electronics

MDPI

Article

Industry 4.0-Based Framework for Real-Time Prediction of Output Power of Multi-Emitter Laser Modules during the Assembly Process

Nikolaos Grigorios Markatos [1,*], Alireza Mousavi [1], Giulia Pippione [2] and Roberto Paoletti [2]

[1] College of Engineering, Design and Physical Sciences, Brunel University, Uxbridge UB8 3PH, UK
[2] Convergent Photonics Italia S.r.l, Via Schiaparelli 12, 10148 Torino, Italy
* Correspondence: nikolaos.markatos2@brunel.ac.uk

Abstract: The challenges of defects in manufacturing and assembly processes in optoelectronic industry continue to persist. Defective products cause increased time to completion (cycle time), energy consumption, cost, and loss of precious material. A complex laser assembly process is studied with the aim of minimising the generation of defective laser modules. Subsequently, relevant data were gathered to investigate machine learning and artificial intelligence methods to predict the output beam power of the module during the assembly process. The assembly process was divided into a number of chain steps, where we implemented a bespoke framework of hybrid feature selection method alongside artificial neural networks (ANNs) to formulate the statistical inferences. A review of existing learning methods in manufacturing and assembly processes enabled us to select XGBoost and random forest regression (RFR) as the two methods to be compared with ANN, based on their capabilities; ANN outperformed both of them, as it avoided overfitting and scored similar test metrics in the majority of the assembly steps. The results of the proposed solution have been validated in a real production dataset, even showing good predictive capability in the early steps of the assembly process where the available information is limited. Furthermore, the transferability of the framework was validated by applying the proposed framework to another product that follows a similar assembly process. The results indicated that the proposed framework has the potential to serve as the foundation for further research on laser modules' sophisticated and multi-step assembly lines.

Keywords: quality prediction; artificial neural network (ANN); XGBoost; random forest regression (RFR); multi-emitter laser; assembly; transferability; optoelectronics

check for updates

Citation: Markatos, N.G.; Mousavi, A.; Pippione, G.; Paoletti, R. Industry 4.0-Based Framework for Real-Time Prediction of Output Power of Multi-Emitter Laser Modules during the Assembly Process. *Electronics* **2023**, *12*, 766. https://doi.org/10.3390/electronics12030766

Academic Editor: Elias Stathatos

Received: 23 December 2022
Revised: 30 January 2023
Accepted: 31 January 2023
Published: 2 February 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

A major challenge in industry is effectively and efficiently maintaining production flow and product quality at minimum cost and waste reduction. Especially with the huge advances and demand for optoelectronics technologies, demand is on the rise. Due to the complexity and the volume of demand, higher customisation and individualisation is needed. As a result, the configurations of the system need to change more frequently and need to be efficient [1]. Especially in an assembly process of laser modules, the completion time is long, so the defect rate needs to stay at the minimal level. There are a number of defect-generating events in the laser production process: for example, the misplacement of mirrors and lenses, faulty glue application, and loss of calibration of the assembly machine.

Energy consumption refers to the energy that is spent during the assembly process and is significant for two reasons: environmental impact and total cost of the product. A higher probability of defect leads to huge amounts of energy being spent for products that in the end are defective. In this research, defective products will be considered laser modules with low output beam power (measured in Watts). The cost of production refers to the total cost of the whole process, and it includes material costs, machine running costs, set up time, and operator's salary. Material reusability refers to the ability to reuse the

materials that are part of a defect product, and it is related to waste and environmental impact. The sooner a product is identified as a potential defect, the higher the chance is that the materials that were used for the assembly of the product can be reused and not be sent for scrapping. This way, the cost of production and the environmental impact can be controlled.

During the 1990s, artificial neural networks were extensively studied in order to identify hidden patterns between input–output data; however, their application was limited due to the lack of computing power and the lack of large amounts of data [2]. For these reasons, a high level of automation could not be implemented in the manufacturing/assembly process in order to improve quality and minimise the completion time and generation of defects.

Nowadays, the small size and low cost of the Internet of Things (IoT) devices that have emerged give the capability of equipping manufacturing lines with a large number of IoT sensors, thus creating a large amount of data [2]. The implementation of IoT sensory networks in manufacturing shop floors has given the ability for continuous monitoring in the production process [3].

