaining data through finite element analysis, training a machine learning model, and performing optimization with the trained model, without claiming exact quantitative results regarding the DAH structure, it is still beneficial to verify the logical soundness of the numerical models used. To achieve this, the experimental results of Eren et al. [9] was

used as the reference for comparison. First, the finite element model from the work of Eren et al. was replicated in the ABAQUS environment, with deviations noted in Table 2.3 and this model made the *comparison model*. It can be seen as a model between the reference model and final FEA models which is labeled as *final model* in the table to carry the differences with minimal impact and have a sound comparison. As it can be seen from the table that the major difference is taking account of damage model which means that after a certain point, some elements which satisfies the damage model starts to carry less load and eventually gets deleted. This is not modeled in this study, and therefore, it is not included in the comparison model. Three parameters in the table which are not provided in the work of Eren et al. which are analysis time, element size and friction coefficient. These are taken as the decided values for this study which are outlined earlier and minimum strut length of the unit-cell divided by five for the element size as it is the case for the all models that are planned to be generated. After achieving logically consistent results in the comparison, the study proceeded with final adjustments to model options for data generation through FEA. The two main adjustment are applying plane strain condition and changing material from Ti-6Al-4V alloy to Al-7075 alloy.

FEA option	**Eren et al. [9]**	**Comparison Model**	**Final Model**
Material	Ti-6Al-4V	Ti-6Al-4V	Al-7075
Plasticity Model	Johnson-Cook	Johnson-Cook	Johnson-Cook
Strain Rate	Yes	No	No
Damage Model	Yes	No	No
Strut Thickness	1 mm	1 mm	0.8 mm to 3.2 mm
Element Type	3-D	2-D	2-D
Analysis Type	Quasi-static	Quasi-static	Quasi-static
Element Size	Not provided	$\min(L_{strut})/5$	$\min(L_{strut})/5$
Analysis Time	Not provided	0.1 s	0.1 s
Friction Coefficient	Not provided	0.2	0.2

Table 2.3: FEA parameters used in the comparative study.



(a) (b)

Figure 2.8: Side by side view of (a) the experimental setup used in the work of Eren et al, adapted from ref. [9], (b) the virtual comparison model generated in this study

The result of the comparison can be seen in Figure 2.9, which shows the force-displacement plots of both the comparison model and the experimental results from the work of Eren et al. The plot displays the approximate average load-displacement behavior under compressive loading for five samples built at a 0-degree orientation, meaning they were produced parallel to the X-Y powder bed plane using an additive manufacturing (AM) method, with no relative angle. In the figure, the linear regions in both models are similar. For instance, in Figure 2.9, the plot representing Eren et al.'s work reaches 114 kN before dropping at around 2.1 mm displacement, while the comparison model shows a load of 123.2 kN at the same displacement, resulting in an error of 8.8%. The difference can be attributed to strain rate effects, which were omitted in the comparison model, as mentioned earlier, and may have resulted in a slightly steeper linear curve. Another factor contributing to this difference could be manufacturing defects present in the experimental model, which are not accounted for in the comparison model. Furthermore, the comparison model, which employs a plasticity approach without damage parameters, displays a logical yet distinct response compared to Eren et al.'s experimental results. In Eren et al.'s reference model, fluctuations occur as the structure sustains damage and undergoes load redistribution, effects that the comparison model does not capture. This difference arises because this study adopts a plastic modeling approach without damage parameters, a common method in the literature [26, 51, 54]. Consequently, this modeling difference contributes to variations in the non-linear region between the virtual and physical environments. Since

both models show damage initiation and plateau behavior beginning at approximately 120 kN, it can be inferred that if the reference model excluded damage and exhibited strain hardening, its behavior would align more closely with the comparison model, displaying a more stable plateau without fluctuations.

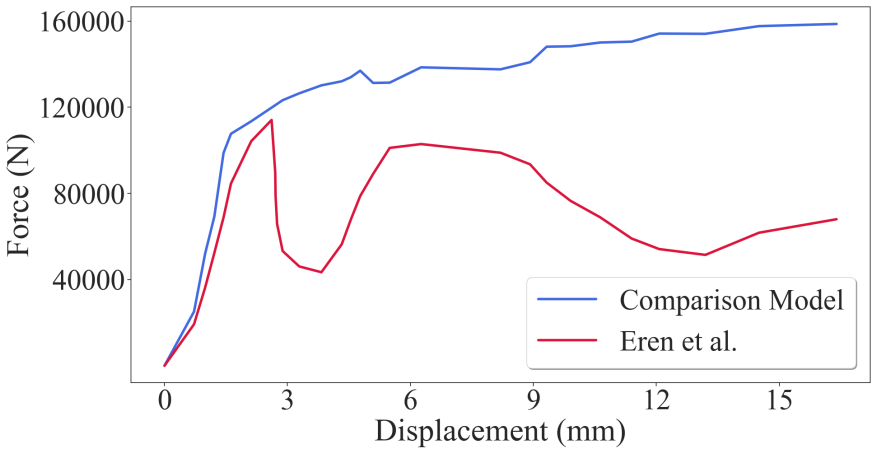


Figure 2.9: Force-displacement curves comparison with a literature work.

2.4 Analysis Results and Discussion

With the aforementioned geometrical definitions, material selection, and application of boundary conditions and loads, results were obtained through computer simulations. It took approximately 303 hours to solve 1401 models and extract the data. The models were run on a hardware with 32 GB of RAM and 6 to 12 cores, including hyper-threads, depending on the processor temperature. Results were extracted as force-displacement data from the analysis steps and saved as output files. The data in these files were then plotted, and additional properties were calculated. Figure 2.10 illustrates the force-displacement plot for all the data. Although the trends in the plots are similar, variations in the yield point, densification threshold, and area under the curve are evident.

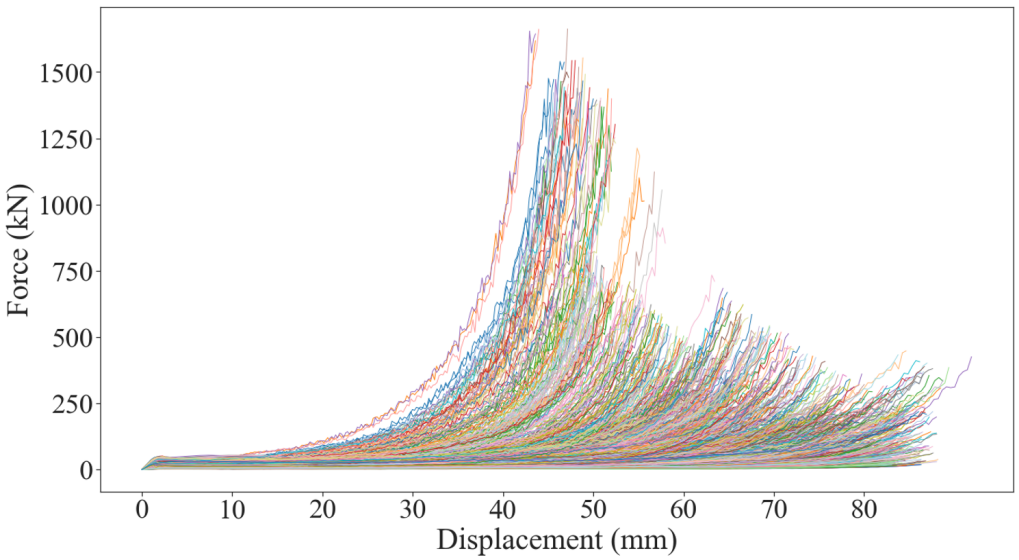


Figure 2.10: Force-displacement plot of 1401 samples

Apart from analyzing the overall plots, the behavior of individual lattice structures can also be extracted and processed from the data. For example, Figure 2.11 illustrates the stress-strain and absorbed energy efficiency plots for a model with parameters $px = 7.5$, $py = 7.1$, and $t = 1.2$. This figure also displays deformation frames and stress fields at various strain levels. Notably, the figure shows that the model undergoes quasi-static deformation, with the bottom portion of the model deforming concurrently with the top portion, indicating no delay in the stress field due to the highly explicit behavior. Additionally, the absorbed energy efficiency peaks around a strain value of 0.55, marking the densification strain and corresponding densification stress. The auxetic behavior of the DAH lattice structure, characterized by its tendency to shrink under compression and its classification as a NPR (Negative Poisson's Ratio) structure, is apparent from the figure, particularly between strains of 0.2 and 0.4.

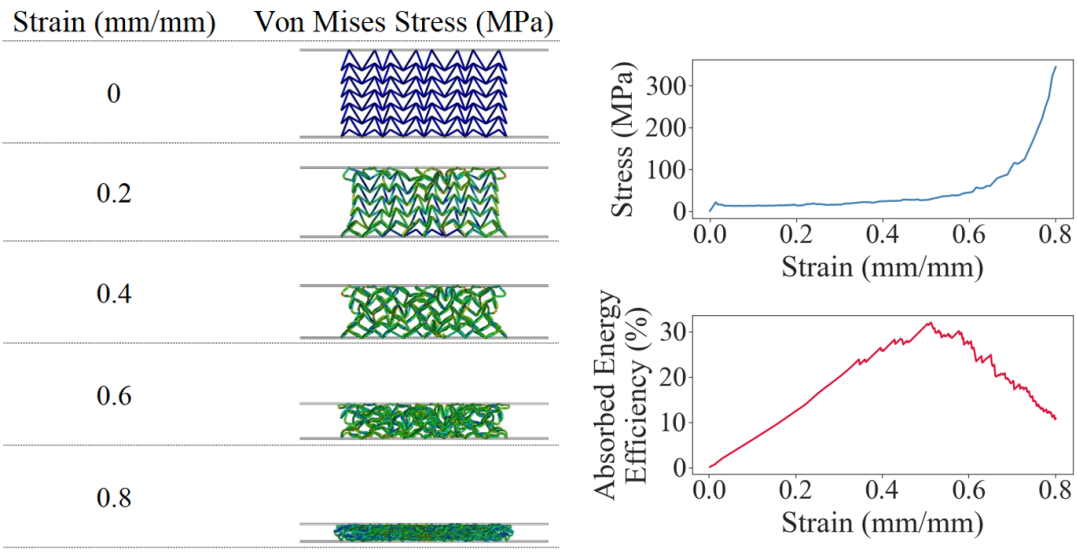


Figure 2.11: Deformation and result plots of one of the FE models, (px = 7.5, py = 7.1, t = 1.2)

Figure 2.12 shows the auxetic behavior of the same lattice structure (px = 7.5, py = 7.1, t = 1.2). In this configuration, a constant downward velocity is applied to the top plate, while the bottom plate is fixed and subjected to a plane-strain condition as previously described. The figure demonstrates that as the compression load increases, the lateral displacement vectors shift inward, characteristic of structures with Negative Poisson's Ratio (NPR). This notable property enhances the structure's resistance to buckling and indentation by increasing its stiffness, making it a promising candidate for energy absorption applications.

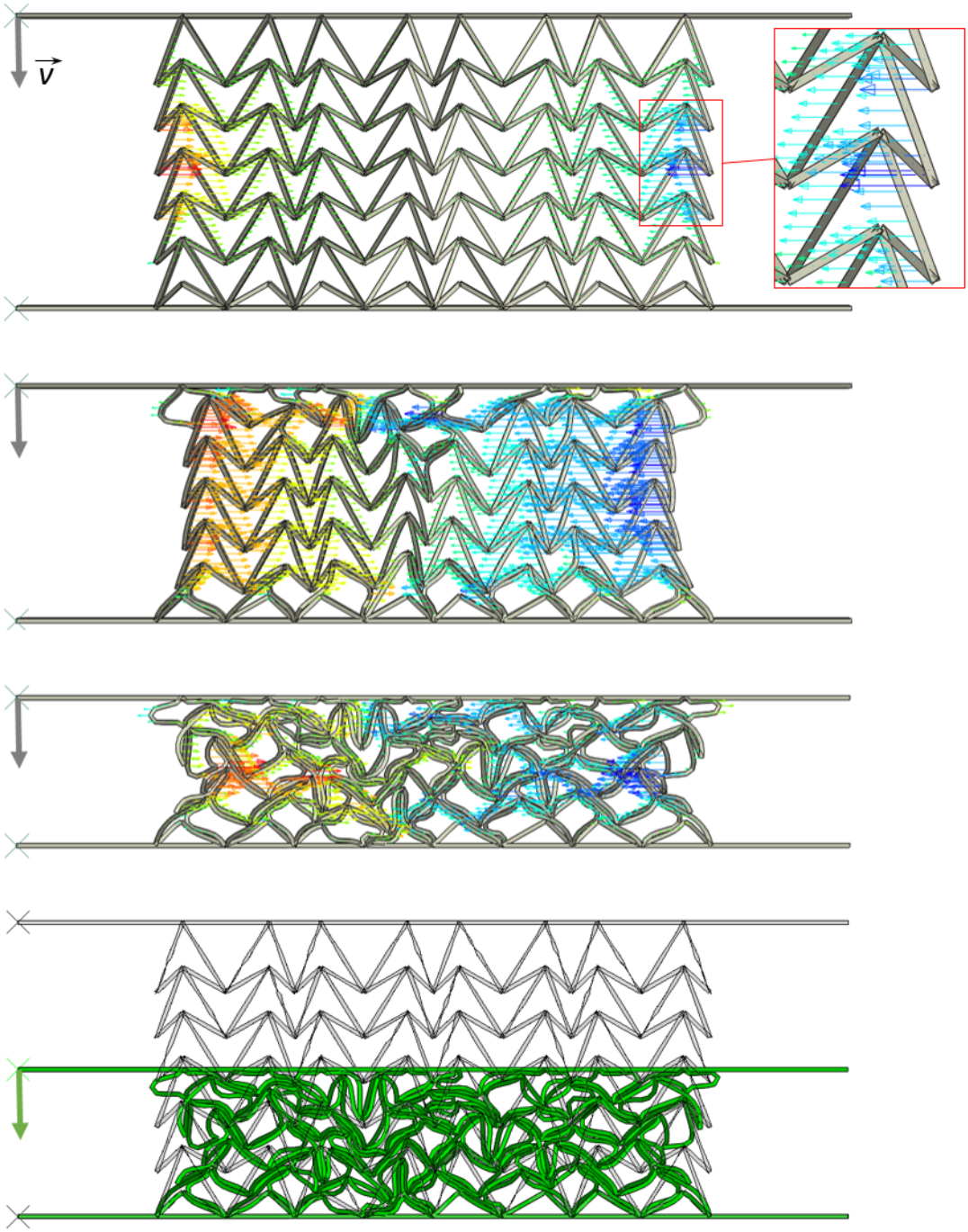


Figure 2.12: Displacement vectors in lateral direction of a lattice structure (px = 7.5, py = 7.1, t = 1.2) for different strain levels. At the bottom, the contraction of the structure can be clearly seen, indicating auxetic or NPR behavior, with the gray transparent model representing the undeformed structure and the green model representing the deformed structure.

The mechanical properties calculated from the raw data include: stress *(σ)*, strain *(ϵ)*, solid volume fraction *(svf)*, effective density *($\rho_{effective}$)*, yield stress *(σ_{yield})*, yield strain *(ϵ_{yield})*, modulus of elasticity *(E)*, densification stress *($\sigma_{dens.}$)*, densification strain *($\epsilon_{dens.}$)* which is also denoted as $\epsilon_{dens.}$, plateau stress *($\sigma_{plateau}$)*, work *(W)* or energy absorption *(EA)*, useful work *(W_{useful})*, specific useful work *(SW_{useful})*, and energy absorption efficiency *(n)* [83]. The formulas related to these properties are provided in Table 2.4. Note that the term *densification* is abbreviated as *dens.* in the equations shown in the table, and V represents volume whereas $w_{lattice}$, $d_{lattice}$, $h_{lattice}$ stands for width, depth and height of the lattice structure respectively.

It can be seen from the table that stress and strain calculated in a straight forward manner. For stress Equation 2.4 is used and applied force value at each time increment which is taken from the finite element analysis result, divided by the whole lattice structure cross section area as if it is a conventional solid structure. Equation 2.5 used to calculate strain and deformation at each time increment divided by the lattice structure height.

Solid volume fraction is a crucial parameter for lattice structures, calculated by dividing the lattice volume by the volume of the equivalent solid structure, based on its height, depth, and width (Equation 2.6). This property directly affects the stiffness, strength, and energy absorption capacity of the structure, but it also increases its weight. Therefore, while higher solid volume fractions are necessary for high energy absorption applications, an optimal value should be determined, taking the associated increase in weight into consideration. Another parameter related to solid volume fraction is effective density, which is calculated by multiplying the material density, in this case aluminum alloy, by the solid volume fraction (Equation 2.7).

After converting force-displacement values into stress-strain values, maximum stress value in the elastic region is taken as yield stress which is denoted in Equation 2.8 and corresponding strain value taken as yield strain, Equation 2.9. Similarly, densification stress is calculated by looking at the corresponding densification strain (Equation 2.11) which is calculated by mapping the strain value that is corresponding to the maximum energy absorption efficiency, Equation 2.12. From here it is also possible to calculate modulus of elasticity of the lattice structure by dividing the yield stress

to the yield strain as it shown in Equation 2.10. This value essentially represents the slope in the linear region of the stress-strain curve.

