Available online at www.sciencedirect.com

Manufacturing Letters

Manufacturing Letters 00 (2023) 000–000

51st SME North American Manufacturing Research Conference (NAMRC 51, 2023)

Grid Search Hyperparameter Tuning in Additive Manufacturing Processes

Michael Ogunsanya[a], Joan Isichei[a], Salil Desai[a,*]

[a]*Center of Excellence in Product Design and Advanced Manufacturing, North Carolina A&T State University, 1601 E Market Street, Greensboro, NC 27401, USA*

* Corresponding author. Tel.: +1-336-285-3725; fax: +1-336-334-7729. E-mail address: sdesai@ncat.edu

Abstract

In Machine learning (ML) and deep learning (DL), hyperparameter tuning is the process of selecting the combination of optimal hyperparameters that give the best performance. Thus, the behavior of some machine learning (ML) and deep learning (DL) algorithms largely depend on their hyperparameters. While there has been a rapid growth in the application of machine learning (ML) and deep learning (DL) algorithms to Additive manufacturing (AM) techniques, little to no attention has been paid to carefully selecting and optimizing the hyperparameters of these algorithms in order to investigate their influence and achieve the best possible model performance. In this work, we demonstrate the effect of a grid search hyperparameter tuning technique on a Multilayer perceptron (MLP) model using datasets obtained from a Fused Filament Fabrication (FFF) AM process. The FFF dataset was extracted from the MakerBot MethodX 3D printer using internet of things (IoT) sensors. Three (3) hyperparameters were considered – the number of neurons in the hidden layer, learning rate, and the number of epochs. In addition, two different train-to-test ratios were considered to investigate their effects on the AM process data. The dataset consisted of five (5) dominant input parameters which include layer thickness, build orientation, extrusion temperature, building temperature, and print speed and three (3) output parameters: dimension accuracy, porosity, and tensile strength. RMSE, and the computational time, CT, were both selected as the hyperparameter performance metrics. The experimental results reveal the optimal configuration of hyperparameters that contributed to the best performance of the MLP model.

Keywords: Grid Search, Hyperparameter Tuning, Additive Manufacturing, Multilayer Perceptron, Machine Learning.

1. Introduction

Additive manufacturing (AM), which is also known as three-dimensional (3D) printing, is a technique rapidly gaining ground in numerous fields including engineering, healthcare, manufacturing, aerospace, and medicine [1], [2]. The global market for AM grew by 7.5% to $12.8 billion in 2020 in spite of the current global Covid-19 pandemic [3]. As its name implies, AM consists of adding materials layer by layer to build a part or product. AM facilitates the production of components, parts, or products with complex geometries with the aid of computer aided design (CAD). Such parts can be manufactured using various materials including polymers [4], biomaterials [5] and metals [6]. In recent times, textiles [7], ceramics [8], biomaterials [9], glass [10], and batteries [11], [12] have been successfully printed using various AM techniques [13]. In additive manufacturing, the relationships between part design, manufacturing parameters, dimensional accuracy, part quality and reliability are not fully understood. However, part quality and reliability are extremely crucial to sustaining the structural integrity required for AM parts. Defects in AM parts could diminish the public's trust in the technology. Subsequently, it is necessary to detect any defects that may occur in the AM process which may cause a discrepancy between the target designs and printed parts. However, identifying the imperfections within complex

printed features accurately and efficiently is a difficult and challenging task. Machine and Deep learning models/algorithms offer a distinctive and intriguing approach to addressing this challenge. Recent advances have witnessed the application of ML/DL to AM techniques such as fused filament fabrication (FFF) [14], [15]. FFF is a prominent AM technique that constructs parts by extruding a semi-molten metal or polymer filament through a heated nozzle in a specified pattern onto a build bed [16]. FFF is the primary AM technique under focus in this paper.

1.1. Machine and Deep learning Techniques

ML/DL models rely on training and test datasets to deduce knowledge and vital relationships between various AM features or attributes and make predictions based on the acquired knowledge. ML/DL algorithms are also extremely beneficial because they can be used to determine optimal AM processing parameters, thus, making them useful for applications such as real time in-situ AM defect detection [17], [18]. Based on the foregoing, it is imminent that ML is a vital aspect of Industry 4.0 [19]–[21]. In this paper, the multilayer perceptron (MLP) ML was considered for our analysis. This is because MLP can deal with highly complex systems and has seen wide adoption in multiple input multiple output (MIMO) systems [22]. An MLP network comprises an input layer, single or multiple hidden layers, and an output layer. The proficiency of a neural network is contingent on appropriately choosing the number of hidden layers, the number of neurons in each hidden layer, activation functions at each layer, and optimizers. The MLP algorithm works on the feed-forward back propagation approach.

While there has been a rapid growth in the application of machine learning (ML) and deep learning (DL) algorithms to generic application. However, in the context of Additive manufacturing (AM), little to no attention has been paid to carefully selecting and optimizing the hyperparameters of these algorithms to investigate their influence and achieve the best possible model performance. Based on an exhaustive literature search, our findings indicate the dearth of hyperparameter tuning of machine learning algorithms to improve the performance of predictive additive manufacturing models. Thus, the current research of hyperparameter tuning in additive manufacturing addresses shortcoming mentioned above.

1.2. Hyperparameter Tuning/Optimization

In machine learning, two main parameters are considered: the model parameters and the hyperparameters [23]. Model parameters are the parameters that are internal, configurable, and can be estimated based on the given data set. These parameters are learned and estimated after the training phase of machine or deep learning models. For example, the weights and biases in deep learning models are often initialized to zeros and the task in the training phase of the deep learning model is to optimize both the weights and biases to give the least loss in a regression problem or the highest accuracy in a classification process without underfitting or overfitting. In summary, model parameters are obtained after training the model. On the other hand, hyperparameters are external parameters of the model. They are parameters that are required to be set before training the chosen machine or deep learning model. They are predetermined before the training commences and used to control the learning process. If the hyperparameters are carefully chosen, then the training phase can guarantee better learning. Thus, leading to better performance of the chosen machine learning algorithm.

Grid search hyperparameter tuning algorithm was used in this work because of the following advantages: 1) Exhaustive: Grid search considers every feasible combination of hyperparameters, and this will always guarantee an optimal solution. 2) Simple and straightforward: Grid search employs a simple and straightforward method that can be easily executed.

The grid search approach is an exploratory algorithm that evaluates hyperparameter performance at all possible settings thus, is an exhaustive search approach. It is an independent search algorithm which entails testing every unique combination of hyperparameters in the search space to determine the combination that yields the best performance. On the contrary, the Bayesian optimization is an informed search approach, which augments the learning behavior from previous iterations. In addition, more time is required to determine the next hyperparameters to evaluate based on the results of the previous iterations. At the expense of minimizing the number of trials, Bayesian optimization requires more time for each iteration. Thus, to ensure an exhaustive yet timely solution a grid search approach was implemented in this research.

One of the main contributions of this work is to demonstrate how hyperparameter tuning can be used to explore the hyperparameter configuration/search space. Furthermore, how these steps when incorporated effectively into any ML/DL algorithms would help to obtain an optimal ML/DL model. This would help researchers and practitioners to always consider hyperparameter optimization as one of the core steps for fully exploiting the potential of their chosen ML/DL model in order to unravel the complexity and nonlinearity in their data.