At the same time, there has been a significant advancement in computing power, which has enabled scientists to focus again on the research of ANNs [2]. ANNs have helped in the development of predictive process modelling systems [4], in manufacturing accuracy [5], in developing control models for the production processes [6], and in providing predictive analytics for the quality assurance of assembly processes [7], and they have been used for early quality classification and prediction [8].

With the evolution of Industry 4.0, machine learning has given the ability to deal with several industrial issues in real-time, such as defect detection [3], flow disruptions [9], real-time monitoring and predictive maintenance [10], and quality prediction [11] [12]. Furthermore, it has given the ability to build prediction models to act as the operator's aid in the diagnosis of faults [13]. In addition, machine learning has been implemented in problems with limited datasets in order to improve the accuracy of predictions [14]. Furthermore, the computational advancement in combination with the sufficient amount of data have led to the development of more complex networks known as deep neural networks (DNNs).

Deep learning algorithms are based on artificial neural networks (ANN) and are capable of automatically extracting higher-level characteristics from raw input data [15]. DNNs are ANNs with multiple layers between the input and output layers [16]. In the last two decades, many deep neural networks, such as the convolutional neural network (CNN) and recurrent neural network (RNN), have emerged [17]. CNNs have advanced the processing of images, video, speech, and audio, whereas RNNs have shed light on sequential data such as text and speech [18].

In the course of our investigation, we concentrated on a laser manufacturing firm that creates high-power multi-emitter laser modules. Their method of assembly is a complex and multi-step process, which results in the generation of defects that, in most cases, cannot be located or detected with ease. During the course of this study, a sophisticated prediction framework was constructed, with the goal of predicting the quality of the produced module, by predicting the final output power (measured in Watts) during the course of the assembly process. This way, we are able to take into account the entirety of the assembly process, provide an accurate interpretation of both it and the potential defects that it may yield, and, as a result, reduce the number of defects that are generated by providing early-stage predictions regarding the end product's quality. The error percentages assessed by the framework dropped from 5% in the earlier stages of the assembly process to 3.5% in the later ones.

Finally, we put this framework to the test on another product manufactured by the same company that adheres to a philosophy of assembly that is similar to the one that was used for the development of the proposed framework. The framework achieved similar errors (4.8% in the early stages and 4% in the later stages), indicating that this framework

can be applied to situations that are similar to the case demonstrated in this research as a proof of concept (as the same methodology in a similar product yielded low errors) by adjusting it appropriately.

Despite, the small- and high-dimensional datasets that we had at our disposal, we managed to efficiently utilise this information to our gain, creating a framework for early quality prediction that is robust and accurate. In previous research, we developed a hybrid method for feature selection (RReliefF-RFE) [19] that identifies the important information in the dataset and reduces its high-dimensionality, which negatively affects the performance of the prediction models. This method creates a subset of important variables on which, in this study, we applied deep ANN, XGBoost, and RFR prediction methods in order to choose the best candidate for the specific case. The reasons that these methods were chosen are:

XGBoost

1. Performs better than support vector machine (SVM) and decision tree (DT) algorithms [20].
2. It incorporates a regularisation term during the modelling process, which prevents overfitting.
3. XGBoost's capacity to process scattered data quickly and effectively has led to its widespread adoption in business intelligence modelling. As a consequence, this has resulted in overall excellent model performance [20].
4. In order to achieve improved prediction performance, XGBoost makes use of the concept of boosting ensemble learning algorithms [20].

As the studied case dealt with low sample datasets of high-dimensionality and sparsity, which can lead to overfitting, XGBoost appeared to be an appropriate candidate for tackling these problems.

Random Forest Regression (RFR)

1. RFR usually yields better results than the decision tree algorithm [20].
2. RF algorithms generate better results in high-dimensional non-linear problems [20,21].
3. RF algorithms maintain their accuracy even when a significant amount of the data is missing [22–24].

As the studied case focused on high-dimensional non-linear data, RFR appeared to be an ideal candidate for this research.

Artificial Neural Network (ANN)

1. ANNs can adapt without the user's assistance [25].
2. ANNs are capable of being utilised in the solution of difficult non-linear problems [22,26].
3. Due to iterative training and mapping of inputs to outputs, ANNs are able to learn by creating an input–output mapping for the problem [25].