Table 2.4: Equations used to calculate the mechanical properties of lattices structures

Property	**Equation**	
Stress (MPa)	$\sigma = \frac{F_{applied}}{w_{lattice} \times d_{lattice}}$	(2.4)
Strain (mm/mm)	$\epsilon = \frac{\Delta h_{lattice}}{h_{lattice}}$	(2.5)
Solid volume fraction (mm^3/mm^3)	$svf = \frac{V_{lattice}}{V_{solid}}$	(2.6)
Effective density (g/mm^3)	$\rho_{effective} = svf \times \rho$	(2.7)
Yield stress (MPa)	$\sigma_{\text{yield}} = \max(\sigma \mid \text{elastic region})$	(2.8)
Yield strain (mm/mm)	$\epsilon_{\text{yield}} = \epsilon \mid \sigma = \sigma_{\text{yield}}$	(2.9)
Elasticity modulus (MPa)	$E = \frac{\sigma_{yield}}{\epsilon_{yield}}$	(2.10)
Densification stress (MPa)	$\sigma_{\text{dens.}} = \sigma \mid \epsilon = \epsilon_{\text{dens.}}$	(2.11)
Densification strain (mm/mm)	$\epsilon_{\text{dens.}} = \epsilon \mid n = \max(n)$	(2.12)
Plateu Stress (MPa)	$\epsilon_{plateau} = \frac{\int_{\epsilon_{yield}}^{\epsilon_{dens.}} \sigma(\epsilon) d\epsilon}{\epsilon_{dens.} - \epsilon_{yield}}$	(2.13)
Work or Energy Absorption (J)	$W \; or \; EA = \int_0^{\epsilon} \sigma(\epsilon) d\epsilon$	(2.14)
Useful work (J)	$W_{useful} = \int_0^{\epsilon_{dens.}} \sigma(\epsilon) d\epsilon$	(2.15)
Specific Useful work ($J.mm^3/g$)	$SW_{useful} = \frac{W}{\rho_{effective}}$	(2.16)
Energy absorption efficiency (%)	$n = \frac{\int_0^{\epsilon} \sigma(\epsilon) d\epsilon}{\int_0^1 \sigma_{ideal}(\epsilon) d\epsilon} \times 100$	(2.17)

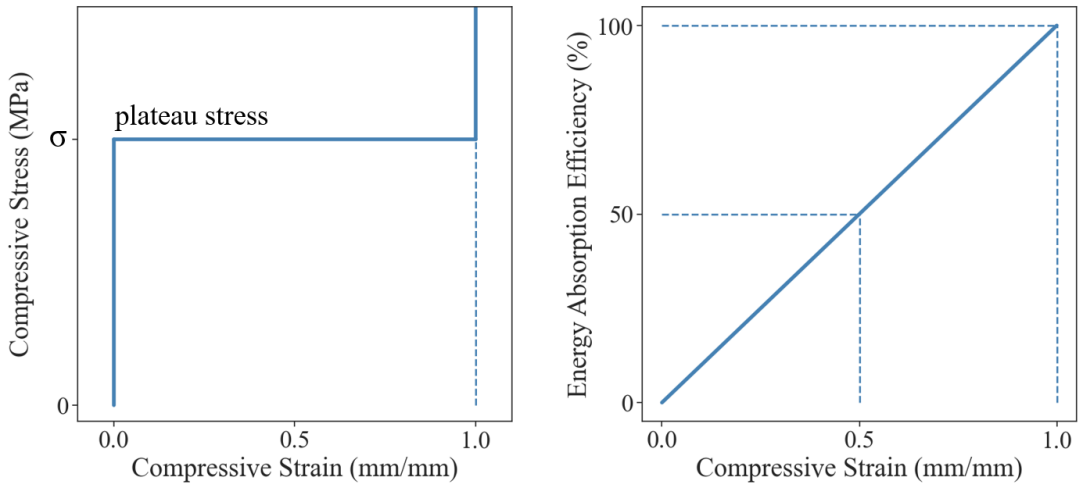


Figure 2.13: Ideal absorber properties, adapted from ref. [83].

Lastly, energy absorption efficiency is determined by dividing the energy absorbed by the lattice structure up to a specific strain value by the total energy absorption of an ideal absorber [83, 19] and is shown in Equation 2.17. An ideal absorber is a theoretical structure designed to absorb the maximum possible amount of energy while staying within a specified stress limit, plateau stress in this case. It immediately rises to this stress level and stays at this stress until 100% strain. This behaviour can be seen in Figure 2.13. It is desirable to achieve high energy efficiency values for an optimized lattice structure in impact-absorbing applications [83].

After calculating various mechanical properties of the lattice structures, common data exploration techniques are employed. Data exploration involves extracting information or knowledge from data [84]. This process may include detecting anomalies such as outliers, identifying trends, observing correlations between variables, and obtaining statistical values such as minimum, mean, and maximum. In this work, various data exploration techniques were applied to the tasks at hand.

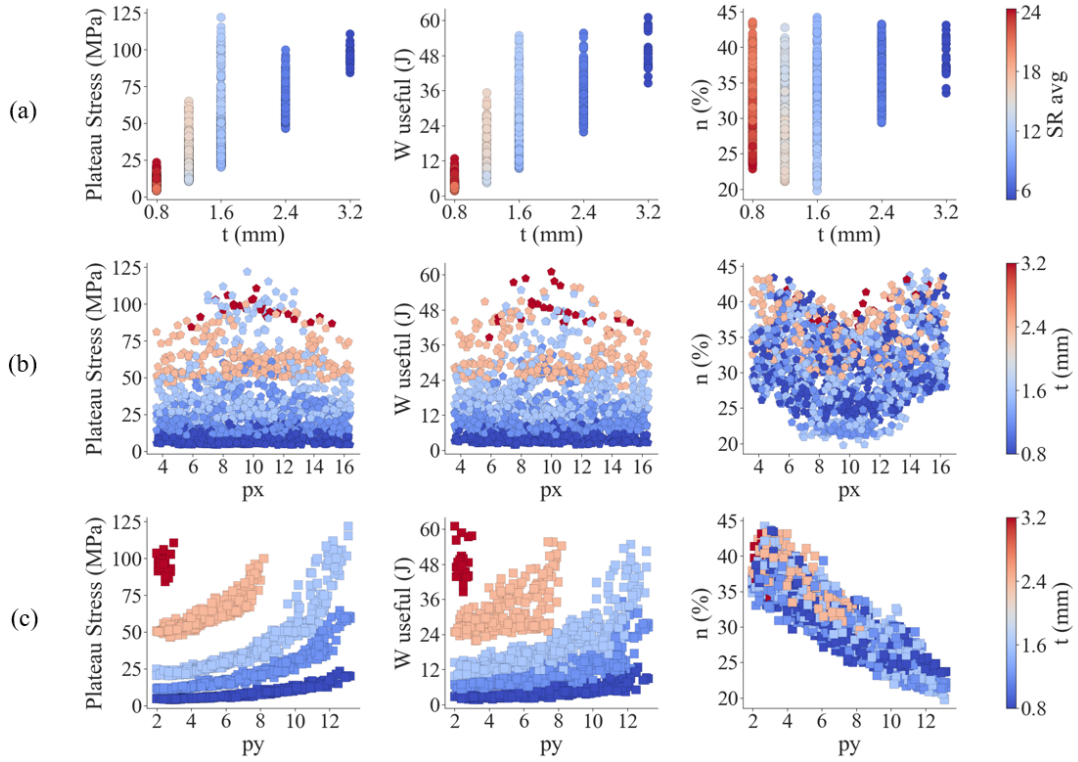


Figure 2.14: Trends in the data: (a) illustrates the relationship between thickness and the three output results, (b) depicts the trends between *px* and these outputs, and (c) highlights the correspondence between *py* and the same results.

From Figure 2.14, the different trends in the data can be seen. Figure 2.14a shows the effect of thickness on three important outputs. The average slenderness ratio in this figure, as formulated in Equation 2.18, is defined as the average length of the four main beams forming the unit cell divided by the unit cell thickness, t. It is observed that as the thickness increases, both the plateau stress and the useful work or energy also increase. However, no clear correlation or effect between thickness and energy absorption efficiency is observed. A reversed trend of it applies to the average slenderness ratio in which increasing average slenderness ratio decreases plateau stress and useful work with no effect on energy absorption efficiency. These observations suggest that higher thickness values enhance the unit cells' resistance to buckling, thereby increasing the plateau threshold. This effect also raises the useful energy by increasing the work required to buckle the unit cell beams. However, since energy absorption efficiency depends on both absorbed energy and the maximum stress

absorbed, an increase in both parameters does not result in a clear correlation with thickness in this case.

$$SR_{avg} = \frac{1}{4t} \sum_{i=1}^{4} \text{Strut Length}_i \tag{2.18}$$

Figure 2.14b illustrates the effect of *px* on the same three outputs. It shows no significant relationship with plateau stress and useful energy. However, *px* exhibits a subtle correlation with energy absorption efficiency. Although this correlation may not be immediately apparent, as *px* approaches its boundary values (e.g., 4 or 16 in the figure), energy absorption efficiency tends to increase, forming a recognizable V-shaped trend. This observation is further detailed in the correlation matrix discussed in the following sections.

Lastly, Figure 2.14c illustrates the effect of *py* on the same properties. It reveals a subtle relationship between *py* and plateau stress, as well as useful work. By examining the data with the same thickness values, an exponential curve is discernible. This trend can be attributed to the following: as *py* increases, the second arrowhead point (P4) becomes more vertically oriented, which allows the structure to absorb more energy before buckling, thereby increasing both plateau stress and useful work. Conversely, *py* exhibits a more pronounced negative relationship with energy absorption efficiency, as clearly depicted in the figure.

Another examination of the variable *py*, as depicted in Figure 2.15, reveals a positive correlation between *py* and energy absorption. Counter-intuitively, higher energy absorption values are associated with lower energy absorption efficiencies. Thus, while energy absorption increases with *py*, the energy absorption efficiency decreases.

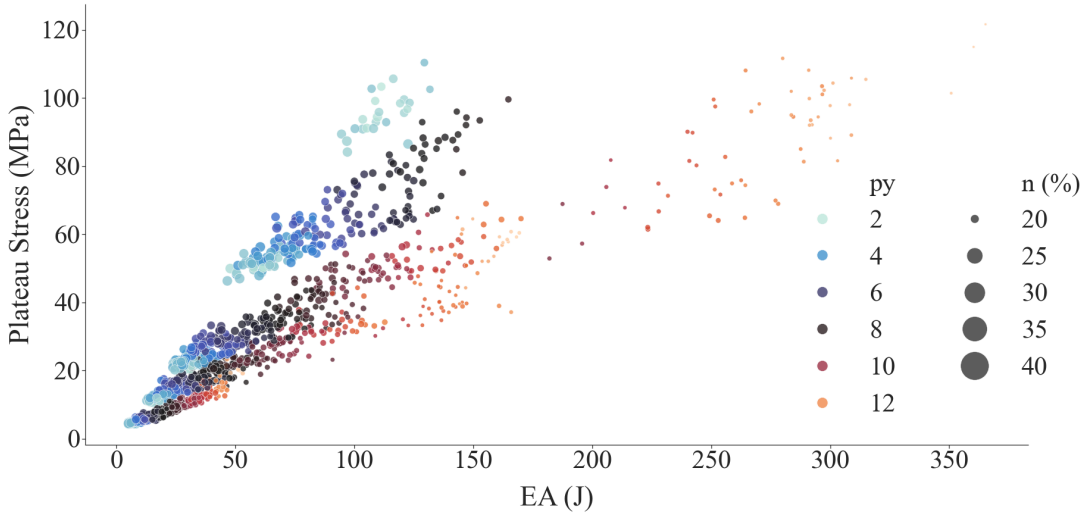


Figure 2.15: Relation between four parameters: Plateau Stress, Energy Absorption, Energy Absorption Efficiency and *py*.

Another method for analyzing the data is through a *correlation matrix*, which visually represents pairwise correlations between variables. This matrix provides a clear overview of trends: light colors indicate positive correlations, dark colors denote negative correlations, and intermediate colors reflect weak correlations.

Figure 2.16 presents the correlation matrix derived in this study. Examination of the figure reveals that the average slenderness ratio exhibits a negative correlation with most mechanical properties, with the exceptions of energy absorption efficiency and, to a lesser extent, densification strain. In contrast, thickness (t) generally demonstrates a positive correlation with most properties, excluding densification strain and energy absorption efficiency.

The parameter *py* shows a strong negative correlation with energy absorption efficiency and densification strain but displays a positive correlation with energy absorption, modulus of elasticity, densification stress, and the average slenderness ratio. Conversely, *px* does not exhibit significant correlation with any specific outputs. However, applying the transformation defined by Equation 2.19 to convert *px* into *px'* uncovers notable relationships. This transformation emphasizes the lateral displacement of *px* from the center of the unit cell. Specifically, as *px'* increases, the second arrowhead point moves farther from the center, leading to an asymmetrical unit cell.

Following this transformation, *px'* is observed to have a positive association with energy absorption efficiency and the modulus of elasticity, while generally showing a negative correlation with energy absorption and yield strain. While this behavior may initially seem counterintuitive, machine learning models are capable of capturing the V-shaped trend exhibited by *px* (Figure 2.14b) in relation to energy absorption efficiency. Consequently, *px* was retained in its original form for training the models rather than using the transformed version, *px'*.

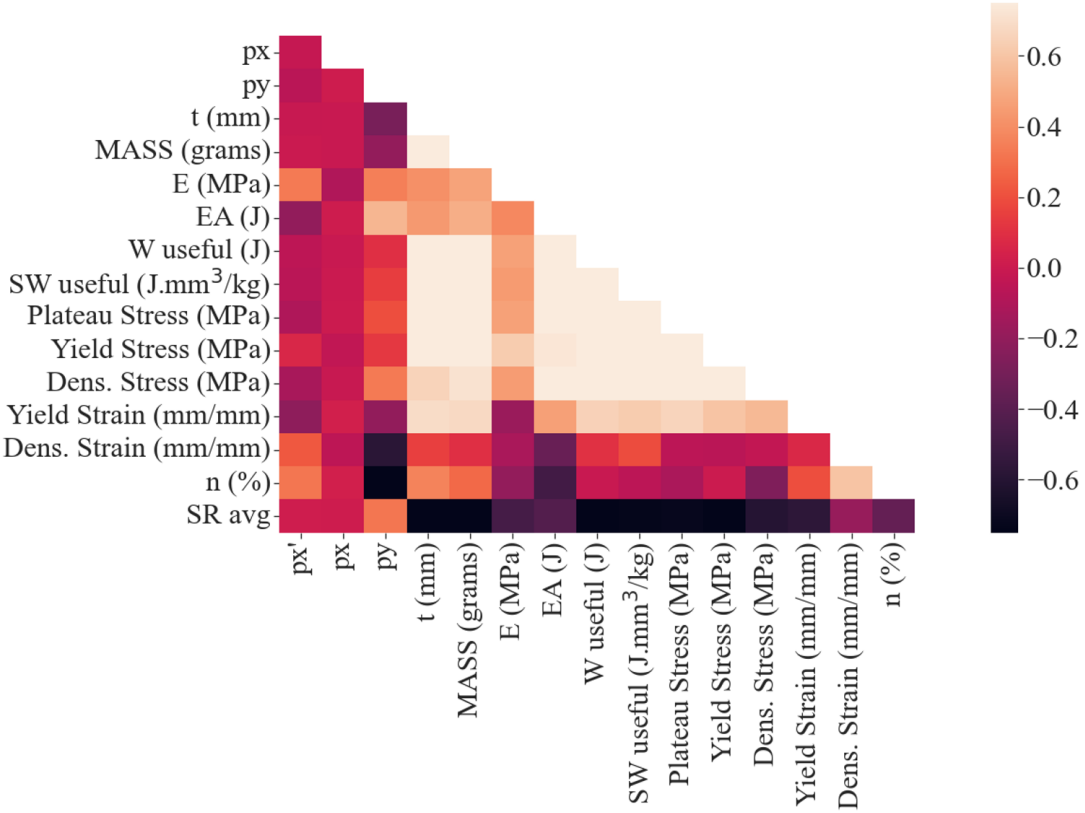


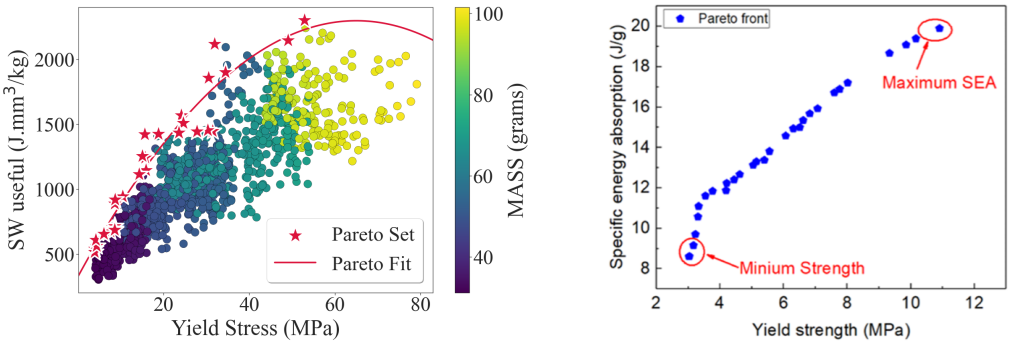
Figure 2.16: Correlation matrix

The figure also reveals that stress-related properties and energy absorption characteristics, with the exception of energy absorption efficiency, exhibit a positive correlation, as indicated by the light colors in the matrix.

$$px'_i = px_i - \frac{Unit\ cell\ width}{2} \tag{2.19}$$

where *px'* represents the lateral distance from the center of the unit cell, *px* is the default value, and the unit cell width divided by two provides the lateral coordinate of the center of the unit cell.