The remainder of this work consists of the system overview in Section 2, the chosen ML/DL learning model and the grid search hyperparameter tuning technique were fully explained in Section 3, Section 4 shows the results and discussion on the findings, Section 5 concludes the work and further directions are provided as well.

2. System Overview

For this work, additive manufacturing process data were obtained for the most widely used fused filament fabrication (FFF) which is classified as a material extrusion AM process by the American Society for Testing and Materials (ASTM) International D638 for tensile stress testing.

2.1. Printer descriptions

The Makerbot MethodX 3D printer shown in Fig.1a and Fig.1b was used to print the tensile strength specimen as given by ASTM International D638 for tensile strength testing [24] (Fig. 2a). The MethodX was used because it has a closed chamber for controlling the build temperature. Fig. 2b shows a sample printed at a given set of input parameters.

Fig. 1a. The MakerBot MethodX mounted with IOT sensors for capturing fused filament fabrication data

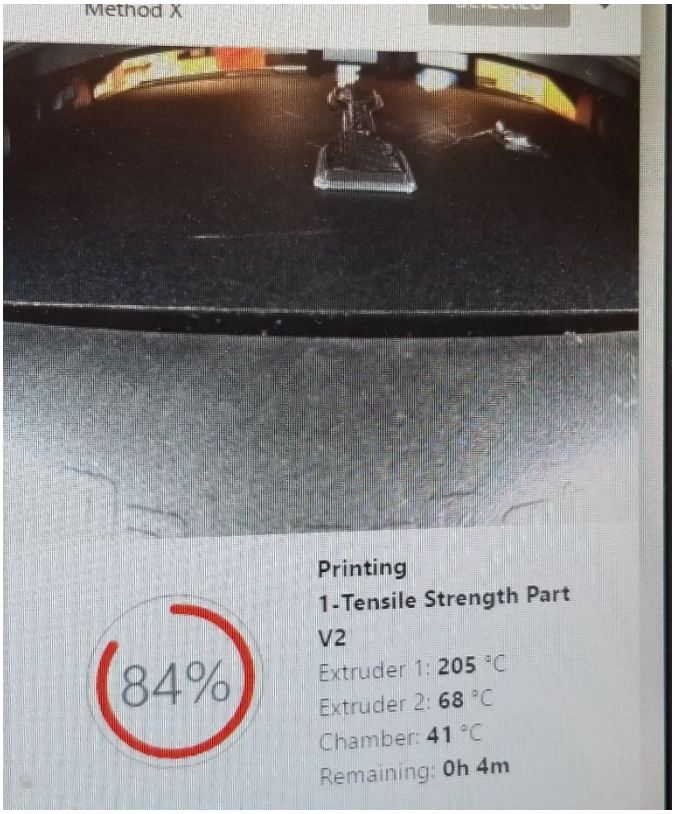

Fig. 1b. The MakerBot MethodX camera feed showing printing in action.

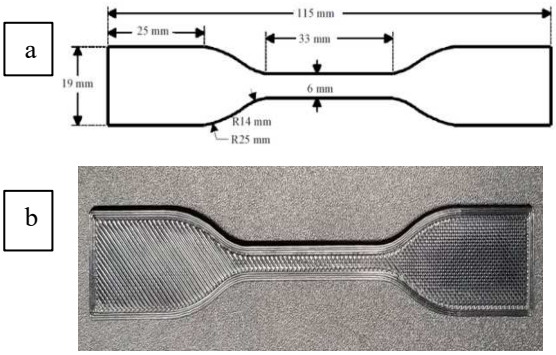

Fig. 2. a. ASTM specimen for AM tensile strength b. A printed sample of the ASTM standard at candidate input parameters

3. Methodology

This section introduces the selected deep learning model's description, the model's architecture, types of activation functions, and the chosen rectified linear unit. Second, the hyperparameter tuning problem which was the major contribution in this paper was defined mathematically. Third, the grid search technique, which was the chosen hyperparameter tuning technique for this work was introduced and the steps used were itemized. Fourth, the fused deposition modeling additive manufacturing process dataset and the evaluation metric for the model at a given hyperparameter vector, λ, were explained. Last, the Python scripting detail was fully explained.

3.1. The multilayer perceptron deep learning model

Multilayer perceptron (MLP) was used in this work to study the interplay between the considered input and output parameters. Multilayer perceptron is a type of feedforward artificial neural network that is fully connected [25]. For an MLP, there must be at least three layers – an input layer, a hidden layer, and an output layer. This can also be referred to as a "vanilla" neural network. MLP is a type of supervised learning, and this learning is done via the backpropagation during the training phase [26], [27]. MLP perceptron was chosen over other ML/DL algorithms for this work because of its appropriateness for the FFF dataset. Deep learning (DL) models such as Recurrent neural networks (RNN) and long short-term memory networks (LSTM) are best suited for sequential and time-series data [28]. Another widely used DL algorithm is convolutional neural networks (CNN), however it is best used for image classification [29]. Other machine learning (ML) algorithms such as support vector machines (SVM), Random Forest, K-Nearest Neighbor (KNN), and Stochastic Gradient Boosting (SGB) can be used for the FFF hyperparameter study. However, MLP has a higher ability of unraveling complex non-linearity that exist in systems such as additive manufacturing process [22], [30], [31]. Also, it works well with smaller data set as it does with large input data. Also, it has multiple hyperparameters, for example, number of hidden layers, number of neurons in each hidden layers, activation function, learning rate, number of epochs, etc. that can be tuned to obtain a model with optimal hyperparameter vectors for the given dataset. These hyperparameters help to explicitly demonstrate our work for an FFF additive manufacturing processes.

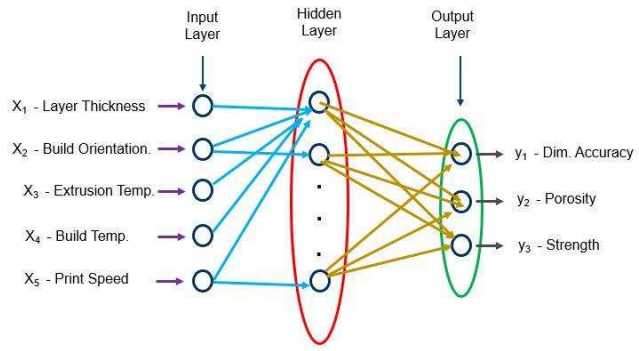

Fig. 3. A multilayer perceptron (MLP) architecture with one input layer, one hidden layer, and output layer

Fig. 3 shows the MLP model architecture – the input layer, hidden layers, and output layer used in this work as similar to [27] and definition on each layer is given by [32]. The input layer serves as the layer that contains the neurons; each neuron represents an input parameter. The output is the layer containing the node that measures the output parameter. In this work, the three outputs were considered at a time. Lastly, the hidden layer, which is the layer between the input and output layers. The hidden layer can be as many as one or more depending on how dense the MLP would be. It helps the model to learn some complexity in the given data set. For an illustrative case, one dense layer was considered in this work.

3.2. Activation function

It is a type of function used in deep learning to unravel the nonlinearity or complexity in any given data. It goes further than that by removing linearity from the neural network. There are various types of activation functions as explicitly defined in the work of [33] such as sigmoid function, tanh function, rectified linear unit (ReLU), exponential linear unit (ELU), exponential function, scaled exponential linear unit (SELU), etc. Rectified linear unit (ReLU) function was used as the activation function in this work.