ANNs can tackle the problem of non-linearity in the studied case. The ANN is capable of efficiently handling this kind of dataset, and for that reason, it was considered a capable candidate for the specific research.

The novelty of the present work is that an adaptable inline real-time prediction framework for optoelectronics assembly lines is proposed that can be applied to any low volume–high variety optoelectronics production system that follows similar assembly philosophy as the studied cases. To the best of our knowledge, these cases have not been studied before. The proposed framework achieves accurate early-stage prediction, making it a valuable tool especially for customised and demanding assembly lines.

2. Methods

2.1. Artificial Neural Network (ANN)

ANNs are composed of many "neurons," which are organised into three types of layers: input, hidden, and output. The input and output layers are the layers from which the inputs and outputs are introduced and delivered, respectively. As a result, the number of neurons corresponding to them is the same as the number of inputs and outputs describing the system. The hidden layers are critical, because the learning process takes place within

them primarily by transforming the inputs into high-dimensional nonlinear systems using activation functions [27]. An ANN illustration can be seen in Figure 1. The ANN shown in Figure 1 has n inputs, l hidden layers, and m nodes. The inputs are indicated as $I_1, I_2, \ldots, I_n$, the weights between the input layer and the hidden layer as $W_{h_{nm}}$, and the weights between the hidden layer and the output layer as W_{o_m}. The matrix W_h consists of all weights between the input layer and the hidden layer, and $W_{h_{nm}}$ represents the weight between the n^{th} input and the m^{th} node of the hidden layer. Similarly, the matrix W_o consists of all weights between the hidden layer and the output layer, and W_{o_m} represents the weight between the m^{th} node of the hidden layer and the output. Functions $A_n(H)$ where $n = 1,2, \ldots, l$ are called activation functions and take the linear combiner output of the node as the input [28]. Each hidden layer has its own activation function. The terms B_{h_m} and B_{o_m} represent the bias in the H_m node of the hidden layer and the output layer, respectively. B_{h_m} is also a matrix that consists of all the biases of the nodes in the hidden layer. The mathematical equation that describes the linear combiner output on the m^{th} node of the hidden layer is (Equation (1)):

$$H_m = B_{h_m} + \sum_{i=1}^{n} W_{h_{im}} I_i \quad (1)$$

where H_m is the linear combiner of the m^{th} node of the hidden layer, B_{h_m} is the bias of the m^{th} node of the hidden layer, $W_{h_{im}}$ is the weight between the i^{th} input and the m^{th} node of the hidden layer, and I_i is the i^{th} input and n is the number of inputs.