Moreover, Figure 2.16 shows a notable negative correlation between densification strain and *py*, with no similar correlation observed with *px* or *t*. Furthermore, densification strain does not significantly correlate with other outputs, in contrast to the mutual correlations observed among those outputs. This lack of correlation may suggest less reliable predictions for densification strain. While a simulation error could potentially explain this anomaly, it is unlikely given that densification strain is the only output exhibiting this trend. Consequently, this study excludes the prediction of densification strain and yield strain to avoid incorporating strain value predictions into the results.



(a) (b)

Figure 2.17: Pareto plots: a) Pareto set obtained in this work for the following objectives in the data: maximize specific useful work, minimize the yield stress, minimize mass of the structure, b) Pareto set from the work of Wang et al. [85] with the objectives of: maximize specific energy absorption, minimize the yield strength)

Additionally, multi-objective optimization can be performed using the Pareto set technique. It is generally used where several conflicting objectives need to be optimized at once. Figure 2.17 illustrates the application of this method. In the figure, the

Pareto set identifies optimal points within the data, achieving the objective of maximizing specific useful work while minimizing the yield stress and mass of the structure. These three objectives are selected based on the work of Wang et al. [85]. A second-order curve can be fitted to the Pareto set to delineate the edge of the optimal design space. This construction is consistent with the findings of Wang. Such a curve provides flexibility in selecting solutions optimized for this multi-objective task.

CHAPTER 3

MACHINE LEARNING METHODOLOGY AND MODEL TRAINING

The research outlined in the preceding chapters produced a dataset of 1401 samples, comprising features and corresponding mechanical properties derived through mathematical operations. Subsequently, the data was explored to extract meaningful insights. This dataset was then prepared and utilized to train three machine learning models. After presenting the results, an optimization study was performed by integrating machine learning with a genetic algorithm, which significantly enhanced the efficiency of the optimization task.

3.1 Pre-processing the Data

The extracted mechanical properties and input features are compiled into a structural dataset, which is then utilized to train three machine learning models: Neural Networks (NN), Random Forest (RF), and Extreme Gradient Boosting (XGB). These algorithms were chosen for their advantages, including popularity, accuracy, resistance to overfitting, and suitability for structural data.

Before training the model, to enhance its generalization capabilities and avoid overfitting, outliers among the features of Young's modulus, energy absorption, and useful work were detected and removed from the dataset. These features were selected based on trial and error, aiming to balance prediction accuracy with retaining a significant number of samples. Figure 3.2 shows the data distribution before and after the outlier removal process. Outliers were detected using Equation 3.1, which is a well-established method for such tasks [86].

As shown in Figure 3.1, the interquartile range (IQR) was calculated based on Q1 (the lower quartile), which represents the lowest 25% of the data, and Q3 (the upper quartile), which represents the lowest 75% of the data. Upper and lower thresholds were then determined using these values, and data outside these thresholds were removed. This technique reduced the sample size from 1401 to 1293. The figure also illustrates that most outliers were removed from the energy absorption (EA) and densification stress (Dens. Stress) features.

$$
\begin{aligned}
Q1 &= 0.25 \times (n+1)^{th}\, sample \\
Q3 &= 0.75 \times (n+1)^{th}\, sample \\
IQR &= Q3 - Q1 \\
T_{upper} &= Q3 + 1.5 \times IQR \\
T_{lower} &= Q3 + 1.5 \times IQR \\
data &= T_{lower} \leq data \leq T_{upper}
\end{aligned}
\tag{3.1}
$$

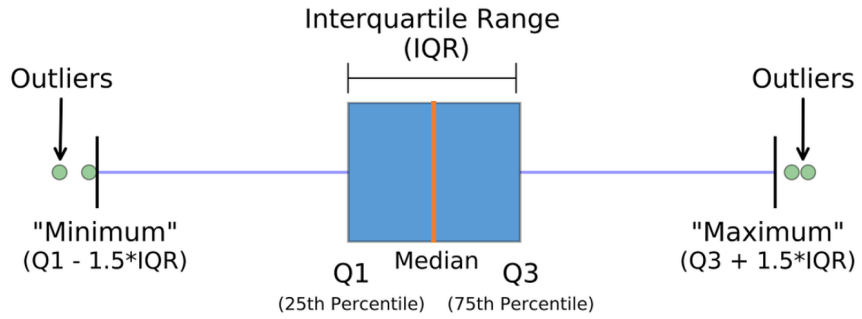
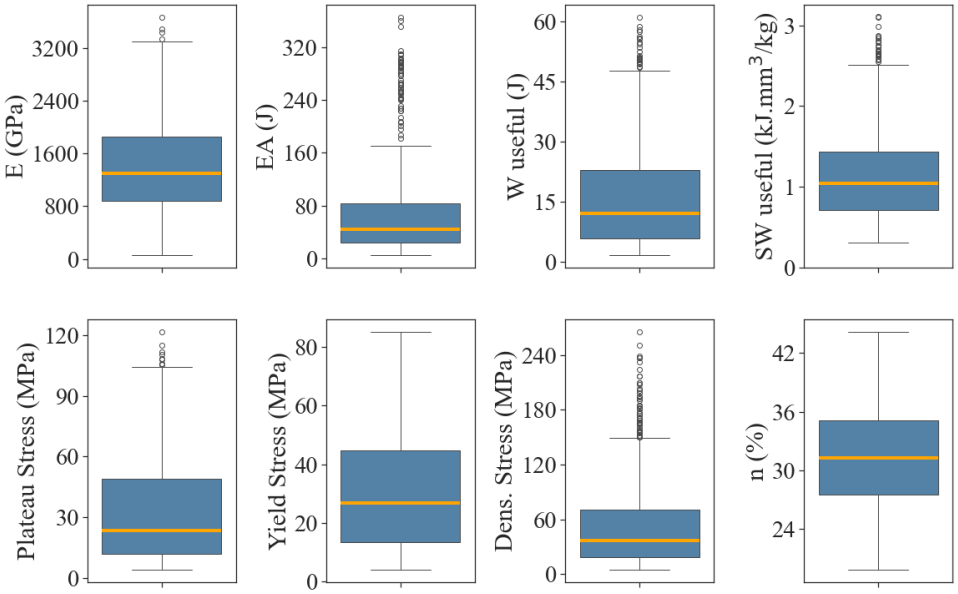
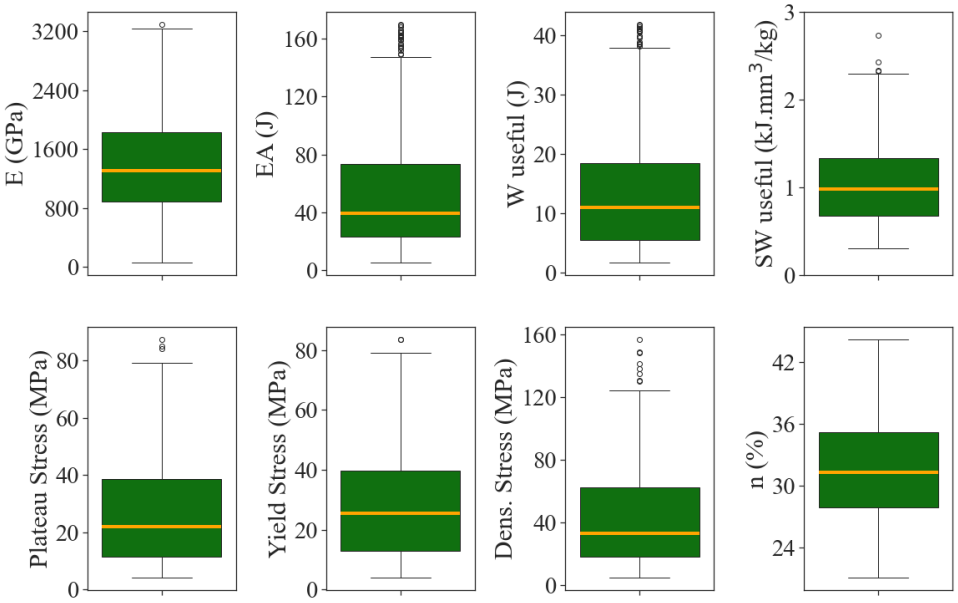


Figure 3.1: IQR box representation and outliers detection, adapted from ref. [87].

Another crucial step before training and prediction is data scaling, if necessary. Scaling is required because features with higher gradients can disproportionately influence the model weights, potentially dominating the model. This is particularly relevant for gradient-based algorithms, such as neural networks, which benefit from such preprocessing. In contrast, tree-based models, such as random forests and extreme gradient boosting, are less affected by feature scaling, even though extreme gradient boosting utilizes gradient-based operations.

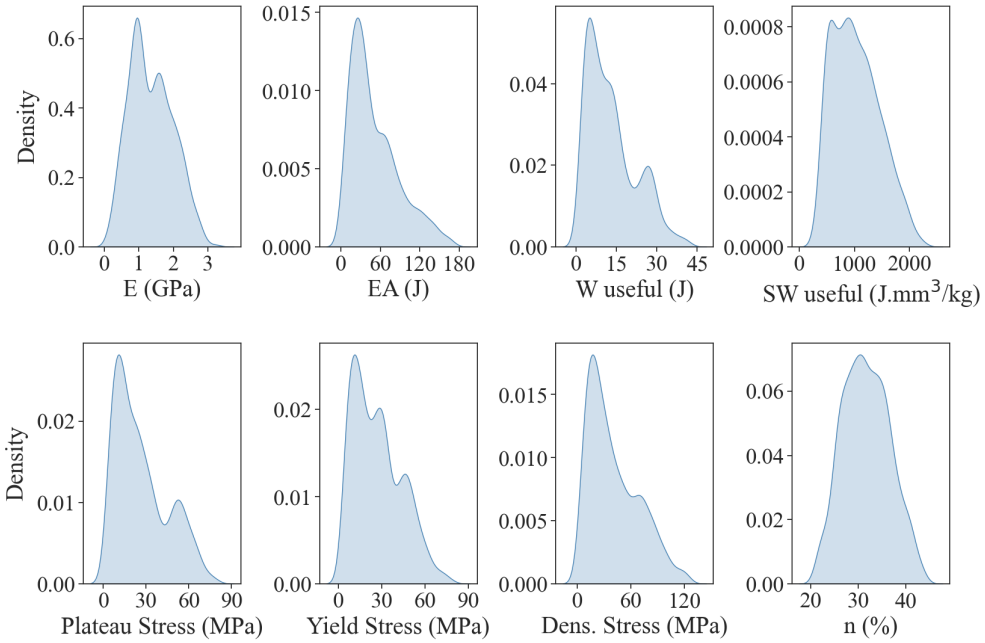


(a) Data distribution before outliers are removed.

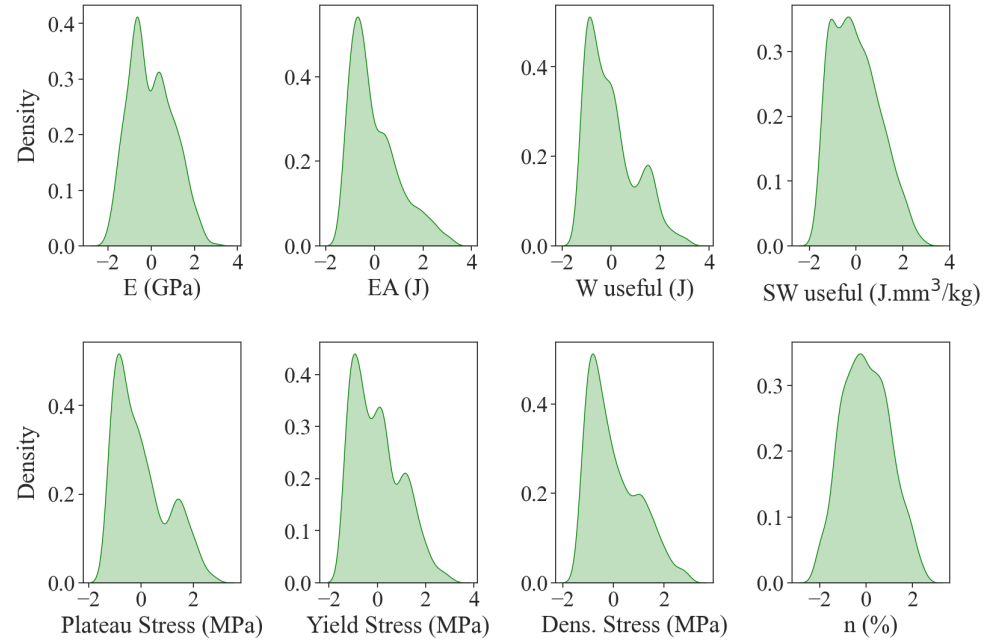


(b) Data distribution after outliers are removed.

Figure 3.2: Detecting and removing the outliers



(a) Before standardization



(b) After standardization

Figure 3.3: Standardization of target values is illustrated, with the y-axis representing the density which is the smoothed approximation of the probability distribution of a dataset at a specific point.

In this study, standardization was employed as the scaling method for the data. This process adjusts the data by setting its mean to zero and scaling it to unit variance, as outlined in Equation 3.2. Through trial and error, it was observed that standardization offered superior benefits to neural networks compared to normalization, making it the preferred scaling technique. Additionally, the literature indicates that effective regularization requires all features to be on comparable scales [44], a requirement that standardization fulfills effectively.

$$x' = \frac{x - \bar{x}}{\sigma} \tag{3.2}$$

where x' is the standardized value, x is the raw value, $\bar{x}$ is the mean of the feature across all samples, and σ is the standard deviation. The result of this operation is illustrated in Figure 3.2.

3.2 Hyperparameter Optimization

Machine learning algorithms are known to have numerous hyperparameters that can be adjusted to enhance model performance [41]. This process, known as hyperparameter optimization, is crucial and should not be overlooked. In this study, the hyperparameters for three models were tuned using the *grid search* method. This technique evaluates all possible combinations of given parameters and identifies the optimal hyperparameters within the specified search space based on their performance scores. Essentially, it involves systematically searching a grid space and storing the results.

This search method also incorporates the *k-fold cross-validation* technique, with 10 folds (k = 10) used in this study. Cross-validation is employed to mitigate overfitting [41]. The search space and optimal hyperparameters are detailed in Table 3.1. It is important to note that hyperparameter optimization was performed using only the training dataset to avoid data leakage; the test dataset was not used during this process. The dataset, consisting of 1293 samples after outlier removal, was divided into 1163 training samples and 130 test samples, with the latter constituting 10% of the data. A representative workflow of hyperparameter optimization, cross-validation, and subsequent performance evaluation is illustrated in Figure 3.4.

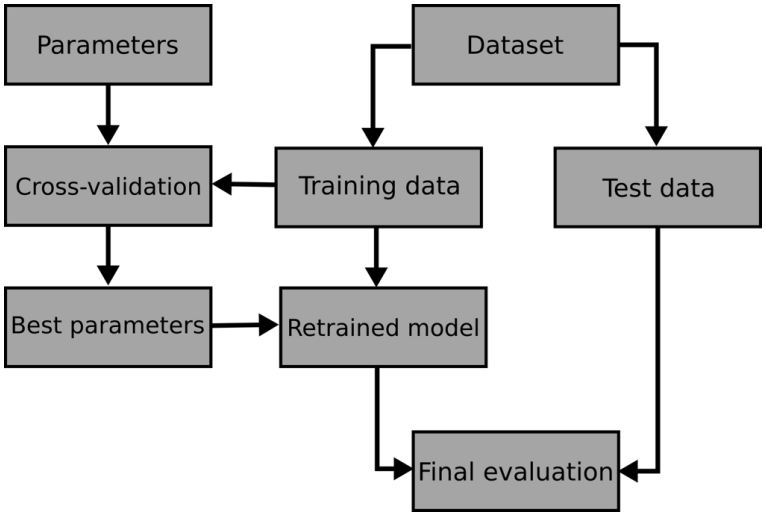


Figure 3.4: The training and evaluation workflow includes cross-validation and hyperparameter tuning, adapted from ref. [88].

Another important point is that hyperparameter optimization was not performed for each individual output, although such an approach would be valid. Instead, it was performed on a combined output vector, termed the *hypervector*, with each selected output being scaled prior to their combination. This process is illustrated in Equation 3.3, where H represents the hypervector, serving as the basis for hyperparameter optimization, and y_i denotes the individual target values.

$$H = \sqrt{\sum_{i=1}^{n} y_i^2} \tag{3.3}$$

In Table 3.1, for the Neural Network (NN), the *hidden layer size* provides information about the number of hidden layers and neurons in the network, excluding the input and output layers. For instance, $6, 12, 6$ indicates three hidden layers with 6, 12, and 6 neurons, respectively. To optimize the number of neurons and layers, two configurations were considered during hyperparameter optimization: 6, 12, 6 neurons with 3 layers, and 3, 6, 6, 3 neurons with 4 layers. The rationale behind this choice is to avoid introducing too many neurons, which could lead to overfitting, while also exploring the effects of deep learning by increasing the number of hidden layers to more than three [44].