3.2.1. Rectified linear unit (ReLU)

A rectified linear unit, ReLU, is an activation function, f(x) that returns the value of the independent variable if positive, and zero, otherwise. Mathematically, a ReLU function is given in Equation 1:

$$f(x) = max\ (0, x) \qquad (1)$$

3.3. Mathematical definition of hyperparameter tuning

The formal mathematical hyperparameter tuning definition given by [34] was adapted to this work. Given a machine learning algorithm, $\mathcal{A}$, with N number of hyperparameters, d_n is the domain of the n -th hyperparameter, such that the hyperparameter configuration space, $d = d_1 \times d_2 \times ... \times d_N$, $\lambda \in d$ is a vector of hyperparameters, $\mathcal{A}_\lambda$ denoted a machine learning $\mathcal{A}$ having its hyperparameters instantiated to λ. The problem at hand is to find the optimal vector of hyperparameters, λ^*, that has the least loss, L, of a model generated by the machine learning algorithm, A, instantiated at a vector of hyperparameters, λ, on training data, D_{train}, and evaluated on test data, D_{test} as given in Equation 2 :

$$\lambda^* = \underset{\lambda \in d}{argmin}\, f(L, \mathcal{A}_\lambda, D_{train}, D_{test}) \qquad (2)$$

3.4. Hyperparameter tuning techniques

Many works exist in hyperparameter tuning or optimization. Hyperparameter tuning is often referred to as a black box problem [35], and many techniques are found in the literature. Some of these techniques are manual tuning [36], [37], grid search [36], [38], [39], random search [36], [40], Bayesian optimization [35], [41], genetic algorithm [33], particle swarm optimizations [42], etc. Some of the techniques have variants as explored by different researchers in the field, especially for Bayesian optimization.

The grid search is considered in this work for exploration purposes and for a deep illustrative case of the additive manufacturing process. As seen in many recent times, much attention has been on developing new machine learning models, whereas there is a need to explore the effects of hyperparameter tuning of the chosen model on the data.

3.5. Grid search algorithm

Grid search hyperparameter tuning is a systematic way of creating a grid from the considered hyperparameters in which each possible combination is used to tune the chosen model [15]. It is an exhaustive search method, as each combination is observed one at a time [20]. In addition, the search range for each hyperparameter is many folds. Also, in grid search, all hyperparameters are assumed to have equal weights irrespective of their effect on the machine learning training phase. Grid search was used because it is an exhaustive method as it considers all feasible hyperparameter combinations which supports the objective of this work. Also, it is simple and straightforward as it can be applied without much mathematical expertise.

3.5.1. Grid search hyperparameter tuning procedure

To fully see the workings of hyperparameter tuning, the grid search technique is chosen for full hyperparameter exploration, and the following steps were carried out:

1. Select the hyperparameters to be tuned.
2. For each hyperparameter, determine the search range.
3. Systematically, obtain all possible combinations.
4. In turn, each combination during the training of the model obtains the RMSE values and computational time based on the stopping condition. In this case, the number of epochs.
5. Rank your hyperparameter combination from the least RMSE values, then the computational time. Break tie arbitrarily.

For the chosen deep learning model, that is, the multilayer perceptron, three hyperparameters were considered – the number of neurons in the hidden layer, learning rate, and the number of epochs. The hyperparameter list and search ranges for the considered hyperparameters are given as:

1. Number of neurons in each hidden layer = [3, 6, 9]
2. Learning rate = [0.001, 0.0001, 0.00001]
3. Epoch = [5000, 10000, 20000]

From the hyperparameter sets above, there are $3 \times 3 \times 3 = 27$ different hyperparameter combinations. In addition to these, two different train-to-test ratios were considered to investigate their effects on the AM process data. The considered train-to-test ratios in this work are 70/30 and 80/20.

The framework in this section is extensible to other chosen machine learning models. The hyperparameters and hyperparameter ranges can be chosen accordingly for the considered model to give the hyperparameter combinations/vectors.

3.6. Dataset and evaluation metrics

3.6.1. Dataset

From the hyperparameter tuning framework in Section 3.5.1, a combination of AM input and output data can be used. The AM input and output parameters of interest need to be measured, and the appropriate design of experiment can be performed. The obtained data serves as the dataset.

For this work, the AM dataset was obtained using the MakerBot MethodX 3D printer using internet of things (IoT) sensors. Five of the dominant input parameters were considered [43] and three print outputs [44] were measured. The input parameters include layer thickness, build orientation, extrusion temperature, building temperature, and print speed while the measured outputs were dimension accuracy, porosity, and tensile strength. The feed rate of the filament was held constant as is typically done for FFF systems. For each input parameter, three different levels were considered, which led to a full factorial experimental design There were a total of $3^5 = 243$ data points which were split based on the two train-to-test split ratios. See Table 1, for a snapshot of the 243 data points.

3.6.1.1. Output parameters

Dimension accuracy was measured in this work by measuring how closely the 3D printed structure match the tensile strength specimen as given by ASTM International D638 computer-aided design (CAD). The deviation was calculated following the guide in [43] as given below. Nine different measurements were measured for individual printed specimen. The overall length (OL) of the 3D printed part was measured with a vernier caliper. Other measurements such as the total width (OW), the thickness and the inner width were measured at three different points which resulted into three measurements. The three measurements for the thickness and width were averaged to a single value thickness (T) and width (W). Individual dimensional error was computed using equation 3 while equation 4 was used to compute the overall dimensional accuracy.

$$Error,\ \varepsilon = Measured\ value - CAD\ value \quad (3)$$

$$Dimensional\ Accuracy\ (\%) = \frac{\varepsilon_L + \varepsilon_{OW} + \varepsilon_W + \varepsilon_T}{4} \times 100 \quad (4)$$

$$\varepsilon_{\mathrm{L}} = error\ in\ overall\ length,$$
$$\varepsilon_{OW} = error\ in\ overall\ width$$
$$\varepsilon_W = error\ in\ inner\ width$$
$$\varepsilon_T = error\ in\ overall\ thickness$$

Table 1: Snapshot of FFF Dataset

Color key: good level – green, medium level – yellow, and low level - red

	Input/process parameters					Mean Output		
	Layer thickness (mm)	Build orientation(degree)	Build temperature (°C)	Extrusion temperature (°C)	Print speed (mm/s)	Dimensional Accuracy (%)	Porosity (%)	Tensile Strength (MPa)
1	0.2	0	30	160	25	7	13	29
2	0.2	0	30	180	50	4.5	9.5	33.5
3	0.2	0	30	200	75	6.5	11	30.5
4	0.2	0	60	160	50	7.5	13	29.5
5	0.2	0	60	180	75	4	8.5	32.5

Although, porosity of FFF AM printed parts can be measured using different methods, such as, microscopy of a polished cross-section, Archimedes' principle, X-Ray computed tomography [45], and simple ratio of mass to volume approach [46]. This work used the latter method to measure the FFF AM printed parts as explicitly explained in the works of [47] Equation 3 shows the formula for computing the porosity.

$$Porosity\ \% = \frac{V_b - V_s}{V_b} * 100 \qquad (5)$$

$$where\ V_b = the\ volume\ of\ bulk\ specimen,$$

$$V_s = the\ volume\ of\ the\ specimen$$

Each 3D printed was tested for tensile strength for a given set of input parameters. Tensile strength testing was performed using the universal testing machine - Instron 5542 (Canton, MA, USA) with a 500N load cell and a displacement rate as detailed for the ASTM D628. We adopted the specific details on tensile strength testing of FFF-printed parts using the work of [48].