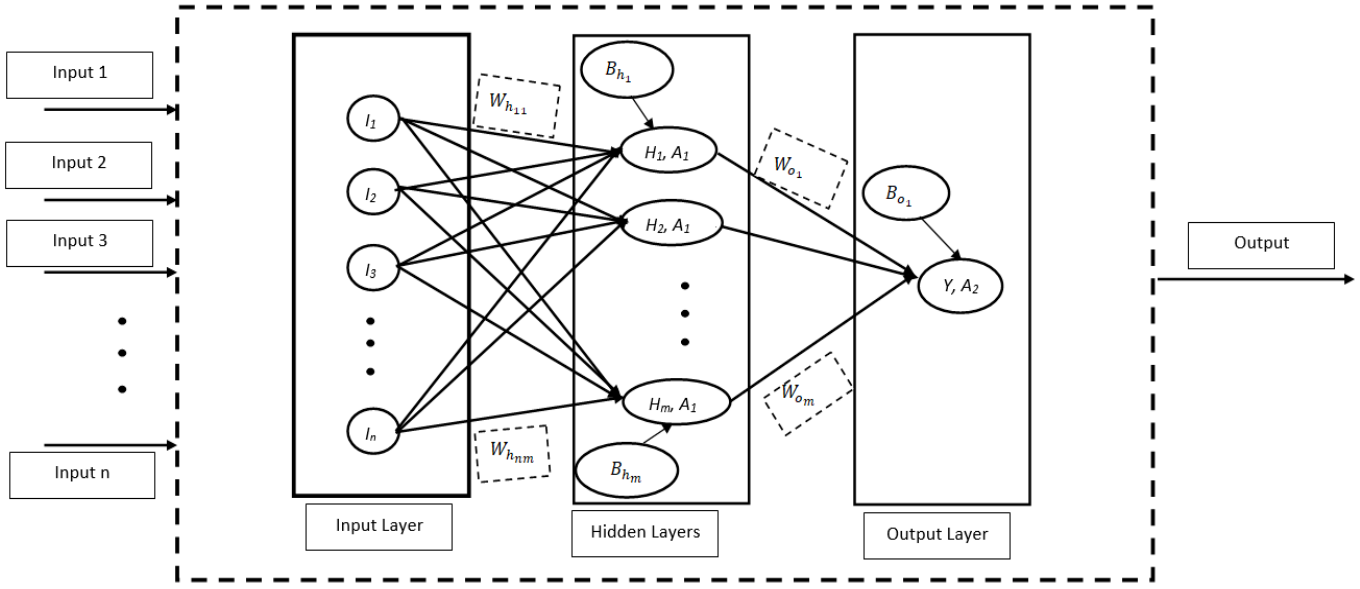

Figure 1. ANN architecture.

The activation function A_1 is then applied on the linear combiner output H_m, and the final output FH_m of the m^{th} node can be described by Equation (2) [27]:

$$FH_m = A_1(H_m) \quad (2)$$

The linear combiner output of Y can be described by Equation (3):

$$Y = B_{o_1} + \sum_{i=1}^{m} W_{o_i} FH_i \quad (3)$$

where B_{o_1} is the bias of the output layer's node, W_{o_i} is the weight between the i^{th} node and the output layer, FH_i is the final output of the i^{th} node of the hidden layer, and m are the nodes of the hidden layer.

In this case, as the model will be predicting only one value (output power of the laser module), the output layer has only one node. In other cases, the output can have multiple nodes (e.g., in the case of multi-class classification).

The final output of the ANN can be described by Equation (4) [27]:

$$Output = A_2(Y) \tag{4}$$

where A_2 is the activation function.

Activation functions A_n can take many forms, such as:

- Linear:

$$A_n(x) = ax + b \tag{5}$$

- Rectified Linear Unit (ReLU):

$$A_n(x) = \max(0, x) \tag{6}$$

- Sigmoid:

$$A_n(x) = \frac{1}{1 + e^{-x}} \tag{7}$$

In general, the linear activation function is used in the output layer for a regression problem. For $a = 1$ and $b = 0$, the linear activation becomes the identity function. ReLU gives x as the output if x is positive; otherwise, it gives zero as the output. Trial and error showed that the activation function that best suits the hidden layers for regression modelling is the rectified linear unit (ReLU), and the output layer for regression is the linear activation function. Trial and error refers to different experiments and combinations that are conducted in order to find the best activation function based on evaluation metrics such as RMSE and MAE. The sigmoid function is mostly used in the output layer for binary classification.

Through iterative training, the ANN adjusts the weights and bias until they give the best performance based on some predefined criteria (root-mean-square error, mean absolute error, etc.) [27]. To decide the optimal architecture of the ANN, hyperparameter tuning needs to take place. Hyperparameters are called the architectural parts of an ANN, such as nodes, hidden layers, dropout layers, batch size, learning rate, and activation function. A more detailed explanation of these hyperparameters along with the hyperparameter tuning will be presented in the next paragraphs.

2.2. XGBoost

XGBoost, developed by T. Chen et al. in 2016 [29], is an ensemble learning algorithm. It is among the most cutting-edge algorithms in the field of machine learning. Its fundamental concept is to successively train predictors, with each tree aiming to minimise the MSE of its predecessor until the error can no longer be decreased. Training an XGBoost model is therefore an iterative process [29]. Its mathematical representation is as follows (as presented in [30]):