These two neuron configurations used to initiate hyperparameter optimization in the Neural Network algorithm are consistent with the number of neurons suggested by Sheela et al. [89], which is 39, as shown in Equation 3.4, where N_h represents the number of neurons in the hidden layers, and n denotes the number of inputs. Furthermore, it should be noted that the number of neurons required to build the network may increase as the number of samples used to train the Neural Network grows [90]. Another alignment can be found in the work of Xu et al., where Equation 3.5 provides a rounded value of 28 neurons, ensuring that this value does not exceed the suggested threshold. This work satisfies the equation with options of 24 and 18 neurons [91]. In this equation, N_t denotes the number of training samples, and N_i represents the number of inputs. Ultimately, it should be noted that, at the time this study was conducted, there is no *golden rule* for determining the optimal number of neurons and layers for Neural Networks. Such parameters must be tailored to each specific problem individually [90].

$$N_h = \frac{4n^2 + 3}{n^2 - 8} \tag{3.4}$$

$$N_h = \begin{cases} \frac{1}{2}\frac{N_t}{N_i \log N_t}, & \text{if } \frac{N_t}{N_i} > 30, \\ \frac{N_t}{N_i}, & \text{else } \frac{N_t}{N_i} \leq 30 \end{cases} \tag{3.5}$$

For the ensemble methods of Random Forest (RF) and Extreme Gradient Boosting (XGB), hyperparameter options are broadly chosen as multiples of the default value of 100, which is provided by the machine learning library. This parameter is represented as *n estimators* in Table 3.1. The starting value of 100 decision trees aligns with the findings of Oshiro et al., who suggested using trees in the range of 64 to 128, although their work is based on classification rather than regression [92]. However, in a more recent work, Curth et al. noted that using more trees does not increase the generalization error and it results in smoother forest predictions by reducing the tendency to fit noise in the data [93]. Since many decision trees share their own portion of this noise, while others share no noise in this case, having more trees averages the model fit down to smoother levels. Consequently, the default value of 100 decision trees is retained, while additional options of 200 and 400 are explored to determine

if the model benefits from an increased number of trees. As it is mentioned before, there is no universal guideline for hyperparameters applicable to all problems when utilizing these models; thus, hyperparameter selection is problem-dependent and requires a trial-and-error approach. By employing the *grid search* algorithm within this parameter space, it is possible to identify optimal parameters for the problems under investigation. Furthermore, the *grid search* algorithm systematically searches through the parameter space to discover optimal values, but it necessitates a relatively high computational effort. Therefore, the provided options for each hyperparameter are limited to two to four choices.

Another important hyperparameter observed in the neural network algorithm is the *activation function*. This function introduces non-linearity to the network by applying either the hyperbolic tangent function (*tanh*) or the rectified linear unit (*relu*) to the nodal values. The L2 regularization term, represented by *alpha* in Table 3.1, is employed to prevent overfitting by appending a penalty to the loss function that is proportional to the summation of the squared values of the weights. Thus, it effectively penalizes excessive weight accumulation, contributing to better generalization. Other parameters such as the type of learning rate, in conjunction with the learning rate itself, determines whether the learning rate should vary during training. This can be set as either constant or adaptive. For the tree ensemble algorithms, Random Forest (RF) and Extreme Gradient Boosting (XGB), the *max depth* hyperparameter controls the depth of the trees, while *min child weight* sets the minimum weight threshold required to create a new node in the tree. The *max features* parameter defines the number of features considered when selecting the optimal split, and *min samples split* determines the minimum number of samples necessary to split an internal node.

Table 3.1: Hyperparameter optimization results

Algorithm	**Search Space**	**Optimum Hyperparameters**
Neural Network	*hidden layer size*: [(6,12,6), (3,6,6,3)]	*hidden layer size*: (6,12,6)
	activation function: [tanh, relu]	*activation function*: tanh
	alpha: [0.0001, 0.001]	*alpha*: 0.0001
	learning rate type: [constant, adaptive]	*learning rate type*: constant
	learning rate: [0.001, 0.01, 0.1]	*learning rate*: 0.01
XGBoost	*n estimators*: [100, 200, 400, 1000]	*n estimators*: 100
	max depth: [4, 8, 16, None]	*max depth*: 4
	min child weight: [1, 2, 4, 8]	*min child weight*: 1
	learning rate: [0.0001, 0.001, 0.01, 0.1]	*learning rate*: 0.1
Random Forest	*n estimators*: [100, 200, 400, 1000]	*n estimators*: 1000
	max features: ['sqrt', 'log2', None]	*max features*: None
	max depth: [4, 8, 16, None]	*max depth*: 16
	min samples split: [2, 4, 8, 16]	*min samples split*: 8

3.3 Training of the Models

After removing outliers, scaling the data, and performing hyperparameter optimization, the data was ready for model training. In this study, three machine learning models were employed: NN, RF, and XGB. These models were chosen based on their accuracy, computational efficiency, compatibility with the data, ease of use, and widespread popularity.

3.3.1 Neural Network

The neural network (NN) architecture used in this thesis, as shown in Figure 3.5, comprises three hidden layers with 6, 12, and 6 neurons, respectively. These configurations were determined through hyperparameter optimization, as discussed in the previous chapter. Although there is no strict rule regarding the subject, a neural network with three hidden layers can be considered a *deep artificial neural network*, and training such models is referred to as *deep learning* [44]. This type of neural net-

work, referred to as a multi-layer perceptron (MLP), serves as a foundational architecture in neural networks. More advanced architectures, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and long short-term memory (LSTM) networks, are developed based on this core structure.

The goal of training a neural network model is to optimize the weights, which represent the connections between neurons, in order to minimize the output error, which is the difference between the true and predicted values. This optimization process uses a method known as backpropagation, which involves calculating the gradient of the error with respect to each weight in the network to determine how the error changes as each weight is adjusted. The weights are then iteratively updated during each batch of data until a stopping criterion is satisfied, such as reaching a predefined maximum number of iterations or attaining an acceptable error tolerance. Once this process is complete, the neural network is trained and ready for making predictions.

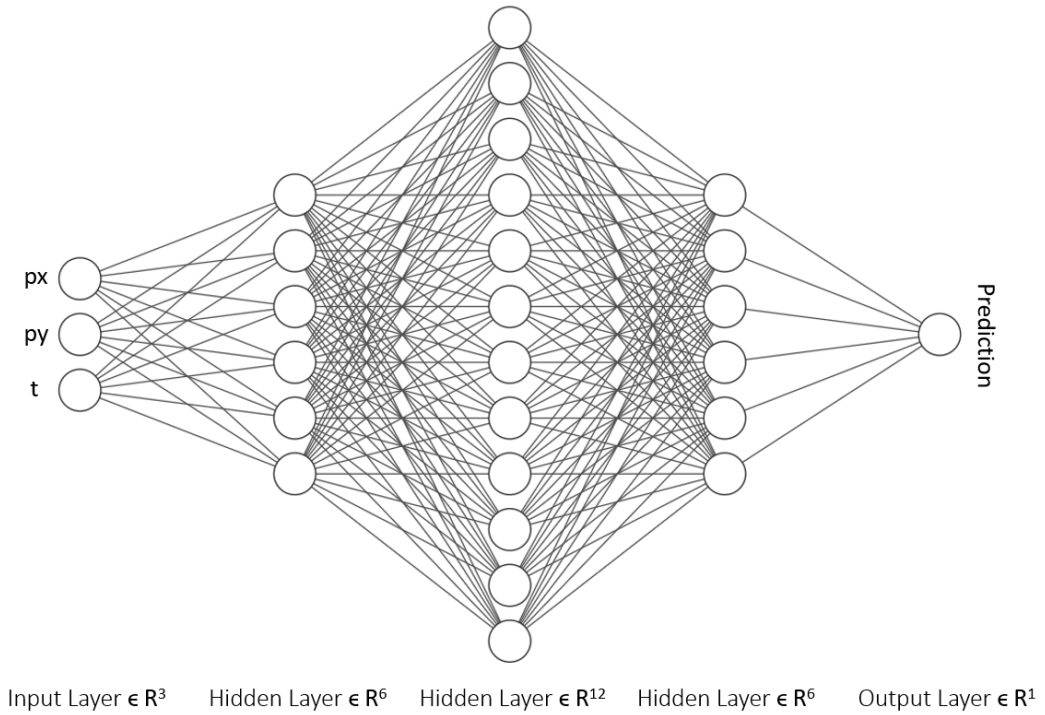


Figure 3.5: Neural network representation used in this study, adapted from [94].

3.3.2 Extreme Gradient Boosting

To understand the logic behind Extreme Gradient Boosting (XGB), it is essential to first grasp the concept of decision trees. Decision trees are a machine learning algorithm used for both regression and classification tasks. At each node of the tree, a condition is applied to the data. Depending on whether the condition is met, the data is split into subsets. This process continues recursively until the tree meets a stopping criterion, such as reaching an impurity level or a maximum depth. The final prediction is made by aggregating the target values of the data points in the terminal nodes (or leaves) of the tree. By combining multiple decision trees through various ensemble methods, algorithms such as XGB are developed. The basic structure of XGB is illustrated in Figure 3.6.

XGBoost (XGB) is widely recognized as one of the most effective models for regression tasks involving tabular data [95] and is frequently recommended for such applications [96, 97]. XGB utilizes the *boosting* technique, where each subsequent decision tree is constructed to correct the errors made by the preceding trees, thereby aiming to minimize residuals. This correction is achieved by employing the *gradient* of the loss or error with respect to the previous predictions, along with additional regularization terms to enhance model performance and make it *extreme*. Hence, the algorithm is called *extreme gradient boosting*, which highlights its advanced optimization techniques and robust regularization methods.

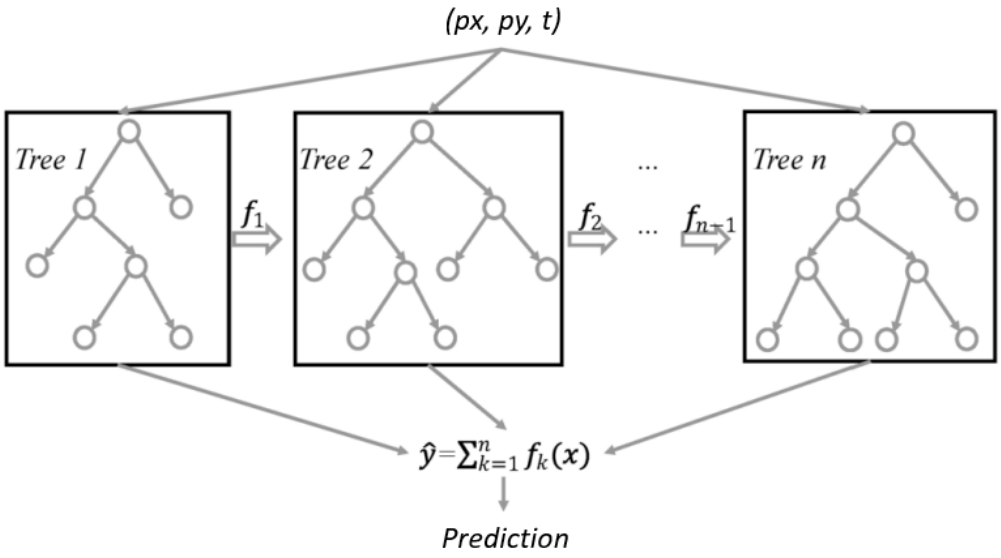


Figure 3.6: XGBoost illustration, adapted from ref. [98].

In Figure 3.6, f_k denotes an individual decision tree, while $f_k(x_i)$ represents the prediction of the kth tree for the input x_i. The symbol $\hat{y}_i$ indicates the predicted value for the ith sample [98]. Unlike Random Forest, which aggregates tree predictions by averaging, XGBoost combines them by summing, as illustrated in the figure with $\hat{y}$. In this study, hyperparameter tuning determined that the number of trees used in XGBoost, denoted as n in the figure, is 100.

3.3.3 Random Forest

Another ensemble method utilizing decision trees is Random Forest (RF). In this regard, it is comparable to XGBoost (XGB). An illustration of the Random Forest model is provided in Figure 3.7. In this study, hyperparameter optimization established that the Random Forest model comprises 1000 decision trees. Predictions in Random Forest are generated using a technique known as *bagging*. This technique involves training each decision tree on a randomly selected subset of the dataset, with each subset corresponding to a distinct tree. Additionally, Random Forest employs random feature selection at each split of the decision trees, further enhancing the model's robustness. The term *random* in its name reflects the use of randomness in

both data subset selection and feature choice, while *forest* denotes the ensemble of multiple decision trees. Hence, the name *Random Forest*.

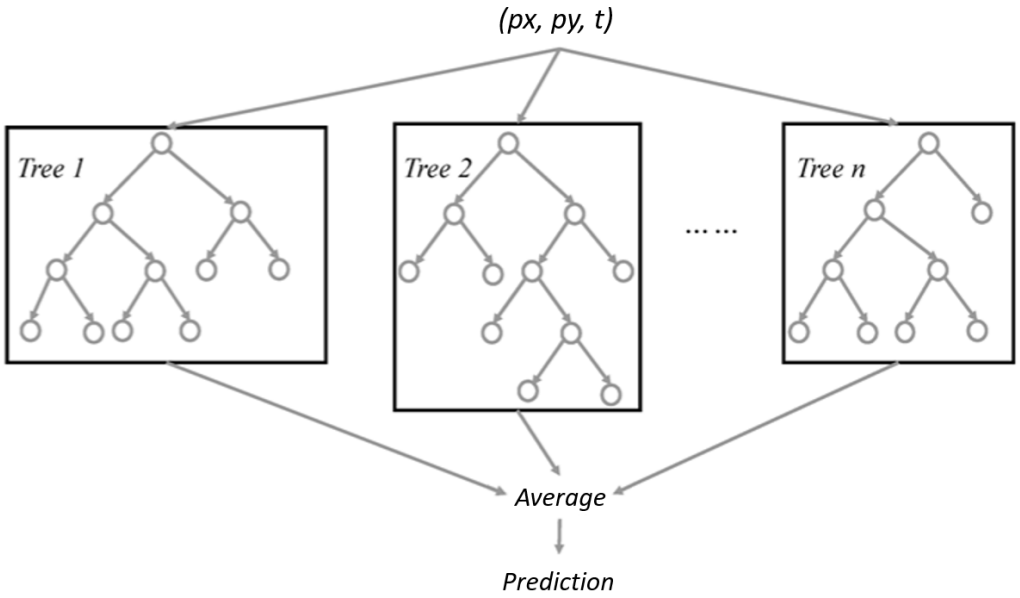


Figure 3.7: Random Forest illustration, adapted from ref. [98].

The final prediction in Random Forest is obtained by averaging the outputs of all the decision trees, as illustrated in Figure 3.7. Consequently, Random Forest can be described as a method for averaging decision trees. This approach enhances the model's robustness and reduces the risk of overfitting by reducing variance. However, it may also lead to increased bias and reduced interpretability of the model.

3.4 Predictions and Discussion

The training was performed on NN, RF and XGB models. A common mistake among beginners in the field is to evaluate performance based on the training data, which can create an illusion of model accuracy [41]. To avoid this error, performance evaluation in this study is conducted using the test set, which was held out and not used until this point in the process to ensure an unbiased assessment of model performance.

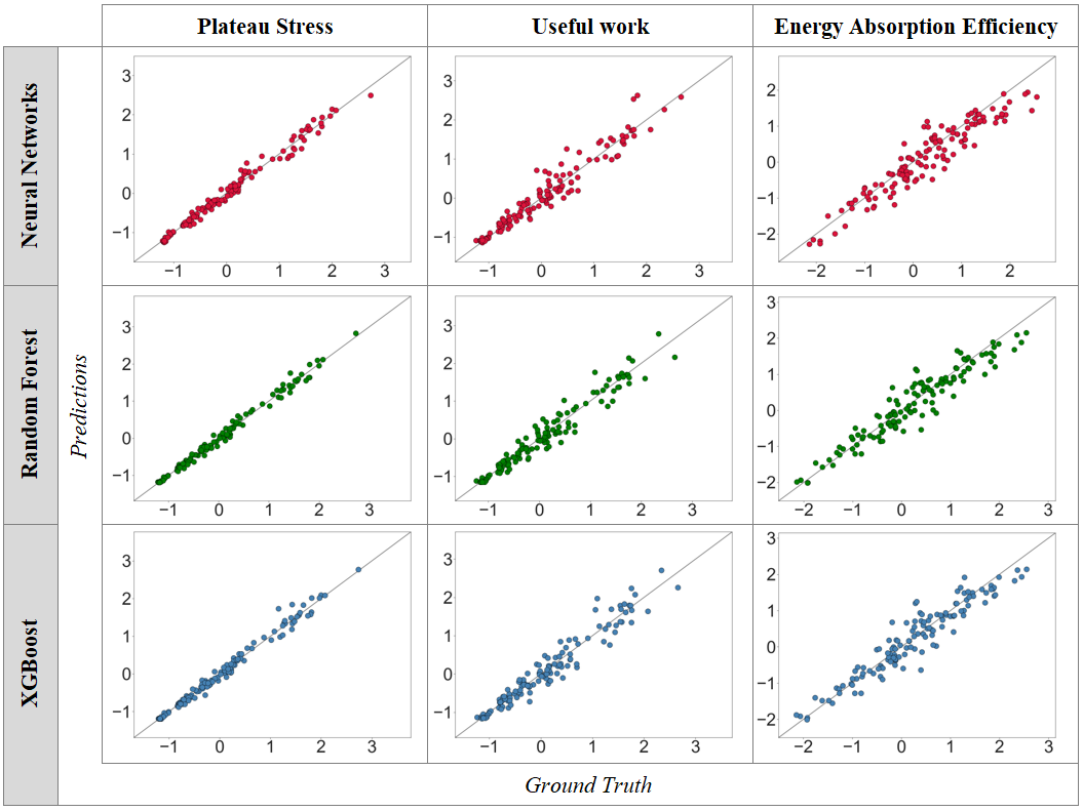


Figure 3.8: Scatter plot of model predictions versus simulation results (ground truth).