3.6.2. Evaluation metrics

For each vector of hyperparameters or hyperparameter combination, λ, performed by the grid search technique, both the root-mean-squared error, RMSE, and the computational time, CT, were computed and they both served as the hyperparameter performance metrics. RMSE serves the first performance metric in this work, then the computational time in this work. Computational time would serve as a tiebreaker when two or more hyperparameter combinations have the same or close RMSE values. In some other applications, especially where time cannot be compromise, computational time might take the lead.

3.6.2.1. Root-mean squared error (RMSE)

The root-mean-squared error is the square root of the mean of the squared prediction errors across all the output test data sets as given in Equation 6. The root-mean-squared error had the same unit as the measured output parameter.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n_{test}} (y_{i,test} - \hat{y}_{i,\ pred})^2}{n_{test}}} \qquad (6)$$

where,
$y_{i,test}$ is the ith actual output test data
$\hat{y}_{i,\ pred}$ is the ith predicted output test data
n_{test} is the total number of the output test data

3.6.2.2. Computational time (CT)

In this work, computation time was measured as the time taken from the training of the MLP model at the given hyperparameter combination to the plotting of the visuals and computing the errors.

3.6.3. Model training status

After obtaining both the RMSE values and computational time, before the results can be used, the status of the MLP model during the training phase must be checked. There are two possible scenarios – "Learning" or "No-Learning".

3.6.4. Learning status

Learning occurs when the MLP model can be trained using the backpropagation technique. The ability of the MLP to be trained at the given hyperparameter combination is observed from the MLP prediction on the input test data set. It is easily noticeable from the plotted visuals as illustrated in Fig. 4.

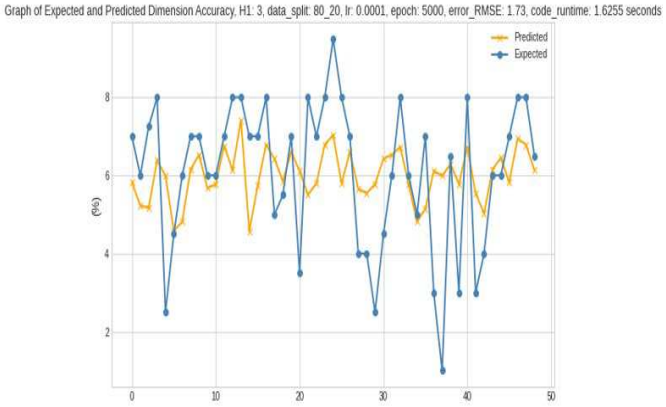

Fig. 4. A plot of actual/expected vs predicted dimension accuracy from the test data set showing learning status

3.6.5. No-learning status

No learning occurs if the prediction from the MLP model on the input test data set is either all zeros or it has the same values all through the prediction as shown in Fig. 5. This could be a result of being trapped in a local optimal during the training phase using the backpropagation technique. It further means that at the considered hyperparameter combination, the complexity and nonlinearity of the data set cannot be learned. Interestingly, this is one of the benefits of hyperparameter tuning in machine learning and deep learning domains.

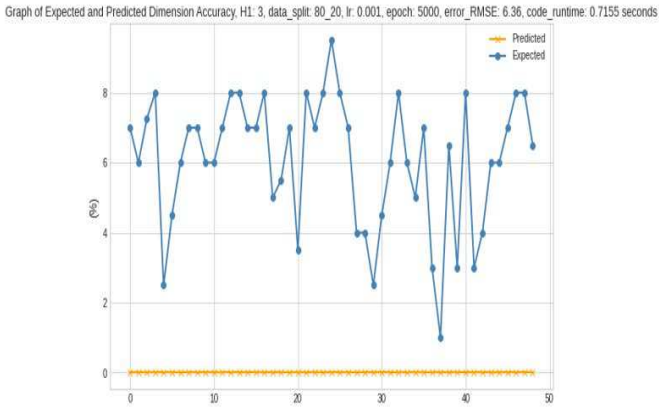

Fig. 5. A plot of actual/expected vs predicted dimension accuracy from the test data set showing no-learning status

3.7. Python scripting procedure

The pseudocode for the implementation of this work is given as:

START PROGRAM

IMPORT necessary libraries or modules

LOAD dataset

BUILD the multi-layer perceptron deep learning model

LIST SET of MLP hyperparameters with their range

LIST dataset test size

FOR each test size

Split dataset into train and test datasets

Split train dataset into input and outputs

Split test dataset into input and outputs

INITIALIZE random weights and biases

FOR each output parameter

FOR each hyperparameter vector

OPEN a file to save desired results

SET test size iteration COUNTER, n = 0

WHILE n < 2

START timer == initial time

FOR the range of the epoch

DO forward propagation

DO backward propagation

UPDATE both weights and biases

GET weights and biases

PREDICT on the trained model with test data

COMPUTE root-mean-squared value

STOP timer == final time

COMPUTE CT = final time - initial time

SAVE all results

n = n + 1

COMPUTE means of RMSE and CT

PLOT all needed visuals

END PROGRAM

4. Results and Discussions

Table 2: Mean RMSE values and computational times for dimension accuracy, porosity, and strength for the 18 hyperparameter vectors with "Learning" status

					Dimension Accuracy		Porosity		Strength		
	H1	LR	Epoch	Hyperparameter Vector	Mean RMSE	Mean CT	Mean RMSE	Mean CT	Mean RMSE	Mean CT	Status
1	3	0.0001	5000	3,0.0001,5000	1.74	1.91	3.60	1.98	6.03	2.01	Learning
2	3	1.00E-05	5000	3,0.00001,5000	1.7	2.92	3.51	3.09	6.07	3.09	Learning
3	3	0.0001	10000	3,0.0001,10000	1.75	6.31	3.60	6.49	6.03	6.53	Learning
4	3	1.00E-05	10000	3,0.00001,10000	1.7	8.00	3.51	8.23	6.04	8.30	Learning
5	3	0.0001	20000	3,0.0001,20000	1.75	13.92	3.60	14.20	6.03	14.44	Learning
6	3	1.00E-05	20000	3,0.00001,20000	1.7	16.87	3.56	17.21	6.03	18.54	Learning
7	6	0.0001	5000	6,0.0001,5000	1.73	1.98	3.60	1.98	6.03	2.37	Learning
8	6	1.00E-05	5000	6,0.00001,5000	1.715	3.07	3.51	3.13	6.08	3.58	Learning
9	6	0.0001	10000	6,0.0001,10000	1.75	6.66	3.60	6.81	6.03	7.30	Learning
10	6	1.00E-05	10000	6,0.00001,10000	1.705	8.47	3.51	8.66	6.05	9.20	Learning
11	6	0.0001	20000	6,0.0001,20000	1.75	14.96	3.60	15.51	6.03	15.74	Learning
12	6	1.00E-05	20000	6,0.00001,20000	1.7	18.16	3.54	19.52	6.03	19.01	Learning
13	9	0.0001	5000	9,0.0001,5000	1.74	2.14	3.60	2.17	6.03	2.21	Learning
14	9	1.00E-05	5000	9,0.00001,5000	1.7	3.34	3.51	3.39	6.08	3.43	Learning
15	9	0.0001	10000	9,0.0001,10000	1.75	7.82	3.60	7.40	6.03	7.44	Learning
16	9	1.00E-05	10000	9,0.00001,10000	1.7	10.27	3.51	9.40	6.05	9.48	Learning
17	9	0.0001	20000	9,0.0001,20000	1.75	17.28	3.60	16.49	6.03	17.73	Learning
18	9	1.00E-05	20000	9,0.00001,20000	1.7	20.78	3.55	20.09	6.03	21.35	Learning

As stated in Section 3.6.3, the training status of the MLP model must first be ascertained before the model's RMSE values and computational time are considered.