Let us assume a dataset $D = \{(X_i, y_i)\} (|D| = n,\ X_i \in \mathbb{R}^m, y_i \in \mathbb{R})$ with n samples and m features. A tree-boosting model with S additive functions can be written as:

$$\hat{y}_i = \sum_{s=1}^{S} f_s(X_i), \quad f_s \in \mathrm{F} \tag{8}$$

where $\mathrm{F} = \left\{ f(X) = w_{q(X)} \right\} (q : \mathbb{R}^m \to T,\ w \in \mathbb{R})$ is the regression trees space, q describes the structure of each tree, and T is the leaves' number. Each f_s is associated with its own distinct tree of structure q and leaf weight w. Because of this, it is possible to arrive

at the final forecast by adding the predictions of each tree. The loss function (MSE) is described as:

$$l(y_i,\ \hat{y}_i) = (y_i - \hat{y}_i)^2 \tag{9}$$

In the objective function, besides the loss function, a regularisation term Ω is included in order to prevent overfitting. The regularisation term helps to smooth the final learned weights. Intuitively, the regularised objective tends to choose a model with simple and predictive functions of the model [29]. The objective function is written as (Equation (10)):

$$L = \sum_{i=1}^{n} l(y_i,\ \hat{y}_i) + \sum_{s=1}^{S} \Omega(f_s) \tag{10}$$

where $\Omega(f) = \gamma T + \frac{1}{\lambda} w^2$ and γ, λ are parameters that need to be fine-tuned. The tuning of these parameters was based on experimentation, and the methodology that was followed will be presented in the next chapters, where the tuning of the models is explained in detail.

The objective function may be changed such that it reads as follows in order to make a prediction about the i$^{\text{th}}$ instance at the t$^{\text{th}}$ iteration:

$$L^{(t)} = \sum_{i=1}^{n} l\left(y_i,\ \hat{y}_i{}^{(t-1)} + f_t(X_i)\right) + \Omega(f_t) + C \tag{11}$$

where $\Omega_{(t-1)}$ regularisation terms are considered as the constant C. By applying a second-order Taylor series expansion (Equation (12)):

$$f(x + \Delta x) \cong f(x) + f'(x)\Delta x + f''(x)\Delta x^2 \tag{12}$$

we can approximate Equation (11) as:

$$\widetilde{L}^{(t)} = \sum_{i=1}^{n} [g_i f_t(X_i) + \frac{1}{2} h_i f_t{}^2(X_i)] + \Omega(f_t) \tag{13}$$

where $g_i = \partial_{\hat{y}^{(t-1)}} l\left(y_i, \hat{y}^{(t-1)}\right)$ and $h_i = \partial^2_{\hat{y}^{(t-1)}} l\left(y_i, \hat{y}^{(t-1)}\right)$ are the first- and second-order gradient statistics on the loss function. C is not considered here, as it can be neglected. Equation (13) can be written as:

$$\begin{aligned} \widetilde{L}^{(t)} &= \sum_{i=1}^{n} \left[g_i f_t(X_i) + \frac{1}{2} h_i f_t^2(X_i) \right] + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \\ &= \sum_{j=1}^{T} \left[\left(\sum_{i \in I_j} g_i \right) w_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) w_j^2 \right] + \gamma T \\ &= \sum_{j=1}^{T} \left[G_j w_j + \frac{1}{2} (H_j + \lambda) w_j^2 \right] + \gamma T, \end{aligned} \tag{14}$$

where $G_j = \sum_{i \in I_j} g_i$ and $H_j = \sum_{i \in I_j} h_i$, and I_j is the instance set of leaf j, defined as $I_j = \{(i \mid q(X_i) = j\}$.

The optimal weight (Equation (15)) and the associated optimal objective function (Equation (16)) are, respectively:

$$w_j^* = -\frac{G_j}{H_j + \lambda} \tag{15}$$

$$\widetilde{L}^{(t)} = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T \tag{16}$$

After a split, the reduction in the objective function is given by Equation (17):

$$\begin{aligned} L_{\text{split}} =& -\frac{1}{2}\sum_{j=1}^{T}\frac{(G_L+G_R)^2}{H_L+H_R+\lambda}+\gamma T \\ & -\left[-\frac{1}{2}\sum_{j=1}^{T+1}\left(\frac{G_L^2}{H_L+\lambda}+\frac{G_R^2}{H_R+\lambda}\right)+\gamma(T+1)\right] \\ =& \frac{1}{2}\left[\frac{G_L^2}{H_L+\lambda}+\frac{G_R^2}{H_R+\lambda}-\frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right]-\gamma \end{aligned} \tag{17}$$