Figure 3.8 presents the prediction results from the three machine learning models for three different outputs. The results are depicted with scaled axes. The *ground truth* axis represents the true values obtained from the simulations, while the *predictions* axis displays the values predicted by the models. The gray line with a 1:1 slope denotes ideal prediction accuracy; points lying on this line correspond to accurate predictions, whereas deviations from the line indicate varying degrees of prediction accuracy.

Although the visual results are generally satisfactory, the prediction accuracy for plateau stress and useful work decreases as the values increase. This trend is also observed in other studies in the literature [71, 70]. This decreased accuracy for higher values may be due to low probability outcomes or skewness in the training data, as seen in Figure 3.9. Addressing this issue would require exponentially more data, as each sample is more likely to be closer to the average rather than being an extreme value.

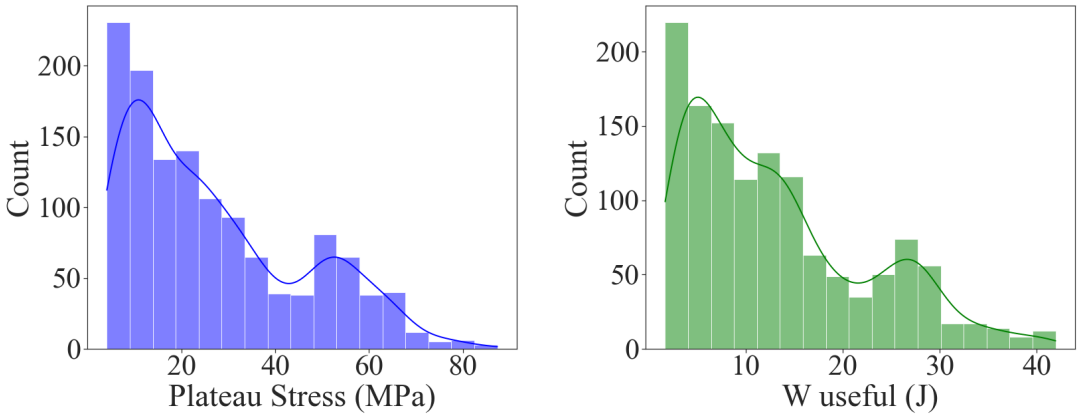


Figure 3.9: Skewness in the data distribution of two outputs

The final results of the predictions, along with quantitative metrics, are presented in Table 3.2. Two regression metrics are used in this study: mean absolute percentage error (MAPE) and correlation coefficient (R^2). Lower percentage values for MAPE indicate better results, while R^2 ranges between 0 and 1, with higher values closer to 1 indicating a stronger correlation between the predictions and the test data. The mathematical formulas for these two metrics are provided in Equation 3.6 and 3.7.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \tag{3.6}$$

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{3.7}$$

where n is the number of samples, y_i represents the ground truth values, $\bar{y}$ is the mean of the ground truth values, and $\hat{y}_i$ denotes the predicted values.

From the results shown in Table 3.2, the RF model achieved the best performance with an average R^2 of 0.936 and a mean absolute percentage error (MAPE) score of 1.775. This outcome is consistent with the observation that tree-based methods, particularly XGB, often outperform NN on tabular or structured data [95, 96, 97]. Conversely, DNNs generally excel with larger datasets compared to traditional methods [99]. Table 3.2 also reveals that specific useful work exhibited the poorest performance, with an average error of 11.7% across the three models. This target value is calculated by

Table 3.2: Prediction results of the machine learning models

	NN		**RF**		**XGB**	
Young's Modulus	0.30	0.93	0.33	0.91	0.37	0.92
Energy Absorption	0.23	0.99	0.20	0.98	0.19	0.98
Useful Work	1.53	0.93	1.44	0.94	1.48	0.94
S. Useful Work	14.23	0.86	8.43	0.85	12.46	0.84
Plateau Stress	0.23	0.98	0.19	0.99	0.22	0.98
Yield Stress	0.15	0.99	0.09	0.99	0.15	0.99
Dens. Stress	2.40	0.91	2.35	0.91	2.25	0.91
Energy Absorp. Eff.	1.37	0.88	1.18	0.91	1.37	0.90
	MAPE	R^2	*MAPE*	R^2	*MAPE*	R^2

dividing the effective density, which reduces the correlation between thickness and this target. Since independent variables with strong correlations to the target value are crucial for accurate predictions, this calculation adversely affects performance. Additionally, the table indicates that as R^2 increases, MAPE tends to decrease, suggesting that improved correlation is associated with reduced percentage error.

In this study, the performance differences between the models were not substantial enough to justify selecting one over another based solely on accuracy. Consequently, other factors, such as ease of use and optimization, preprocessing requirements, sample size needs, computational efficiency, and interpretability, were taken into account. Considering these criteria, tree-based methods such as RF and XGB are preferable for their interpretability, ease of use, and relatively faster computation. Thefore, the study proceeds with the XGB model for further optimization.

Another insight into the performance of the XGB model can be derived from Figure 3.10. In this figure, XGB predictions are compared with two traditional methods: linear regression using Python and the LINEST function from Microsoft Excel, based on the R^2 metric. Both of these statistical methods attempt to fit a linear four-dimensional geometry (three inputs and one output) into the design space, minimizing error by evaluating the distance to given points within the space. As a result, such models can be used to predict or interpolate unknown values. However, when examining the figure, there is only a minor difference between linear regression and LINEST. This

is because these built-in functions use different hyperparameters, leading to LINEST performing slightly better. On the other hand, the overall results indicate that the XGB machine learning model outperforms both of these traditional, relatively basic methods, especially in predicting outputs such as Young's Modulus and energy absorption. In the latter case, XGB achieves an R^2 score of 0.98, while linear regression and LINEST obtain scores of 0.79 and 0.8, respectively. It is important to note that accurately predicting the energy absorption properties of a lattice structure with minimal error is crucial for designing it for related applications, and from this perspective, the performance boost provided by XGB is significantly valuable.

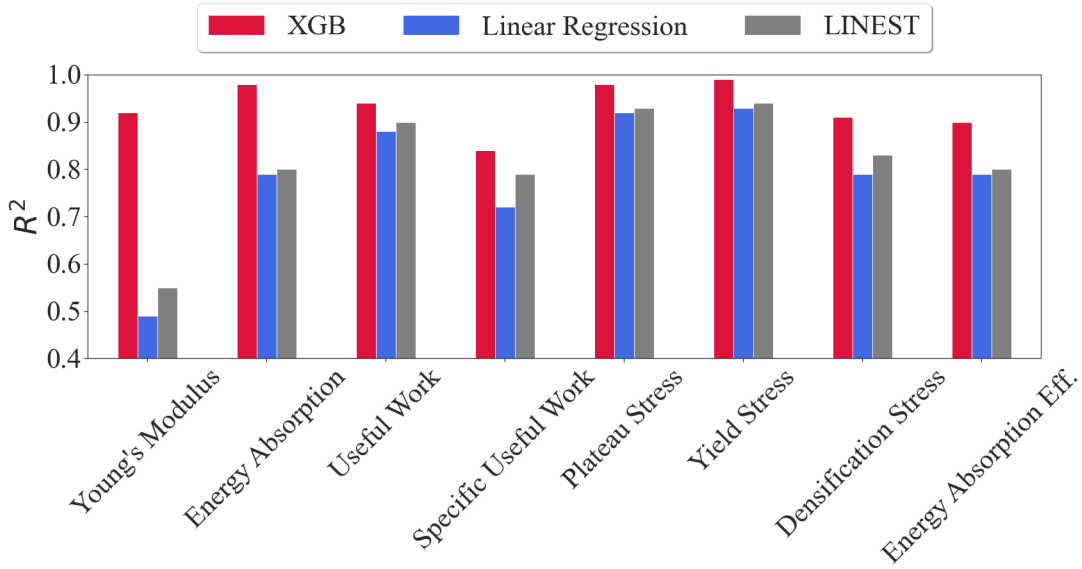


Figure 3.10: Performance comparison of XGB with two traditional linear regression methods

Figure 3.11 illustrates the learning curves of XGB for different outputs. In this figure, the y-axis represents the number of samples, while the x-axis represents the performance metric of the model. The R^2 score is used as the metric, calculated using 10-fold cross-validation, consistent with the method applied during hyperparameter optimization. From the figure, it is observed that while the performance gap between training and test scores diminishes toward the end of the dataset, outputs with a significant remaining gap, such as densification stress or specific useful work, exhibit relatively lower prediction performance. Additionally, it appears that convergence has not yet been reached for these outputs, suggesting that additional data may be beneficial to narrow the gap between training and test performance and to achieve convergence. In contrast, well-performing outputs, such as energy absorption (EA) or yield stress, demonstrate a different trend. They converge to a point with a minimal gap between training and test scores, indicating a satisfactory training process. This suggests that the predictions for these outputs were notably accurate by the end of the training, and no further data is required for their improvement [100].

Overall, the learning curves demonstrate no apparent tendency towards overfitting in any of the outputs. Overfitting is typically characterized by a marked divergence between training and testing scores, indicating that the model is excessively fitting the training data while performing poorly on the testing data, thus exhibiting suboptimal performance characteristics [101]. In such cases, different hyperparameters, including higher regularization strength, could be employed to mitigate overfitting.

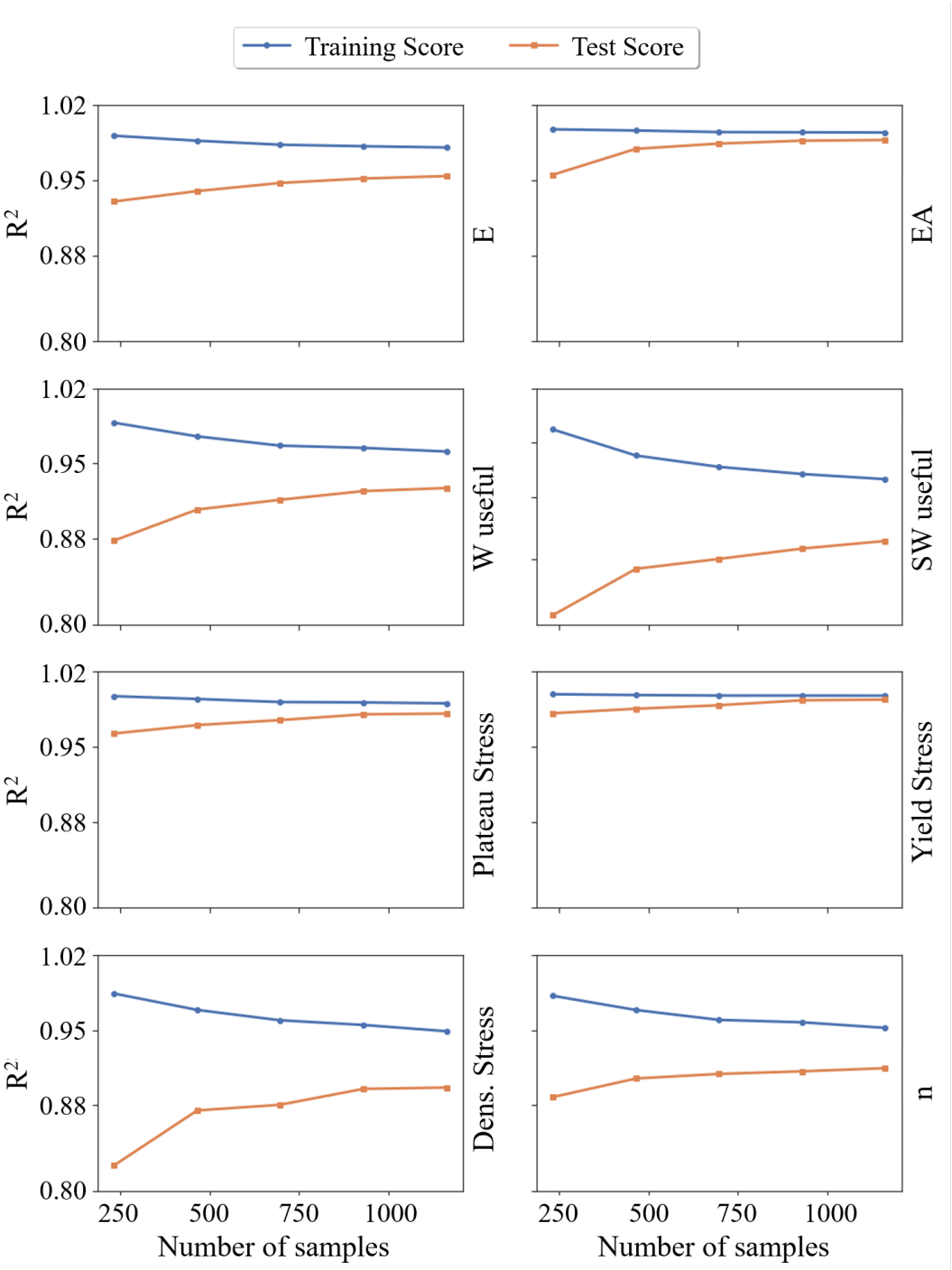


Figure 3.11: Learning curves of XGB

The interpretability of the XGB model is illustrated in Figure 3.12. XGB allows for the extraction of feature importances, revealing which parameters most significantly influence the desired property. This capability helps mitigate the *black-box* nature of the model. The feature importance scores are based on the *gain* metric, which indicates the relative contribution of each feature to the model. A higher gain value indicates a more significant impact on the prediction. From Figure 3.12, it can be observed that, overall, thickness is the most important parameter, except in the case of energy absorption efficiency, where *py* emerges as the most critical parameter. In contrast, the least important parameter is *px*.

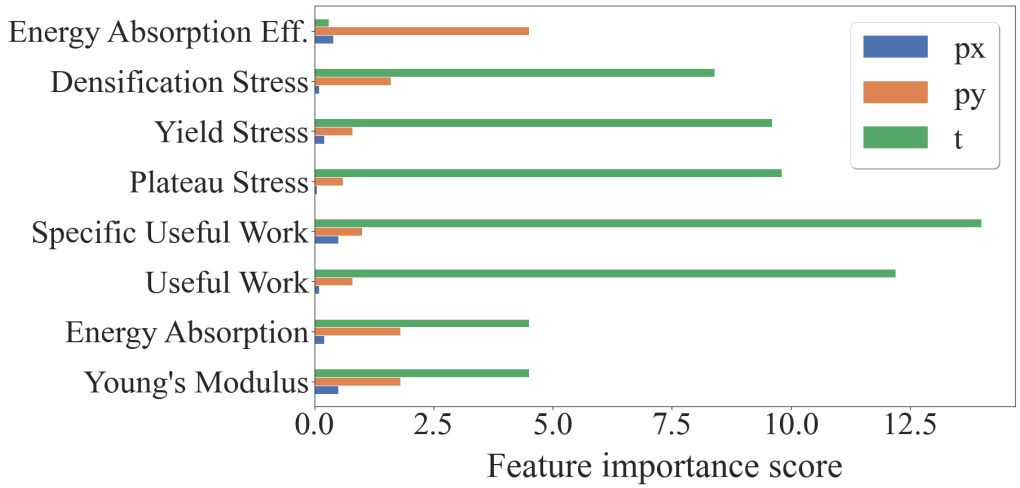


Figure 3.12: Feature importances for different properties provided by XGB

Achieving sufficient results with machine learning often requires considerable effort before training. Essential preprocessing steps, such as removing outliers, scaling data, and optimizing hyperparameters, were demonstrated in this section. Once these steps were completed, the models achieved sufficient accuracy, making them ready for deployment and prediction tasks. These predictions can be utilized independently, such as in design for additive manufacturing (DfAM) to estimate manufacturing costs or part quality [79], or as part of a larger framework, including integration into optimization processes discussed later in this study.

3.5 Optimization with Genetic Algorithm

Optimization is a fundamental aspect of engineering, focused on finding the most optimal or near-optimal solutions to complex problems. In this study, a genetic algorithm (GA) is employed alongside machine learning to identify these optimal solutions. GA serves as both a research and optimization tool, applicable to a broad range of problems [102]. It utilizes a computational approach to achieve its objectives, searching for optimal solutions within a candidate population [103]. However, it does not guarantee the discovery of the absolute best solution. Therefore, it can be categorized as a heuristic method.

In this study, the trained ML model serves as the fitness function within the GA framework. Specifically, candidate solutions are evaluated by the ML model, which provides predictions used as fitness scores. These scores help identify the best solutions in the population for the given objective.