Table 2 shows the mean RMSE values and mean computational times (CTs), at 70:30 and 80:20 train-to-test ratios each ran for two iterations, where training status was "Learning". Learning occurred in 18 hyperparameter vectors out of the possible 27.

For further analysis, all hyperparameter vectors, λ, that have "No-Learning" status are not considered. Only 18 out of the 27 hyperparameter vectors have "Learning" as their training status for dimension accuracy, porosity, and strength. Therefore, the remaining 9 hyperparameter vectors were not further considered.

4.1. Dimension Accuracy

For the MLP, predictions were made at 18 different hyperparameter vectors. The values of the mean RMSE values for the two different train-to-test ratios of 70:30 and 80:20 for two iterations each range from 1.70 to 1.75 and at various computational times. Fig. 6 illustrates sorted hyperparameter vectors first on the least RMSE value then on the computational time. The optimal hyperparameter vector rate was 0.00001. No learning occurs at a learning rate of 0.001 which implies, learning rate is key to learning of the complexity and nonlinearity of the dimensional accuracy data. Some of the highest computational times are observed when the hidden layer has 9 neurons which translates to the need for more computations as the neurons increase. Thus, computational time increased with an increase in the number of epochs and also increased as the learning rate was reduced. Based on the evaluation metric selection given in Section 3.6.2, the optimal hyperparameter vector for the given MLP is 3 neurons in the hidden layer, learning rate of 0.00001, and epoch of 5000 for the given dataset. On the other hand, if CT is given more weight, then an optimal hyperparameter vector is 3 neurons in the hidden layer, learning rate of 0.0001 and epoch of 5000 for the MLP which is applicable in a process where computational time cannot be compromised. Interestingly, a compromise on the RMSE value by 0.03 gives about 35% reduction in CT. This is achieved by using a learning rate of 0.0001 in place of 0.00001. Thus, an application determines if this compromise can be made.

Fig. 7 shows the comparison of the RMSE values for the dimensional accuracy at train-to-test of 70:30 and 80:20. The curves shows that increasing the training data set from 70% to 80% of the data set improved RMSE values except when the hyperparameter vectors has its learning rate to be 0.00001. So, all improved RMSE values occurred at learning rate of 0.0001 irrespective of other hyperparameters.

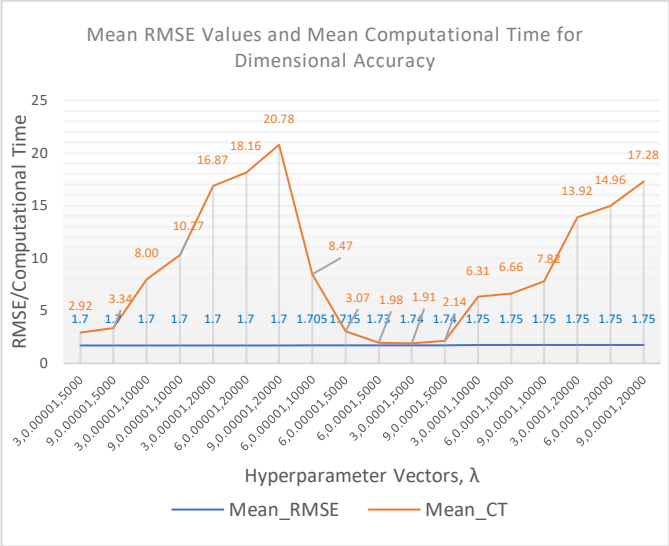

Fig. 6. Mean values for the RMSE values and the computational time for dimensional accuracy.

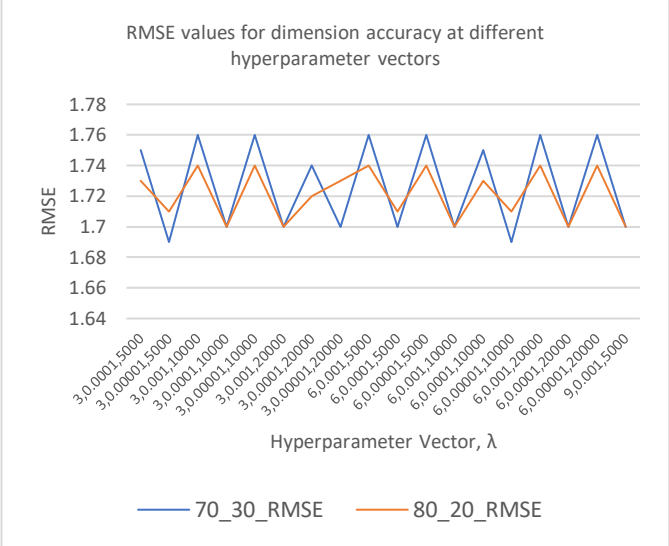

Fig. 7. Comparison of the RMSE values for dimension accuracy on the test data set at different hyperparameter vectors for 70:30 and 80:20 train-test split

4.2. Porosity

Porosity has its optimal hyperparameter vector when the hidden layer has 3 neurons, learning rate of 0.00001, and 5000 epochs. This corresponds to a mean RMSE value of 3.51 and occurred at computational time of 3.09s. For the porosity as shown in Fig.8, the least RMSE value occurred at 3.51 and the worst at 3.60. Most of the highest computational time occurred when the learning rate was 0.00001. In this case, the optimal hyperparameter vector occurs with the hidden layer having 3 neurons, 0.00001 learning rate, and 5000 epochs. On the other hand, if CT has a higher weight compared to RMSE, then an optimal hyperparameter vector occurs when the number of neurons in the hidden layer is 3, learning rate is 0.00001, and epochs of 5000. Similarly, to Section 4.1, if RMSE can be compromised a bit, from 3.51 to 3.60, the computational time can be dropped by about 37% by using a learning rate of 0.0001 instead of 0.00001 From Fig. 9, increasing the training data set from 70% to 80% of the entire dataset improved the RMSE values at all hyperparameter vectors. This guarantees that having more porosity data would help reduce improve the MLP model.