2.3. Random Forest Regression (RFR)

Random forest regression (RFR) is a method of ensemble learning that creates a large number of regression tree models using bootstrap samples of training data. Multiple decision trees are utilised in the RFR algorithm to prevent overfitting to training datasets [31]. Each tree is built from several bootstrapped datasets, with a random sample of n predictors being chosen as candidates from the full array selected at each split [32]. Thus, when performing a split, the chance of picking the same strong predictor variables is decreased, thereby preventing regression trees from becoming too correlated. Multiple regression tree predictors are knitted together to reduce prediction variance and enhance prediction accuracy. The method predicts the mean of all individual regression tree predictors' predictions [33]. In mathematical terms, the RFR model can be described as follows [34]:

A random forest is a collection of tree predictors that have the format $y(\boldsymbol{x};\theta_k)$, k = 1,…,N, where $\boldsymbol{x}$ is the observed input vector and θ_k are independent and identically distributed random vectors. The random forest prediction for regression is the unweighted average over the collection of all trees:

$$\overline{\mathrm{y}}(\mathbf{x}) = \frac{1}{\mathrm{N}}\sum_{k=1}^{N} y(\boldsymbol{x};\theta_k) \tag{18}$$

3. Use Case and Data Pre-Processing

3.1. Use Case

For the purpose of this study, the company provided us with datasets for two of their products that differ on the number of emitters, wavelength, and kind of package (s-serie and d-serie). The product d-serie was the one we focused on for designing our framework, as it had a more intricate architecture than the other product. In this manner, we would be able to determine whether or not our solution could be applied to the other product, therefore testing the transferability of the proposed framework to the assembly procedures of another optoelectronic product. The assembly process of a laser module consists of emitters, mirrors, and lenses, using spatial and polarisation multiplexing to combine the laser beams of the emitters into one unified beam focused on an output optical fiber (output beam). For the d-serie laser module, a combination of spatial and polarisation multiplexing lead to 62 steps (fast-axis collimating (FAC) placements, slow-axis collimating (SAC) placements, mirrors placements, turning mirror placement, polariser and fiber lens placement) in the specific assembly process. For the s-serie laser module, where only spatial multiplexing is used, there are 31 steps (FAC placements, SAC placements, mirror placements, and fiber lens placement). The difference in the process flow in the automatic assembly system of these two products lies in the turning mirror. In the d-serie module, two clusters of beams are firstly collimated and stacked together; then, the stacked beams are combined with a turning mirror and polarisation beam combiner and are redirected towards a unique focusing lens. A graphical representation of the assembly process for both products can be seen in Figures 2 and 3. Noteworthily, from this section onwards, some of the details of the assembly process and associated components are proprietary, and there are some limitations in sharing some of the information. Therefore, the case is explained in a wider context that could be applied to similar cases.

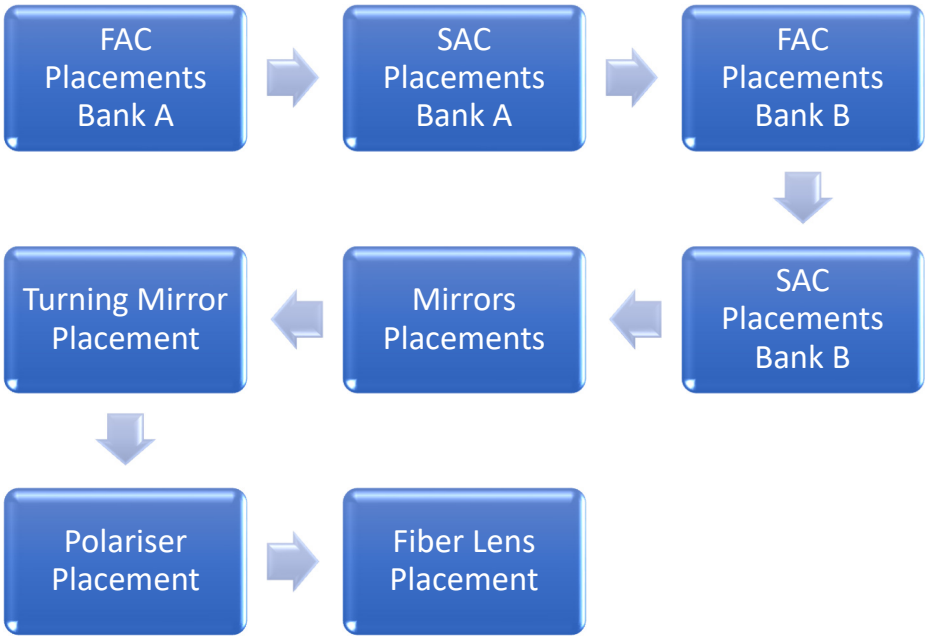

Figure 2. Assembly process from start to finish of the d-serie laser module.