Algorithm 2 Basic Genetic Algorithm

1: **begin**
2: *generations* $\leftarrow$ 600
3: *population size* $\leftarrow$ 12
4: *count* $\leftarrow$ 0
5: Set boundary values for solutions
6: Initialize a population
7: **while** $count < generations$ **do**
8: **calculate** fitness
9: **select** parents from the population
10: **perform** crossover and mutation
11: **update** the population with the new generation
12: *count* $\leftarrow$ *count* + 1
13: **end while**
14: Give the best individual
15: **end**

The input parameters are the features *px*, *py*, and *t*, with the objective of maximizing energy absorption efficiency (*n*) and having approximately a given plateau stress of 40 MPa is set for the GA. This process can be simplified as shown below,

$$\begin{aligned} &\textit{Maximize} \quad n\,(\%) \\ &\textit{Subject to} \quad \sigma_{\text{plateau}} \approx \textit{ 40 MPa} \\ &\textit{Variables} \quad p_x,\ p_y,\ t \end{aligned}$$

The stopping criterion is based on number of generations which is set to 600. When the *count* number being equal to number of *generations*, the algorithm terminates. The number of parents selected per generation is set to 12, which each generation must satisfy. After setting these parameters, selected parents undergo mutation and crossover to generate a new population. This process continues in a loop until the convergence criterion is met, with each new generation's fitness score determined by the ML model. The related algorithm is outlined in Algorithm 2. Boundary values are set to range from 6 to 14 for *px* and *py* by considering constraints applied to data generation phase which can also serve as ease for manufacturing. For the other parameter *t* (thickness, mm) the range is set to 0.8 to 3.2.

After setting the values, the optimization process is conducted, and the identified optimal values are $px = 6.04$, $py = 8.66$, and $t = 1.63$, as shown in Figure [**?**]. In this figure, the axes denoted as *px* and *py* represent the coordinate points of two design parameters, while the dark blue tones indicate the accumulation of these design points across generations. It can be observed from the figure that, although *py* consistently remains around 8, *px* fluctuates between approximately 10 and 5, resulting in a significant peak region around 5, ultimately leading to the value of 6.04. On the right-hand side of the same figure, the optimum shape of the unit cell derived from this result is depicted.

From the optimization results, it is noted that the energy absorption efficiency value is 30.53% and the plateau stress is 39.95 MPa, as can be seen in [**?**]. The optimization results begin at zero and exhibit a sharp increase to higher values in the first iteration, which accounts for the steep slope at the beginning. Throughout the generations, it

can be indicated from the figure that, while the energy absorption efficiency stabilizes around 30%, the plateau stress exhibits greater fluctuations around 40 MPa. The reason for this fluctuation is a variable introduced to the genetic algorithm in this study, referred to as the *mutation rate*.

$$x_{i+1} = x_i + random(-1, 1) \times \textit{mutation rate} \tag{3.8}$$

Mutation rate adjusts the magnitude of change or mutation strength for each individual solution across generations, as shown in Equation 3.8. In this equation, x_i represents an individual value of a member in the current generation (e.g. *px*), while x_{i+1} represents the corresponding value in the next generation. The mutation is performed by adding a value between -1 and 1, multiplied by the mutation rate, to the current solution. As a result, a higher mutation rate allows for greater exploration of different values, which helps to avoid local minima. However, this also introduces fluctuations. In this study, the mutation rate was set to 1. Despite these fluctuations, the end result demonstrates that the genetic algorithm (GA) and machine learning (ML) operate effectively to maintain the plateau stress at approximately 40 MPa, as intended.

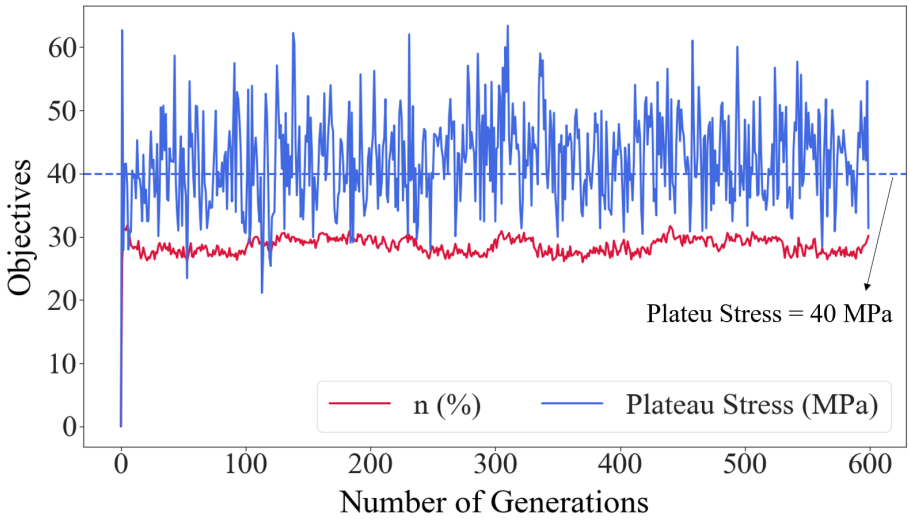


Figure 3.13: Objectives history of GA and ML integration include maximizing energy absorption efficiency (n) and having plateau stress close to 40 MPa. Both values are predicted by using the trained ML model, XGB.

The optimization results of $px = 6.04$, $py = 8.66$, and $t = 1.63$ align with the study's findings, which indicate that plateau stress is highly correlated with thickness. Consequently, the relatively low thickness value of 1.63 mm maintains the plateau stress around 40 MPa, which is significantly lower than the maximum thickness of 3.2 mm. Additionally, *py* exhibits a subtle positive correlation with plateau stress while demonstrating a negative correlation with energy absorption efficiency. This relationship explains why a *py* value of 8.66 helps to achieve an almost 39.95 MPa plateau stress, albeit at the cost of energy absorption efficiency, which is approximately 30.52%. Lastly, although *px* does not correlate with plateau stress, it appears to be optimized in relation to energy absorption efficiency concerning its boundary values. In other words, it reflects its lateral distance from the center of the unit cell, resulting in a value of 6.04. This explains the right-side bias observed in the *px* value in Figure 3.14.

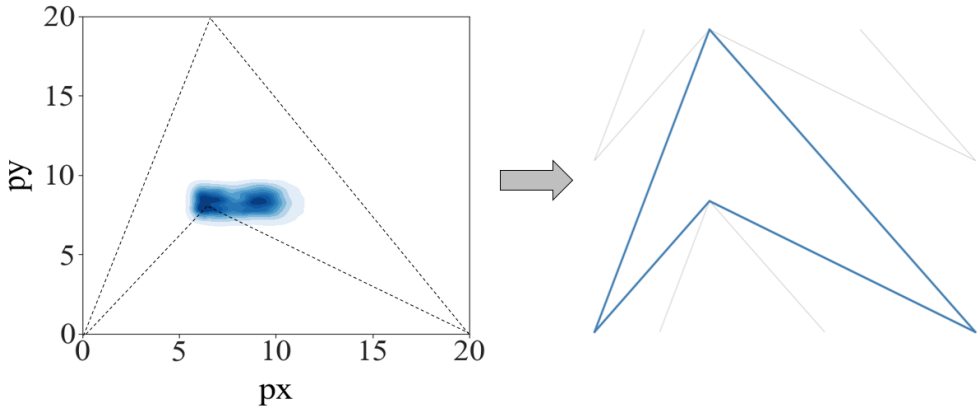


Figure 3.14: Optimum shape of the unit-cell given by GA and ML integration which is $px = 6.04$, $py = 8.66$, and $t = 1.63$. The blue color tones in the figure indicate the accumulation of design points at specific locations throughout the generations.

Applying the specified parameters in the finite element analysis (FEA) environment produces the model shown in Figure 3.15. The same boundary conditions and material model are applied to this optimized model to compare the predicted plateau stress and energy absorption values from the genetic algorithm (GA) with those obtained from the FEA results. This comparison indicates that while the GA yields an energy absorption efficiency of 30.53% and a plateau stress of 39.95 MPa, the FEA

provides values of 32.56% and 43.44 MPa for these properties, resulting in errors of 6.23% and 8.05%, respectively. As error values below 10% are generally satisfactory, these results are considered adequate. Further reduction in error may be achievable by focusing the optimization on a single objective.

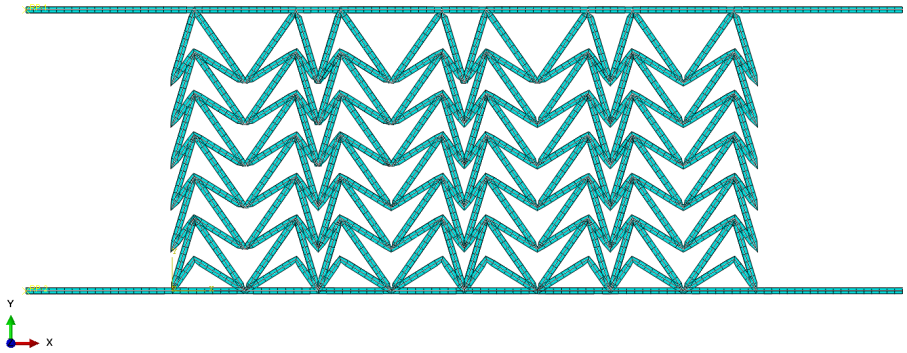
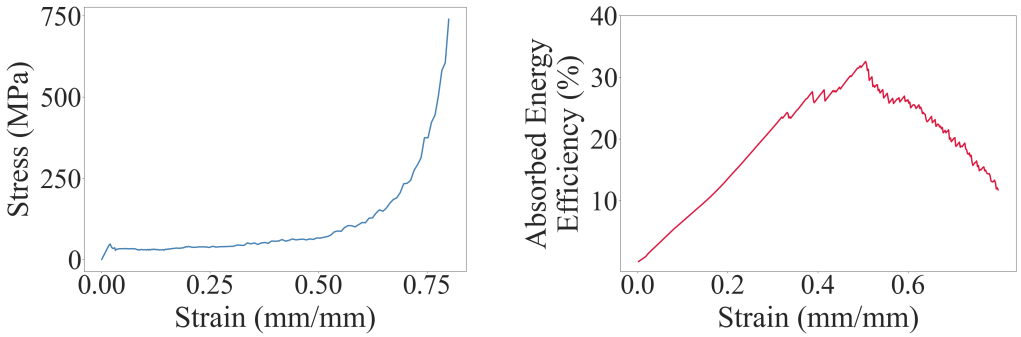


Figure 3.15: FEA model of the optimum result ($px = 6.04$, $py = 8.66$, $t = 1.63$) given by the GA and ML.



(a) (b)

Figure 3.16: Plots of the same optimum structure ($px = 6.04$, $py = 8.66$, $t = 1.63$), (a) stress-strain plot (b) energy absorption efficiency plot

Since the ML model was employed as the provider of fitness scores in this context, the entire optimization process required approximately 2465 seconds for 600 generations, with each prediction taking less than 1 second. In contrast, a single simulation in this study took about 780 seconds. It is also important to note that each individual in the population must have its fitness score assessed, resulting in a total of 7200 fitness evaluations for the GA (600 generations multiplied by a population size of 12).

If FEA had been used to evaluate these fitness scores, it would have required approximately 5616000 seconds, or 1560 hours, or 65 days. It is worth noting that these times may vary depending on the hardware used. Moreover, while FEA provides various data about the structure, such as stress or displacement, obtaining a property like energy absorption efficiency necessitates additional calculation steps. This comparative estimate is illustrated in Table 3.3. As demonstrated, integrating the ML model as the fitness function estimator significantly accelerated the optimization process. Although an analytical function could be utilized for the fitness function, it may lack the complexity necessary to accurately capture the real behavior of the problem and manage the numerous variables required in more complex scenarios than those presented in this study. Additionally, in terms of speed, analytical formulations might not provide any advantage over ML models after the training phase. Therefore, the efficiency and capability of ML models to capture complexity in a time-efficient manner make them valuable tools for optimization tasks, as evidenced by this study.

Table 3.3: Comparison of different approaches in terms of tasks and the estimated time required to perform them.

Approach	**Task**	**Time (seconds)**
ML	One property estimation	<1
FEA	One property estimation	780
ML and GA	Structural optimization	2465
FEA and GA	Structural optimization	5616000

It can be concluded that the machine learning model was successfully integrated with the genetic algorithm, providing essential fitness scores required for the optimization process. This integration demonstrates the model's capability to be effectively utilized within other pipelines. Moreover, it has the potential to substantially reduce computation time, as evidenced in this study.

CHAPTER 4

CONCLUSION

4.1 Summary of the Key Findings

In this study, the highly nonlinear mechanical behavior of auxetic lattice structures is investigated. To facilitate this investigation, a double-arrowhead unit cell is selected as the fundamental building block for the lattice structures. These structures were constructed by replicating the unit cell in a two-dimensional array with dimensions of 6 cells in width and 8 cells in height. A total of 1401 models were digitally generated and analyzed using finite element analysis (FEA). These models incorporated material, geometric, and contact non-linearities, rendering analytical formulation of the problem extremely difficult. Consequently, to predict the outcomes, a state-of-the-art data-driven technique, machine learning, was employed.

In the exploration of the data which is generated through finite element analysis it is seen that *px*, it appeared insignificant across all eight predicted properties. However, its lateral distance from the middle of the unit cell, *px'*, demonstrated a positive correlation with energy absorption efficiency and modulus of elasticity, and a negative correlation with energy absorption and yield strain, as indicated by the correlation matrix. On the other hand *py* showed a subtle relationship with plateau stress and useful work whereas strong negative correlation with energy absorption efficiency observed for it. Lastly, *t* showed positive correlations with most properties with exception of densification strain and energy absorption efficiency. These findings suggest that, maximize the energy absorption efficiency of a lattice structure with double-arrow head, a low *py* value and a *px* value further from the middle should be used regardless of the thickness. Additionally, the correlation among the data revealed that stress-

related properties are positively aligned with energy absorption characteristics of the structure with the exception of energy absorption efficiency. Moreover, a Pareto set and a corresponding Pareto fit curve were provided to illustrate the optimum design points for the multi-objective optimization task, which seeks to maximize useful work while minimizing plateau stress and mass.

After data exploration, outlier removal, and scaling were applied, three machine learning models were optimized through hyperparameter tuning on the training dataset. The dataset was divided into training and test sets, with 10% reserved for testing. The machine learning models utilized three features (*px*, *py*, and *t*) to predict eight mechanical properties of the structure, making this a regression task. The models, including the neural network, random forest, and extreme gradient boosting, achieved an average R^2 score of 0.936 and a mean absolute percentage error (MAPE) of 1.775 on the test dataset, which was held out until this evaluation point, indicating high prediction performance. It was observed that the lower prediction accuracy for higher values of plateau stress and useful work is due to sparse data in these high-value regions, leading to less accurate pattern capture by the machine learning models. However, the overall impact of these high values on accuracy was minimal. Although the random forest model performed best on average, the prediction metrics among the models were closely aligned. This minor performance difference may be due to variations in hyperparameter optimization. Consequently, extreme gradient boosting was selected for the optimization task. Additionally, the learning curves of extreme gradient boosting revealed notable R^2 score differences between training and test scores for certain outputs, such as densification stress and specific useful work, suggesting that these targets could benefit from more training data. No significant overfitting tendency was observed from the same curves. The interpretative capabilities of extreme gradient boosting also indicated that *t* (thickness) is the most important feature, being effective for seven out of eight outputs, followed by *py*. For the remaining output, *py* was identified as the most crucial parameter, specifically for energy absorption efficiency, although it negatively impacts this property.

The trained Extreme Gradient Boosting (XGB) model was employed to provide the fitness score, while a genetic algorithm was utilized as the optimizer, demonstrating the feasibility of using machine learning models to predict the mechanical properties

of lattice structures and optimize their design accordingly. This combined approach of machine learning and genetic algorithms was applied to identify a double arrow-head lattice structure, with the objective of maximizing energy absorption efficiency, while constraining the plateau stress to approximately 40 MPa. The optimization process, which spanned 600 generations and took 2,465 seconds, yielded an estimated energy absorption efficiency of 30.53% and a plateau stress value of 39.95 MPa. These results correspond to error percentages of 6.23% and 8.05%, respectively, when compared to the finite element analysis (FEA) solution. The optimization results align well with the trends observed in the data, further demonstrating the effectiveness of integrating machine learning with genetic algorithms for optimization tasks.

Lastly, it should be noted that although this study has demonstrated the methodology of training ML models using FEA solutions with three variables, the same approach could be applied to problems with significantly more variables and may also extend to other types of engineering challenges with similar methodologies. Therefore, this data-driven approach, utilizing machine learning, has the potential to open new avenues for addressing complex engineering problems.

4.2 Suggestions for Future Works

As is often the case in science and engineering, there are still opportunities for improvement in this work and related topics. Potential enhancements include, but are not limited to:

- This study exclusively uses the double arrowhead unit cell geometry, with no consideration of other geometries. As the machine learning model is trained solely on this structure, its predictions cannot be generalized to other lattice geometries. However, by including different unit cell types in the training data, the model could learn to recognize and predict across various structures.
- Similarly, the study is limited in predicting results for different materials, as it was trained with only one material. Incorporating a range of materials as variables could be considered for future research and applications.