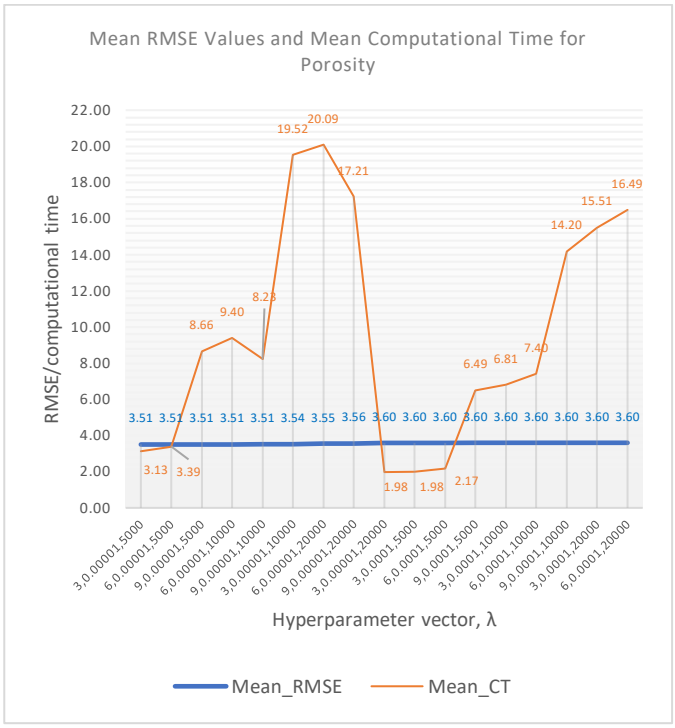

Fig. 8. Mean values for the RMSE values and the computational time for porosity

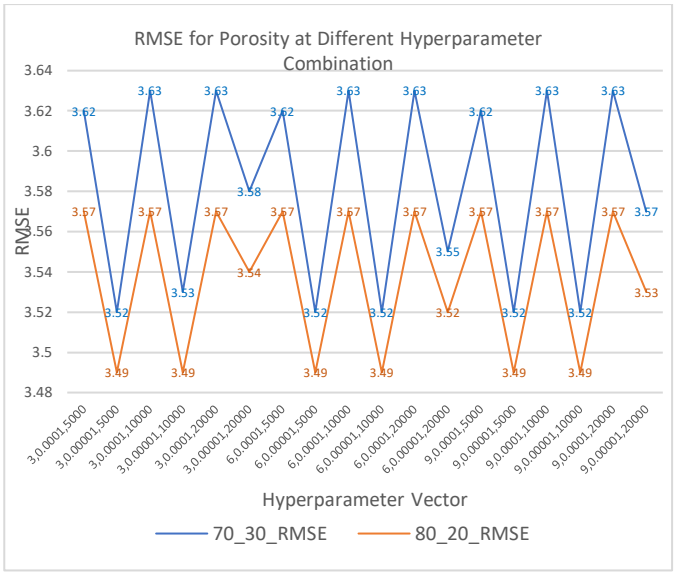

Fig. 9. Comparison of the RMSE values for porosity on the test data set at different hyperparameter vectors for 70:30 and 80:20 train-test split

4.3. Strength

The RMSE values for strength vary from 6.03 to 6.08, and the least and highest computational time are at 2.01 s and

21.35 s respectively as shown in Fig.10. Although, the best optimal hyperparameter vectors occurred when the hidden layer has 3 neurons, learning rate of 0.0001, and 5000 epochs at an RMSE value of 6.03 with a computational time of 2.01, but other hyperparameter vectors within a closed computational time occurred at the same learning rate of 0.0001 and 5000 epochs but at 6 and 9 neurons in the hidden layer. Other lower computation times were observed at epochs of 5000 in which the highest was at 3.58 s. Either a learning rate of 0.00001, 20000 epochs, or both contributed to higher computational time although some of the hyperparameter vectors gave RMSE values of 6.03.

Fig. 11 shows that increasing the training data from 70% to 80% improved the RMSE values by about 8% corresponding to all hyperparameter vectors. This implies that the more data available for training, the RMSE values can be improved for the chosen hyperparameter vector.

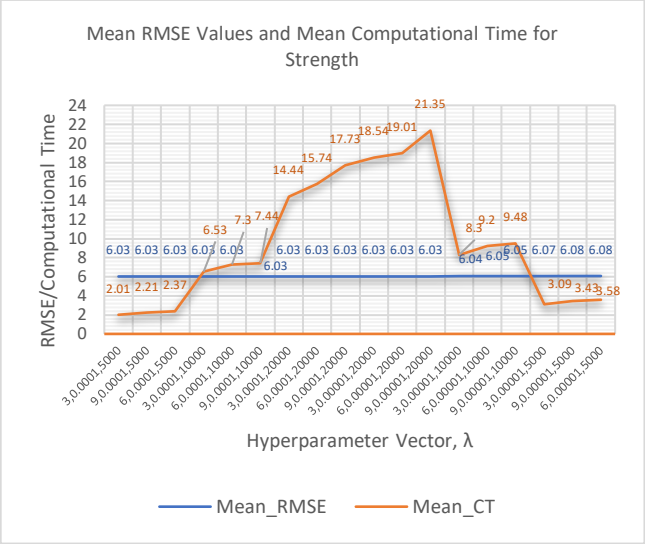

Fig. 10. Mean values for the RMSE values and the computational time for strength

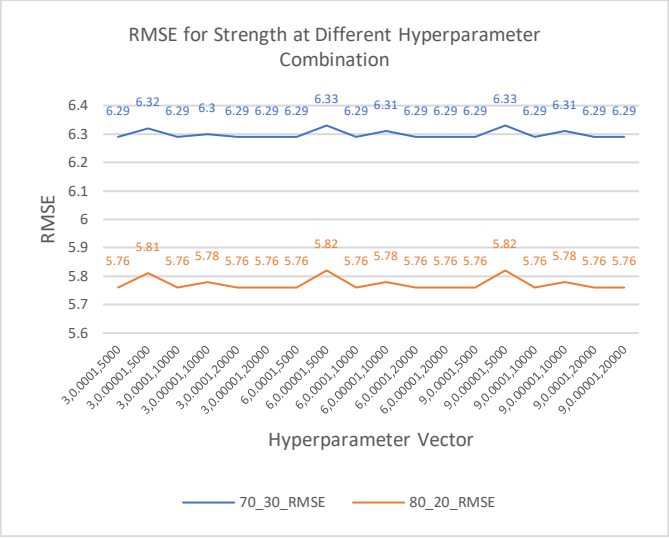

Fig. 11. Comparison of the RMSE values for strength on the test data set at different hyperparameter vectors for 70:30 and 80:20 train-test split

5. Conclusion and Future Works

Hyperparameter tuning is a crucial aspect of controlling the performance of a machine learning model. Improper tuning of an ML/DL model's hyperparameters may lead to suboptimal results and affect the model's loss function. In this paper, we shine the spotlight on the importance and influence of hyperparameters on the performance of machine and deep learning algorithms when applied to an FFF AM process. The grid search technique along with a Multilayer perceptron (MLP) network were used in this work to illustrate the effectiveness of hyperparameter tuning on ML/DL algorithms. Three (3) hyperparameters were considered – the number of neurons in the hidden layer, learning rate, and the number of epochs. In addition, two different train-to-test ratios were considered to investigate their effects on the AM process data. The FFF dataset was extracted from the MakerBot MethodX 3D printer using internet of things (IoT) sensors. The dataset consisted of five (5) dominant input parameters and three (3) outputs parameters. The input parameters include layer thickness, build orientation, extrusion temperature, building temperature, and print speed while the measured outputs were dimension accuracy, porosity, and tensile strength. The total dataset was made up of 243 data points which were split based on the above-mentioned train-to-test split ratios. RMSE, and the computational time, CT, were both selected as the hyperparameter performance metrics. The results show that for dimensional accuracy and porosity, the optimal hyperparameter vector was obtained when the hidden layer has 3 neurons, learning rate of 0.00001, and an epoch of 5000. An RMSE of 1.7 and a computational time of 2.92 along with an RMSE value of 3.51 and computational time of 3.09s were obtained for dimensional accuracy and porosity respectively. Tensile strength has its optimal hyperparameter vector when the hidden layer has 3 neurons, a learning rate of 0.0001, and 5000 epochs, and at an RMSE value of 6.03 with a computational time of 2.01s.

One of the major findings of this work is that if a single hyperparameter vector is to be used with little tradeoffs, then, the hyperparameter vector for the MLP model can be set at 3 neurons in the hidden layer, learning rate of 0.00001, and 5000 epochs. At this hyperparameter vector, although the computational times were not the optimal but not far off. Both dimensional accuracy and porosity gave optimal RMSE values but 1% off the optimal RMSE value for the strength.

Although we only study an MLP model using a grid search hyperparameter tuning technique. Future research can address the use of other tuning approaches such as random search, Bayesian optimization, genetic algorithm, and particle swarm optimizations, alongside alternative DL algorithms such as recurrent neural networks (RNN).

Acknowledgements

The authors would like to express their gratitude for funding support from the National Science Foundation Grant (NSF CMMI Award #1663128, #2100739, #2100850,

#2200538) and the Center of Excellence in Product Design and Advanced Manufacturing at North Carolina A&T State University.

References

[1] S. K. Parupelli and S. Desai, "A Comprehensive Review of Additive Manufacturing (3D Printing): Processes, Applications and Future Potential," *Am. J. Appl. Sci.*, vol. 16, no. 8, pp. 244–272, 2019, doi: 10.3844/ajassp.2019.244.272.

[2] E. Adarkwa, R. Kotoka, and S. Desai, "3D printing of polymeric Coatings on AZ31 Mg alloy Substrate for Corrosion Protection of biomedical implants," *Med. DEVICES SENSORS*, vol. 4, no. 1, Feb. 2021, doi: 10.1002/mds3.10167.

[3] Wohlers Associates, "Wohlers Annual Report: Additive Manufacturing and 3D Printing State of the Industry," 2021.

[4] F. Khaled Aldawood, A. Andar, S. Desai, G. Giammona, and E. Fabiola Craparo, "A Comprehensive Review of Microneedles: Types, Materials, Processes, Characterizations and Applications," *Polym. 2021, Vol. 13, Page 2815*, vol. 13, no. 16, p. 2815, Aug. 2021, doi: 10.3390/POLYM13162815.

[5] E. Adarkwa, A. Roy, J. Ohodnicki, and S. Desai, "3D printing of drug-eluting bioactive multifunctional coatings for orthopedic applications," *Int. J. Bioprinting*, vol. 110, no. 1, Jul. 2023.

[6] S. K. Parupelli and S. Desai, "Hybrid additive manufacturing (3D printing) and characterization of functionally gradient materials via in situ laser curing," *Int. J. Adv. Manuf. Technol.*, vol. 110, no. 1–2, pp. 543–556, Sep. 2020, doi: 10.1007/s00170-020-05884-9.

[7] D. Sun and A. Valtasa, "3D Printing in Modern Fashion Industry," *J. Text. Sci. Fash. Technol.*, Mar. 2019, doi: 10.33552/JTSFT.2019.02.000535.

[8] C. Wang *et al.*, "A general method to synthesize and sinter bulk ceramics in seconds," *Science (80-.).*, vol. 368, no. 6490, pp. 521–526, May 2020, doi: 10.1126/SCIENCE.AAZ7681.

[9] E. Adarkwa, S. Desai, J. M. Ohodnicki, A. Roy, B. Lee, and P. N. Kumta, "Amorphous calcium phosphate blended polymer coatings for biomedical implants," in *IIE Annual Conference and Expo 2014*, 2014, pp. 132–138, Accessed: Oct. 10, 2022. [Online]. Available: https://www.iise.org/uploadedFiles/IIE/Community/Technical_Societies_and_Divisions/Manufacturing_and_Design/FirstPlace-BestPaper.pdf.

[10] F. Kotz *et al.*, "Three-dimensional printing of transparent fused silica glass," *Nat. 2017 5447650*, vol. 544, no. 7650, pp. 337–339, Apr. 2017, doi: 10.1038/nature22061.

[11] D. W. McOwen *et al.*, "3D-Printing Electrolytes for Solid-State Batteries," *Adv. Mater.*, vol. 30, no. 18, p. 1707132, May 2018, doi: 10.1002/ADMA.201707132.

[12] T. S. Wei, B. Y. Ahn, J. Grotto, and J. A. Lewis, "3D Printing of Customized Li-Ion Batteries with Thick Electrodes," *Adv. Mater.*, vol. 30, no. 16, p. 1703027, Apr. 2018, doi: 10.1002/ADMA.201703027.

[13] T. Erps *et al.*, "Accelerated discovery of 3D printing materials using data-driven multiobjective optimization," *Sci. Adv.*, vol. 7, no. 42, Oct. 2021, doi: 10.1126/SCIADV.ABF7435.

[14] N. Almakayeel, S. Desai, S. Alghamdi, and M. R. N. M. Qureshi, "Smart Agent System for Cyber Nano-Manufacturing in Industry 4.0," *Appl. Sci.*, vol. 12, no. 12, 2022, doi: 10.3390/app12126143.

[15] G. Haeberle and S. Desai, "Investigating Rapid Thermoform Tooling Via Additive Manufacturing (3d Printing)," *Am. J. Appl. Sci.*, vol. 16, no. 8, pp. 238–243, Oct. 2019, doi: 10.3844/ajassp.2019.238.243.

[16] L. Li, Q. Sun, C. Bellehumeur, and P. Gu, "Composite Modeling and Analysis for Fabrication of FDM Prototypes with Locally Controlled Properties," *J. Manuf. Process.*, vol. 4, no. 2, pp. 129–141, Jan. 2002, doi: 10.1016/S1526-6125(02)70139-4.

[17] H. Elhoone, T. Zhang, M. Anwar, and S. Desai, "Cyber-based design for additive manufacturing using artificial neural networks for Industry 4.0," *Int. J. Prod. Res.*, vol. 58, no. 9, pp. 2841–2861, May 2020, doi: 10.1080/00207543.2019.1671627.

[18] M. Ogunsanya, J. Isichei, S. K. Parupelli, S. Desai, and Y. Cai, "In-situ droplet monitoring of inkjet 3D printing process using image analysis and machine learning models," in *Procedia Manufacturing*, 2021, vol. 53, pp. 427–434, doi: 10.1016/j.promfg.2021.06.045.

[19] H. Almakaeel, A. Albalawi, and S. Desai, "Artificial neural network based framework for cyber nano manufacturing," *Manuf. Lett.*, vol. 15, pp. 151–154, Jan. 2018, doi: 10.1016/j.mfglet.2017.12.013.