- Another limitation of this study is the absence of damage modeling. The finite element analysis employed in this work considers only plastic behavior with strain hardening, without accounting for any damage occurrence. Future studies may overcome this limitation by incorporating damage modeling to capture material degradation under loading.

- The accuracy of estimator predictions can be further enhanced by adopting alternative approaches, such as utilizing an ensemble of multiple machine learning models.

- The number of samples required to train the model can and should be reduced through novel approaches, as achieving accurate results with less data is a critical challenge in machine learning. Semi-supervised learning may be applicable to the problems addressed in this study. While it may not reduce the entire dataset size, it can decrease the need for labeled data, representing a significant improvement.

- Unstructured data types, such as images of lattice structures, could be utilized instead of the structured (tabular) data employed in this study. Such data might enable neural network to outperform tree-based algorithms, such as extreme gradient boosting or random forest. Additionally, this approach could offer new insights into mechanical property prediction and other related areas.

It is hoped that implementing these improvements and suggestions, along with the latest machine learning techniques, could advance the field by enabling the design of more efficient structures and contribute to the benefit of humanity.

REFERENCES

[1] B.K. Nagesha, V. Dhinakaran, M. Varsha Shree, K.P. Manoj Kumar, Damodar Chalawadi, and T. Sathish. Review on characterization and impacts of the lattice structure in additive manufacturing. *Materials Today: Proceedings*, 21:916–919, 2020. International Conference on Recent Trends in Nanomaterials for Energy, Environmental and Engineering Applications.

[2] Chen Pan, Yafeng Han, and Jiping Lu. Design and optimization of lattice structures: A review. *Applied Sciences*, 10(18), 2020.

[3] Longfei Zhou, Jenna Miller, Jeremiah Vezza, Maksim Mayster, Muhammad Raffay, Quentin Justice, Zainab Al Tamimi, Gavyn Hansotte, Lavanya Devi Sunkara, and Jessica Bernat. Additive manufacturing: A comprehensive review. *Sensors*, 24, 2024.

[4] Mostafa Yakouta, M. A. Elbestawib, and Stephen C. Veldhuisc. A review of metal additive manufacturing technologies. *Solid State Phenomena*, 278:1–14, 2018.

[5] Kaufui V. Wong and Aldo Hernandez. A review of additive manufacturing. *International Scholarly Research Notices*, 2012(1):208760, 2012.

[6] Liang-Yu Chen, Shun-Xing Liang, Yujing Liu, and Lai-Chang Zhang. Additive manufacturing of metallic lattice structures: Unconstrained design, accurate fabrication, fascinated performances, and challenges. *Materials Science and Engineering: R: Reports*, 146:100648, 2021.

[7] Tobias Maconachie, Martin Leary, Bill Lozanovski, Xuezhe Zhang, Ma Qian, Omar Faruque, and Milan Brandt. Slm lattice structures: Properties, performance, applications and challenges. *Materials Design*, 183:108137, 2019.

[8] Wenjin Tao and Ming C. Leu. Design of lattice structure for additive manufacturing. In *2016 International Symposium on Flexible Automation (ISFA)*, pages 325–332, 2016.

[9] Zana Eren, Ozkan Gokcekaya, Takayoshi Nakano, and Zahit Mecitoğlu. In-plane quasi-static compression deformation of ti6al4v double arrow-headed lattice structures fabricated by electron beam powder bed fusion process: Build orientation, scan speed and failure mechanism. *Journal of Materials Research and Technology*, 27:6192–6210, 2023.

[10] Anthony P. Garland, Katarina M. Adstedt, Zachary J. Casias, Benjamin C. White, William M. Mook, Bryan Kaehr, Bradley H. Jared, Brian T. Lester, Nicholas S. Leathe, Eric Schwaller, and Brad L. Boyce. Coulombic friction in metamaterials to dissipate mechanical energy. *Extreme Mechanics Letters*, 40:100847, 2020.

[11] Jochen Mueller and Kristina Shea. Stepwise graded struts for maximizing energy absorption in lattices. *Extreme Mechanics Letters*, 25:7–15, 2018.

[12] Xiaoyu Zheng, Howon Lee, Todd H. Weisgraber, Maxim Shusteff, Joshua DeOtte, Eric B. Duoss, Joshua D. Kuntz, Monika M. Biener, Qi Ge, Julie A. Jackson, Sergei O. Kucheyev, Nicholas X. Fang, and Christopher M. Spadaccini. Ultralight, ultrastiff mechanical metamaterials. *Science*, 344(6190):1373–1377, 2014.

[13] Claus Claeys, Noé Geraldo Rocha de Melo Filho, Lucas Van Belle, Elke Deckers, and Wim Desmet. Design and validation of metamaterials for multiple structural stop bands in waveguides. *Extreme Mechanics Letters*, 12:7–22, 2017. Frontiers in Mechanical Metamaterials.

[14] Jiping Lu Chen Pan, Yagenf Han. Additive manufacturing - a review. *Apllied Sciences*, 2021.

[15] M.F. Ashby. The properties of foams and lattices. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 364(1838):15–30, Jan 15 2006.

[16] Hanfeng Yin, Wenzheng Zhang, Liangcong Zhu, Fanbo Meng, Jie Liu, and Guilin Wen. Review on lattice structures for energy absorption properties. *Composite Structures*, 304:116397, 2023.

[17] Z.P. Sun, Y.B. Guo, and V.P.W. Shim. Characterisation and modeling of additively-manufactured polymeric hybrid lattice structures for energy absorption. *International Journal of Mechanical Sciences*, 191:106101, 2021.

[18] Xiaofei Cao, Shengyu Duan, Jun Liang, Weibin Wen, and Daining Fang. Mechanical properties of an improved 3d-printed rhombic dodecahedron stainless steel lattice structure of variable cross section. *International Journal of Mechanical Sciences*, 145:53–63, 2018.

[19] F.N. Habib, P. Iovenitti, S.H. Masood, and M. Nikzad. Fabrication of polymeric lattice structures for optimum energy absorption using multi jet fusion technology. *Materials Design*, 155:86–98, 2018.

[20] Simon R.G. Bates, Ian R. Farrow, and Richard S. Trask. 3d printed polyurethane honeycombs for repeated tailored energy absorption. *Materials Design*, 112:172–183, 2016.

[21] Kyung-Sung Min Kwang-Min Park and Young-Sook Roh. Design optimization of lattice structures under compression: Study of unit cell types and cell arrangements. *Materials*, 15, 2021.

[22] Ehsan Etemadi, Mahbubeh Hosseinabadi, Mohsen Taghizadeh, Fabrizio Scarpa, and Hong Hu. Enhancing the energy absorption capability of auxetic metamaterials through auxetic cells within re-entrant circular units. *Engineering Structures*, 315:118379, 2024.

[23] Chang Qi, Feng Jiang, Shu Yang, Alex Remennikov, Shang Chen, and Chen Ding. Dynamic crushing response of novel re-entrant circular auxetic honeycombs: Numerical simulation and theoretical analysis. *Aerospace Science and Technology*, 124:107548, 2022.

[24] Xiaolin Deng and Shangan Qin. In-plane energy absorption characteristics and mechanical properties of novel re-entrant honeycombs. *Composite Structures*, 313:116951, 2023.

[25] Lulu Wei, Xuan Zhao, Qiang Yu, and Guohua Zhu. A novel star auxetic honeycomb with enhanced in-plane crushing strength. *Thin-Walled Structures*, 149:106623, 2020.

[26] Dianwei Gao, Shuhong Wang, Mingzhong Zhang, and Chunwei Zhang. Experimental and numerical investigation on in-plane impact behaviour of chiral auxetic structure. *Composite Structures*, 267:113922, 2021.

[27] Tao Wang, Zhen Li, Liangmo Wang, Xianfeng Zhang, and Zhengdong Ma. In-plane elasticity of a novel arcwall-based double-arrowed auxetic honeycomb design: Energy-based theoretical analysis and simulation. *Aerospace Science and Technology*, 127:107715, 2022.

[28] Qiang Gao, Wei-Hsin Liao, and Liangmo Wang. On the low-velocity impact responses of auxetic double arrowed honeycomb. *Aerospace Science and Technology*, 98:105698, 2020.

[29] J.B. Choi and R.S. Lakes. Non-linear properties of metallic cellular materials with a negative poisson's ratio. *Journal of Materials Science*, 27(19):5375 – 5381, 1992.

[30] Roderic Lakes. Foam structures with a negative poisson's ratio. *Science*, 235(4792):1038–1040, 1987.

[31] Chinmay Kumar Kundu Lovely Sabat. History of finite element method: A review. *15th International Conference on ICT and Knowledge Engineering (ICTKE)*, pages 395–404, 2021.

[32] MathWorks. What is finite element analysis?, 2024. `https://it.mathworks.com/discovery/finite-element-analysis.html`.

[33] Tejal Pore, Sandeep G. Thorat, and Archana A. Nema. Review of contact modelling in nonlinear finite element analysis. *Materials Today: Proceedings*, 47:2436–2440, 2021. International Conference on Materials and System Engineering.

[34] David Roylance. Finite element analysis. *Department of Materials Science and Engineering - Massachusetts Institute of Technology, Cambridge*, 2001.

[35] M. I. Jordan and T. M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.

[36] Hooman H. Rashidi, Nam K. Tran, Elham Vali Betts, Lydia P. Howell, and Ralph Green. Artificial intelligence and machine learning in pathology: The present landscape of supervised methods. *Academic Pathology*, 6:2374289519873088, 2019. PMID: 31523704.

[37] Haochen Hua, Yutong Li, Tonghe Wang, Nanqing Dong, Wei Li, and Junwei Cao. Edge computing with artificial intelligence: A machine learning perspective. *ACM Comput. Surv.*, 55(9), jan 2023.

[38] Iqbal H. Sarker. Machine learning: Algorithms, real-world applications and research directions. *SN Computer Science*, 2(3):160, 2021.

[39] Yann LeCun, Y. Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–44, 05 2015.

[40] Batta Mahesh. Machine learning algorithms - a review. *International Journal of Science and Research*, 9, 2020.

[41] Pedro Domingos. A few useful things to know about machine learning. *Commun. ACM*, 55(10):78–87, oct 2012.

[42] Pariwat Ongsulee. History of finite element method: A review. *15th International Conference on ICT and Knowledge Engineering (ICTKE)*, 2017.

[43] Tarleton Gillespie. *The Relevance of Algorithms*. The MIT Press, 01 2014.

[44] Sebastian Raschka and Vahid Mirjalili. *Python Machine Learning, 2nd Ed.* Packt Publishing, 2 edition, 2017.

[45] Shagan Sah. Machine learning: A review of learning types. *Preprints*, 07 2020.

[46] Yile Zhang and Yadong Zhou. Investigation of bird-strike resistance of composite sandwich curved plates with lattice/foam cores. *Thin-Walled Structures*, 182:110203, 2023.

[47] Yu Chen, Ming-Hui Fu, Hong Hu, and Jian Xiong. Curved inserts in auxetic honeycomb for property enhancement and design flexibility. *Composite Structures*, 280:114892, 2022.

[48] L.L. Hu, M.Zh. Zhou, and H. Deng. Dynamic crushing response of auxetic honeycombs under large deformation: Theoretical analysis and numerical simulation. *Thin-Walled Structures*, 131:373–384, 2018.

[49] Kaustav Bora, Shailendra Varshney, and Cheruvu Kumar. Non local analytical and numerical modelling of re-entrant auxetic honeycomb. *Engineering Research Express*, 6, 05 2024.

[50] Wenwang Wu, Wenxia Hu, Guian Qian, Haitao Liao, Xiaoying Xu, and Filippo Berto. Mechanical design and multifunctional applications of chiral mechanical metamaterials: A review. *Materials Design*, 180:107950, 2019.

[51] Tao Wang, Zhen Li, Liangmo Wang, Xianfeng Zhang, and Zhengdong Ma. In-plane elasticity of a novel arcwall-based double-arrowed auxetic honeycomb design: Energy-based theoretical analysis and simulation. *Aerospace Science and Technology*, 127:107715, 2022.

[52] Qiang Gao, Chengqiang Ge, Weichao Zhuang, Liangmo Wang, and Zhengdong Ma. Crashworthiness analysis of double-arrowed auxetic structure under axial impact loading. *Materials Design*, 161:22–34, 2019.

[53] Yun-Long Chen, Xin-Tao Wang, and Li Ma. Damping mechanisms of cfrp three-dimensional double-arrow-head auxetic metamaterials. *Polymer Testing*, 81:106189, 2020.

[54] Zeyao Chen, Zhe Wang, Shiwei Zhou, Jianwang Shao, and Xian Wu. Novel negative poisson's ratio lattice structures with enhanced stiffness and energy absorption capacity. *Materials*, 11(7), 2018.

[55] M. Smith, Z. Guan, and W.J. Cantwell. Finite element modelling of the compressive response of lattice structures manufactured using the selective laser melting technique. *International Journal of Mechanical Sciences*, 67:28–41, 2013.

[56] Erhan Cetin and Cengiz Baykasoğlu. Energy absorption of thin-walled tubes enhanced by lattice structures. *International Journal of Mechanical Sciences*, 157-158:471–484, 2019.

[57] Hany Hassanin, Yusra Alkendi, Mahmoud Elsayed, Khamis Essa, and Yahya Zweiri. Controlling the properties of additively manufactured cellular structures using machine learning approaches. *Advanced Engineering Materials*, 22(3):1901338, 2020.

[58] Guoji Yu, Lijun Xiao, and Weidong Song. Deep learning-based heterogeneous strategy for customizing responses of lattice structures. *International Journal of Mechanical Sciences*, 229:107531, 2022.

[59] Sayyeda Saadia Razvi, Shaw Feng, Anantha Narayanan, Yung-Tsun Lee, and Paul Witherell. A review of machine learning applications in additive manufacturing. 08 2019.

[60] Sangryun Lee, Zhizhou Zhang, and Grace X. Gu. Generative machine learning algorithm for lattice structures with superior mechanical properties. *Mater. Horiz.*, 9:952–960, 2022.

[61] Hyeongkeun Kim, Sameh H. Tawfick, and William P. King. Modeling and design of zero-stiffness elastomer springs using machine learning. *Advanced Intelligent Systems*, 4(12):2200225, 2022.

[62] Lingbin Meng, Brandon McWilliams, William Jarosinski, Hye-Yeong Park, Yeon-Gil Jung, Jehyun Lee, and Jing Zhang. Machine learning in additive manufacturing: A review. *JOM*, 72(6):2363–2377, June 2020.

[63] Dean Grierson, Allan Rennie, and Stephen Quayle. Machine learning for additive manufacturing. *Encyclopedia*, 1:576–588, 07 2021.

[64] Jianjing Zhang, Peng Wang, and Robert X. Gao. Deep learning-based tensile strength prediction in fused deposition modeling. *Computers in Industry*, 107:11–21, 2019.

[65] Karthic Manoharan, K. Chockalingam, and S. Shankar Ram. Prediction of tensile strength in fused deposition modeling process using artificial neural network technique. *AIP Conference Proceedings*, 2311(1):080012, 12 2020.

[66] Hyeongkeun Kim, Sameh Tawfick, and William King. Modeling and design of zero-stiffness elastomer springs using machine learning. *Advanced Intelligent Systems*, 4:2200225, 11 2022.

[67] B. Veera Siva Reddy, Ameer Malik Shaik, C. Chandrasekhara Sastry, J. Krishnaiah, Chirag Anil Bhise, and B. Ramakrishna. Machine learning approaches for predicting mechanical properties in additive manufactured lattice structures. *Materials Today Communications*, page 109937, 2024.

[68] Yirun Wu, Zhongfa Mao, and Yiqing Feng. Energy absorption prediction for lattice structure based on d2 shape distribution and machine learning. *Composite Structures*, 319:117136, 2023.

[69] Shuai Ma, Qian Tang, Ying Liu, and Qixiang Feng. Prediction of mechanical properties of three-dimensional printed lattice structures through machine learning. *Journal of Computing and Information Science in Engineering*, 22, 06 2022.

[70] Aldair E. Gongora, Caleb Friedman, Deirdre K. Newton, Timothy D. Yee, Zachary Doorenbos, Brian Giera, Eric B. Duoss, Thomas Y.-J. Han, Kyle Sullivan, and Jennifer N. Rodriguez. Accelerating the design of lattice structures using machine learning. *Scientific Reports*, 14(1):13703, 2024.

[71] Adithya Challapalli and Guoqiang Li. Machine learning assisted design of new lattice core for sandwich structures with superior load carrying capacity. *Scientific Reports*, 11(1):18552, 2021.

[72] Jinqi Shang, Kangkang Wang, Dongyang Yan, Fengrui Liu, Linjuan Wang, and Libin Zhao. An analytical method for elastic modulus of the sandwich bcc lattice structure based on assumption of linear distribution. *Materials*, 16(9), 2023.

[73] Rafael Guerra Silva, Cristóbal Salinas Estay, Gustavo Morales Pavez, María J. Torres, and Jorge Zahr Viñuela. Assessment of analytical relationships for mechanical properties of truncated octahedron and diamond lattice structures. *Materials Today Communications*, 29:102756, 2021.

[74] Dassault Systèmes Simulia Corp. Johnson-cook plasticity, 2022. https://classes.engineering.wustl.edu/2009/spring/mase5513/abaqus/docs/v6.6/books/usb/default.htm?startat=pt05ch18s02abm21.html.