[20] S. Desai, P. De, and F. Gomes, "Design for Nano/Micro Manufacturing: A Holistic Approach Towards Achieving Manufacturing Excellence," *J. Udyog Pragati* , vol. 39, no. 2, pp. 18–25, 2015.

[21] S. Desai and C. Dean, "Concurrent material and process selection in a flexible design for manufacture paradigm.," in *IIE Annual Conference. Proceedings*, 2007, p. 764.

[22] A. Yaseer, H. Chen, and B. Zhang, "Predicting Layer Roughness with Weaving Path in Robotic Wire Arc Additive Manufacturing Using Multilayer Perceptron," *2021 IEEE 11th Annu. Int. Conf. CYBER Technol. Autom. Control. Intell. Syst. CYBER 2021*, pp. 61–66, Jul. 2021, doi: 10.1109/CYBER53097.2021.9588272.

[23] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, Nov. 2020, doi: 10.1016/J.NEUCOM.2020.07.061.

[24] ASTM International, "Standard Test Method for Tensile Properties of Plastics." .

[25] V. Schaffer, B. Moyle, and K. Wishaw, *Artificial Neural Networks*. Edward Elgar Publishing, 2022.

[26] S. Abirami and P. Chitra, "Energy-efficient edge based real-time healthcare support system," *Adv. Comput.*, vol. 117, no. 1, pp. 339–368, Jan. 2020, doi: 10.1016/BS.ADCOM.2019.09.007.

[27] T. Menzies, E. Kocagüneli, L. Minku, F. Peters, and B. Turhan, "Using Goals in Model-Based Reasoning," *Shar. Data Model. Softw. Eng.*, pp. 321–353, Jan. 2015, doi: 10.1016/B978-0-12-417295-1.00024-2.

[28] H. Liu and I. Lee, "End-to-end trajectory transportation mode classification using Bi-LSTM recurrent neural network," in *Proceedings of the 2017 12th International Conference on Intelligent Systems and Knowledge Engineering, ISKE 2017*, 2017, vol. 2018-Janua, pp. 1–5, doi: 10.1109/ISKE.2017.8258799.

[29] Z. Liang, A. Powell, I. Ersoy, and M. Poostchi, "CNN-based image analysis for malaria diagnosis," *Ieeexplore.Ieee.Org*, pp. 8–11, 2016, Accessed: Feb. 16, 2023. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7822567/.

[30] M. W. Gardner and S. R. Dorling, "Statistical surface ozone models: An improved methodology to account for non-linear behaviour," *Atmos. Environ.*, vol. 34, no. 1, pp. 21–34, 2000, doi: 10.1016/S1352-2310(99)00359-3.

[31] D. Pérez-Marín, A. Garrido-Varo, and J. E. Guerrero, "Non-linear regression methods in NIRS quantitative analysis," *Talanta*, vol. 72, no. 1. pp. 28–42, 2007, doi: 10.1016/j.talanta.2006.10.036.

[32] S. HAYKIN, "Neural Networks: A Guided Tour," *Soft Comput. Intell. Syst.*, pp. 71–80, Jan. 2000, doi: 10.1016/B978-012646490-0/50007-X.

[33] H. Alibrahim and S. A. Ludwig, "Hyperparameter Optimization: Comparing Genetic Algorithm against Grid Search and Bayesian Optimization," in *2021 IEEE Congress on Evolutionary Computation, CEC 2021 - Proceedings*, 2021, pp. 1551–1559, doi: 10.1109/CEC45853.2021.9504761.

[34] M. Feurer and F. Hutter, "Hyperparameter Optimization," in *library.oapen.org*, 2019, pp. 3–33.

[35] P. I. Frazier, "Bayesian Optimization," *Recent Adv. Optim. Model. Contemp. Probl.*, pp. 255–278, Oct. 2018, doi: 10.1287/EDUC.2018.0188.

[36] P. P. Ippolito, "Hyperparameter Tuning," Springer, Cham, 2022, pp. 231–251.

[37] F. Hutter, J. Lücke, and L. Schmidt-Thieme, "Beyond Manual Tuning of Hyperparameters," *KI - Kunstl. Intelligenz*, vol. 29, no. 4, pp. 329–337, Jul. 2015, doi: 10.1007/s13218-015-0381-0.

[38] L. Zahedi, F. G. Mohammadi, S. Rezapour, M. W. Ohland, and M. H. Amini, "Search Algorithms for Automated Hyper-Parameter Tuning," Apr. 2021, doi: 10.48550/arxiv.2104.14677.

[39] B. H. Shekar and G. Dagnew, "Grid search-based hyperparameter tuning and classification of microarray cancer data," 2019, doi: 10.1109/ICACCP.2019.8882943.

[40] J. Bergstra, J. B. Ca, and Y. B. Ca, “Random Search for Hyper-Parameter Optimization Yoshua Bengio,” *J. Mach. Learn. Res.*, vol. 13, pp. 281–305, 2012, doi: 10.5555/2188385.

[41] R. Martinez-Cantin, “BayesOpt: A Bayesian Optimization Library for Nonlinear Optimization, Experimental Design and Bandits,” *J. Mach. Learn. Res.*, vol. 15, pp. 3735–3739, May 2014, doi: 10.48550/arxiv.1405.7430.

[42] P. R. Lorenzo, J. Nalepa, M. Kawulok, L. S. Ramos, and J. R. Pastor, “Particle swarm optimization for hyper-parameter selection in deep neural networks,” in *GECCO 2017 - Proceedings of the 2017 Genetic and Evolutionary Computation Conference*, Jul. 2017, vol. 8, pp. 481–488, doi: 10.1145/3071178.3071208.

[43] A. Alafaghani, A. Qattawi, B. Alrawi, and A. Guzman, “Experimental Optimization of Fused Deposition Modelling Processing Parameters: A Design-for-Manufacturing Approach,” *Procedia Manuf.*, 2017, doi: 10.1016/j.promfg.2017.07.079.

[44] I. J. Solomon, P. Sevvel, and J. Gunasekaran, “A review on the various processing parameters in FDM,” *Mater. Today Proc.*, vol. 37, no. Part 2, pp. 509–514, Jan. 2021, doi: 10.1016/J.MATPR.2020.05.484.

[45] J. J. Lifton, Z. J. Tan, B. Goh, and B. Mutiargo, “On the uncertainty of porosity measurements of additively manufactured metal parts,” *Meas. J. Int. Meas. Confed.*, vol. 188, 2022, doi: 10.1016/j.measurement.2021.110616.

[46] J. A. Slotwinski and E. J. Garboczi, “Porosity of additive manufacturing parts for process monitoring,” *AIP Conf. Proc.*, vol. 1581 33, pp. 1197–1204, 2014, doi: 10.1063/1.4864957.

[47] A. Aljohani and S. Desai, “3D Printing of Porous Scaffolds for Medical Applications,” *Am. J. Eng. Appl. Sci.*, vol. 11, no. 3, pp. 1076–1085, 2018, doi: 10.3844/ajeassp.2018.1076.1085.

[48] T. J. Gordelier, P. R. Thies, L. Turner, and L. Johanning, “Optimising the FDM additive manufacturing process to achieve maximum tensile strength: a state-of-the-art review,” *Rapid Prototyp. J.*, vol. 25, no. 6, pp. 953–971, Aug. 2019, doi: 10.1108/RPJ-07-2018-0183/FULL/T.J.GORDELIER.