[75] Usama Idrees, Sajjad Ahmad, Imtiaz Alam Shah, Muhammad Talha, Rehman Shehzad, Muhammad Amjad, and Seyed Saeid Rahiamin Koloor. Finite element analysis of car frame frontal crash using lightweight materials. *Journal of Engineering Research*, 11(1):100007, 2023.

[76] Xuan Zhao, Qiang Gao, Liangmo Wang, Qiang Yu, and Z.D. Ma. Dynamic crushing of double-arrowed auxetic structure under impact loading. *Materials Design*, 160:527–537, 2018.

[77] Hu Liu, Ee Teng Zhang, Guangjian Wang, and Bing Feng Ng. In-plane crushing behavior and energy absorption of a novel graded honeycomb from hierarchical architecture. *International Journal of Mechanical Sciences*, 221:107202, 2022.

[78] Xiaoqiang Niu, Fengxiang Xu, Zhen Zou, Tengyuan Fang, Suo Zhang, and Quanmin Xie. In-plane dynamic crashing behavior and energy absorption of novel bionic honeycomb structures. *Composite Structures*, 299:116064, 2022.

[79] Ugur M. Dilberoglu, Bahar Gharehpapagh, Ulas Yaman, and Melik Dolen. The role of additive manufacturing in the era of industry 4.0. *Procedia Manufacturing*, 11:545–554, 2017. 27th International Conference on Flexible Automation and Intelligent Manufacturing, FAIM2017, 27-30 June 2017, Modena, Italy.

[80] Kadir Günaydın, Craig Rea, and Zafer Kazancı. Energy absorption enhancement of additively manufactured hexagonal and re-entrant (auxetic) lattice structures by using multi-material reinforcements. *Additive Manufacturing*, 59:103076, 2022.

[81] Sri Maddukuri, Ravindra Ganta, and Siva Prasad A V S. Numerical study of energy absorption capabilities of periodic cellular structures. October 2023.

[82] Dassault Systèmes Simulia Corp. Matrix storage and solution scheme in abaqus/standard, 2024. https://docs.software.vt.

edu/abaqusv2024/English/?show=SIMACAEANLRefMap/simaanl-c-over.htm#simaanl-c-unsymm.

[83] Mehmet Kepenekci, Bahar Gharehpapagh, Ulas Yaman, and Sezer Özerinç. Mechanical performance of carbon fiber-reinforced polymer cellular structures manufactured via fused filament fabrication. *Polymer Composites*, 44(8):4654–4668, 2023.

[84] Stratos Idreos, Olga Papaemmanouil, and Surajit Chaudhuri. Overview of data exploration techniques. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, SIGMOD '15, page 277–281, New York, NY, USA, 2015. Association for Computing Machinery.

[85] Mingzhi Wang, Yinzhu Wang, Jianjun Wu, and Weidong Wang. Compression response and optimization design of a novel glass-sponge inspired lattice structure with enhanced energy absorption capacity. *Aerospace Science and Technology*, 155:109582, 2024.

[86] Ch. Sanjeev Kumar Dash, Ajit Kumar Behera, Satchidananda Dehuri, and Ashish Ghosh. An outliers detection and elimination framework in classification task of data mining. *Decision Analytics Journal*, 6:100164, 2023.

[87] Karankajrolkar. Why "1.5" in iqr method of outlier detection, 2021. https://medium.com/@karankajrolkar/why-1-5-in-iqr-method-of-outlier-detection-b9301a95c704.

[88] Scikit learn developers. 3.1. cross-validation: evaluating estimator performance, 2024. https://scikit-learn.org/stable/modules/cross_validation.html.

[89] K. Sheela and S N Deepa. Review on methods to fix number of hidden neurons in neural networks. *Mathematical Problems in Engineering*, 2013, 01 2013.

[90] Kostadin Yotov, Emil Hadzhikolev, and Stanka Hadzhikoleva. Determining the number of neurons in artificial neural networks for approximation, trained with algorithms using the jacobi matrix. *TEM Journal*, 9:1320–1329, 11 2020.

[91] Shuxiang Xu and L. Chen. A novel approach for determining the optimal number of hidden layer neurons for fnn's and its application in data mining. *5th International Conference on Information Technology and Applications, ICITA 2008*, pages 683–686, 06 2008.

[92] Thais Mayumi Oshiro, Pedro Santoro Perez, and José Augusto Baranauskas. How many trees in a random forest? In Petra Perner, editor, *Machine Learning and Data Mining in Pattern Recognition*, pages 154–168, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[93] Alicia Curth, Alan Jeffares, and Mihaela van der Schaar. Why do random forests work? understanding tree ensembles as self-regularizing adaptive smoothers, 2024.

[94] Publication-ready nn-architecture schematics, 2024. `https://alexlenail.me/NN-SVG/`.

[95] Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 35(6):7499–7519, June 2024.

[96] Ravid Shwartz-Ziv and Amitai Armon. Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90, 2022.

[97] Shubham Koli. Kaggle winners algorithm xgboost, 2023. `https://medium.com/@MrBam44/kaggle-winners-algorithm-xgboost-87819eb300ae`.

[98] Yuanchao Wang, Z. Pan, J. Zheng, L. Qian, and Li Mingtao. A hybrid ensemble method for pulsar candidate classification. *Astrophysics and Space Science*, 364, 08 2019.

[99] Shichao Jin, Qinghua Guo, Min Li, Qiuli Yang, Kexin Xu, Yuanzhen Ju, Jing Zhang, Jing Xuan, Yanjun Su, Qiang Xu, and Yu Liu. Application of deep learning in ecological resource research: Theories, methods, and challenges. *Science China Earth Science*, 03 2020.

[100] Scikit learn developers. Plotting learning curves and checking models' scalability, 2024. https://scikit-learn.org/stable/auto_examples/model_selection/plot_learning_curve.html.

[101] Defauw, Szoc, Bardadym, Brabers, Everaert, Mijic, Scholte, Vanallemeersch, Winckel, and Bogaert. Misalignment detection for web-scraped corpora: A supervised regression approach. *Informatics*, 6:35, 09 2019.

[102] Kalyanmoy Deb. An introduction to genetic algorithms. *Sadhana*, 24:293–315, 1999.

[103] M. Srinivas and L.M. Patnaik. Genetic algorithms: a survey. *Computer*, 27(6):17–26, 1994.

APPENDIX A

THE CODE USED IN THIS STUDY

The code or scripts used in to generate lattice structures, to perform data analysis, to train machine learning models, to conduct optimization with genetic algorithm and to plot various figures are shared in the below link:

https://github.com/ymnpy/LatticeStru_FEAvML

Some of the parts of the code are also presented below:

```
"""
THE MAIN SCRIPT FOR ABAQUS COMMANDS,
generating, solving, storing FEA results of lattice structures in a loop
"""

def findintersection(p1,p2,p3,p4):
    px=((p1[0]p2[1]p1[1]p2[0])(p3[0]p4[0])(p1[0]p2[0])(p3[0]p4[1]p3[1]p4[0]))
        /((p1[0]p2[0])(p3[1]p4[1])(p1[1]p2[1])(p3[0]p4[0])+1e6)
    py=((p1[0]p2[1]p1[1]p2[0])(p3[1]p4[1])(p1[1]p2[1])(p3[0]p4[1]p3[1]p4[0]))
        /((p1[0]p2[0])(p3[1]p4[1])(p1[1]p2[1])(p3[0]p4[0])+1e6)
    return [px,py]

def getprevdata():
    storage=[]
    count=0
    path=r"C:/temp/data.log"
```

```
    with open(path,"r") as fin:
        for line in fin.readlines():
            try:
                count+=1
                a=line.split(()[4].split())[0]
                x,y=float(a.split(,)[0]),float(a.split(,)[1])
                storage.append((x,y))
            except:
                pass
    return storage,count1



""" INPUTS """
now=time.time()
storage,count=getprevdata() residual aldigim yer
cellsx,cellsy=8,6
depthlength=4
print(storage)

offset=2
count=0
nomodels=1400

massscale=1
steptime=0.1
density=2.7e9
thicknesslist=[0.8,1.2,1.6,2.4,3.2]

jobrun=True
emptypass=0
```

```
if len(storage)==0:
    with open("data.log","w") as fout:
        fout.write("check out github for this")

while count = nomodels:
    r1x,r1y=0,0
    r2x,r2y=20,0
    r3x,r3y=round(random.random()20,1),20
    r4x,r4y=r3x,round(random.random()20,1)
    thickness=random.choice(thicknesslist)

    modelname=str(count)+"Al"
                    +"X"+str(r3x).replace(".","")
                    +"Y"+str(r4y).replace(".","")
                    +"T"+(str(thickness).replace(".",""))

    angle check for p4
    a13=(r3yr1y)/(r3xr1x+0.001)
    a23=(r3yr2y)/(r3xr2x+0.001)
    a14=(r4yr1y)/(r4xr1x+0.001)
    a24=(r4yr2y)/(r4xr2x+0.001)

    lengths of a cell
    l1=((r3xr1x)2+(r3yr1y)2)(0.5)
    l2=((r3xr2x)2+(r3yr2y)2)(0.5)
    l3=((r4xr2x)2+(r4yr2y)2)(0.5)
    l4=((r1xr4x)2+(r1yr4y)2)(0.5)
    lmin=min(l1,l2,l3,l4)
    meshauxetic=lmin/5

    p1,p2,p3,p4=(r1x,r1y),(r2x,r2y),(r3x,r3y),(r4x,r4y)
```

```
off=p3[1]p4[1]
p1top,p1bot=[p1[0],p1[1]+off],[p1[0],p1[1]off]
p2top,p2bot=[p2[0],p2[1]+off],[p2[0],p2[1]off]
p3top,p3bot=[p3[0],p3[1]+off],[p3[0],p3[1]off]
p4top,p4bot=[p4[0],p4[1]+off],[p4[0],p4[1]off]

t1top,t1bot=[0,20],[0,0]
t2top,t2bot=[20,20],[20,0]

pp=findintersection(p1top,p3top,t1top,t2top)
l5=((pp[0]p1top[0])2+(pp[1]p1top[1])2)0.5

pp=findintersection(p1top,p4top,t1top,t2top)
l6=((pp[0]p1top[0])2+(pp[1]p1top[1])2)0.5

pp=findintersection(p2top,p3top,t1top,t2top)
l7=((pp[0]p2top[0])2+(pp[1]p2top[1])2)0.5

pp=findintersection(p2top,p4top,t1top,t2top)
l8=((pp[0]p2top[0])2+(pp[1]p2top[1])2)0.5

pp=findintersection(p1bot,p3bot,t1bot,t2bot)
l9=((pp[0]p4[0])2+(pp[1]p4[1])2)0.5

pp=findintersection(p2bot,p3bot,t1bot,t2bot)
l10=((pp[0]p4[0])2+(pp[1]p4[1])2)0.5

solid volume frac / line density
solidvol=2020
latticevol=(l1+l2+l3+l4+l5+l6+l7+l8+l9+l10)thickness
svf=latticevol/solidvol
roeffective=svfdensity
```

thetas
theta1=180(degrees(atan(a13))+degrees(atan(a23)))
theta2=180(degrees(atan(a14))+degrees(atan(a24)))

horx=p1[0]p2[0]
hory=p1[1]p2[1]
verx=p3[0]p4[0]
very=p3[1]p4[1]

for assembly
height=(r3yr4y)cellsy+r4y
width=r2xcellsx
crushheight=height0.8 making it a variable
velocity=crushheight/steptime

breaking after n amounts of repetitive failures
if emptypass200: break

angle sartlari, svf sarti, noktalar unique
if degrees(atan(a13))80 or degrees(atan(a23))80:
    emptypass=+1
    continue

if degrees(atan(a14))10 or degrees(atan(a24))10:
    emptypass=+1
    continue

if not degrees(atan(a13))=degrees(atan(a14))+10:
    emptypass=+1
    continue

```
if not degrees(atan(a23))=degrees(atan(a24))+10:
    emptypass=+1
    continue

if svf0.8:
    emptypass=+1
    continue

if (r4x,r4y) in storage:
    emptypass=+1
    continue

emptypass=0 unique parameters are found, zeroing it
count+=1
storage.append((r4x,r4y))

make model
mdb.Model(modelType=STANDARDEXPLICIT, name=modelname)

UNIT CELL
mdb.models[modelname].ConstrainedSketch(name=profile, sheetSize=200.0)
mdb.models[modelname].sketches[profile].Spot(point=p1)
mdb.models[modelname].sketches[profile].Spot(point=p2)
mdb.models[modelname].sketches[profile].Spot(point=p3)
mdb.models[modelname].sketches[profile].Spot(point=p4)
mdb.models[modelname].sketches[profile].Line(point1=p1, point2=p3)
mdb.models[modelname].sketches[profile].Line(point1=p3, point2=p2)
mdb.models[modelname].sketches[profile].Line(point1=p2, point2=p4)
mdb.models[modelname].sketches[profile].Line(point1=p4, point2=p1)
```

```
"""
THE MAIN SCRIPT FOR MACHINE LEARNING STUFF, including:
    Three models training  NN, RF, XGB
    Hyperparameter optimization
    Scaling by standardization
    Outlier detection and removal by IQR method
    AND GENETIC ALGORITHM OPTIMIZATION
"""

def plotlearningcurve(model, modelname, cols, X, y):
     sns.settheme(style="ticks", fontscale=2.4)

    fig, axes = plt.subplots(nrows=5, ncols=10, figsize=(15, 20))
    plt.subplotsadjust(left=0.12, bottom=0.08, right=0.95,
                               top=0.92, wspace=0.4, hspace=0.3)

    handles = []
    labels = []

    for i, (ax, col) in enumerate(zip(axes.flat, cols)):
        trainsizes, trainscores, testscores = learningcurve("check out git")
        linetrain, = ax.plot(trainsizes, np.mean("check out git")
        linetest, = ax.plot(trainsizes, np.mean("check out git")

        colname=col.split("(")[0]

        Collect handles and labels for the legend
        if i == 0:
            handles.extend([linetrain, linetest])
            labels.extend(["Training Score", "Test Score"])
```

```
        Set yticks for all axes
        ax.setyticks([0.8, 0.88, 0.95, 1.02])

        Add "R2" only to the leftmost subplots
        if i  2 == 0:
            ax.setylabel(f"R2",labelpad=10)
        else:
            ax.setylabel(f" ")

        if i == 6 or i == 7:  indices for the 4th and 8th subplots
            ax.setxlabel(f"Number of samples", labelpad=10)

        Only set xticks for the bottom row
        if i  len(cols)  2:
            ax.setxticklabels([])

        Only set yticks for the leftmost column
        if i  2 != 0:
            ax.setyticklabels([])

        ax.text(1.06, 0.5, colname, va=center, ha=center,
                rotation=90,transform=ax.transAxes)

Create a single legend for all subplots
fig.legend(handles[:2], labels[:2], loc=upper center,
                bboxtoanchor=(0.5, 0.99), ncol=2,
                shadow=True, fancybox=True)

plt.savefig("learningcurve.png")
plt.show()

return None
```

```
def removeoutliers(df,columns):
    """
        columns  list
        df  data frame
        takes columns, loop them through remove outliers based on
        upper and lower quartile values, a proven method
    """

    for col in columns:
        Q1=np.percentile(df[col],25,method=midpoint)
        Q3=np.percentile(df[col],75,method=midpoint)
        IQR=Q3Q1
        upper=Q3+1.5IQR
        lower=Q11.5IQR
        df=df[(df[col]=lower)  (df[col]=upper)]
    return df

def normalize(dfin):
    normalizer=MinMaxScaler()
    for col in dfin.columns:
        try: dfin[col]=normalizer.fittransform(dfin[[col]])
        except: pass
    return dfin

def standardize(dfin):
    standizer=StandardScaler()
    for col in dfin.columns:
        try: dfin[col]=standizer.fittransform(dfin[[col]])
        except: pass
    return dfin
```

```
def xgbhpopt(model,Xvalid,yvalid):
    kf = KFold(nsplits=5, shuffle=True)
    parameters =
            nestimators: [100,200,400,1000],
            maxdepth: [4,8,16, None],
            minchildweight:[1,2,4,8],
            learningrate: [1e4, 1e3, 1e2,0.1], so called eta value


    xgbgrid = GridSearchCV(model,
                           parameters,
                           cv = 10, KFold number
                           njobs = 1, processors
                           verbose=False)

    xgbgrid.fit(Xvalid,yvalid)
    print(xgbgrid.bestparams)
    return xgbgrid.bestestimator,xgbgrid.bestparams

def runmodel(model,modelname,output,Xtrain,ytrain,Xtest,ytest):
    model.fit(Xtrain,ytrain)
    ypred=model.predict(Xtest)

    r2=r2score(ytest, ypred)
    mape=meanabsolutepercentageerror(ytest, ypred)
    rmse=meansquarederror(ytest, ypred)

    ss[output]=(r2,mape,rmse)
    plot=True
```

```
if plot:
    plt.figure(figsize=(16,12),layout="constrained")

    scatter
    if modelname=="MLP": color="crimson"
    if modelname=="XGB": color="steelblue"
    if modelname=="RF": color="green"
    else: color="orange"

    plt.scatter(ytest,ypred,s=400,color=color,edgecolors="black",linewidths=1)
    plt.savefig(f"modelnameoutput[0:3].png")

    "and it goes on"""
```