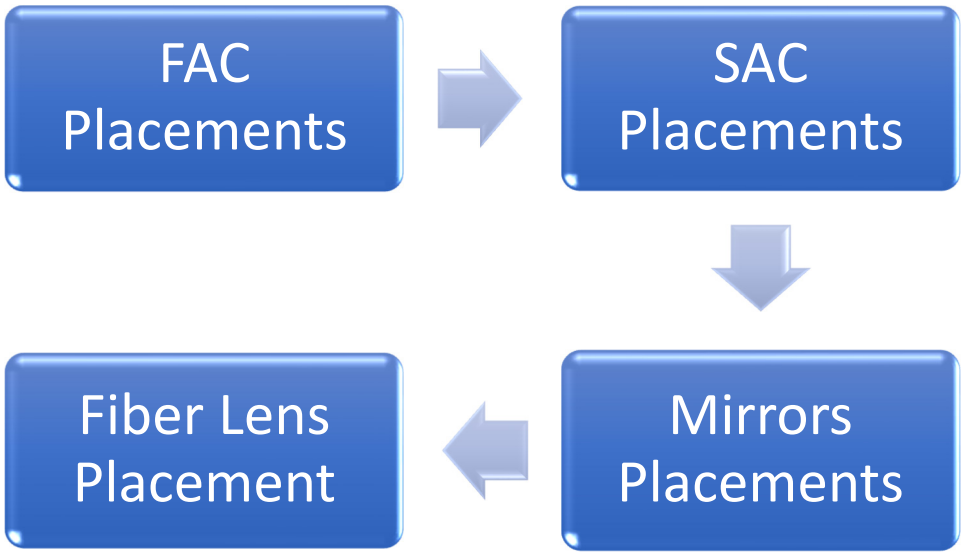

Figure 3. Assembly process from start to finish of the s-serie laser module.

The datasets that have been provided by the company contain information regarding each and every stage of the assembly line. All information mentioned for the dataset of the d-serie module applies just as well to the dataset of the s-serie module. The sole point of distinction is that the s-serie module does not have any turning mirror variables built into it because this module does not have a turning mirror. In addition, the quantity of data associated with every product is unique.

3.2. Datasets and Data Pre-Processing

3.2.1. D-Serie Laser Module

Regarding the d-serie module, two years of production information are used for modelling purposes. This length of time should allow for the stable data acquisition and flow from in-line and offline monitoring and control systems. Though this was not the case and due to the inconsistency and loss of data due to the volatility of the new deployed technologies, censored data points became unavoidable [35]. The data accumulated for

1411 modules and 1181 input variables were deemed suitable. Missing data were taken into consideration while making the decision on the suitability. For example, variables that exhibited a significant degree of inconsistency were ruled inappropriate for use. In addition, variables that were lacking a significant amount of data could not be used in this investigation. The modules were classified based on the categories that were provided by the company's expert. The quality of the output beam power of the modules produced were classified into defective, low, acceptable, good, very good and excellent. This was instead of the binary mode of pass and fail. The decision to use multi-class rather than binary classification was based on the company's needs, as the experts are not only interested in whether the module is operating or not, but also in its exact performance. This stratification helped us to evaluate the risk of removing jobs from the production line or devise real-time corrective actions to reduce waste and maximise recyclability of material. Although in the "Low" class the modules are operational, after discussions with the company's experts, these modules should be considered as "Fail" modules, as they do not adhere to the high-quality standards. All other classes ("Acceptable" to "Excellent") can be considered as "Pass" modules. Table 1 and Figure 4 show the spread of the 1411 modules in the five categories.

Table 1. Quality classes of the produced modules.

Class	Output Beam Power (W)	Number of Modules
Defective	<73	51
Low	73–80	357
Acceptable	80–83	309
Good	83–85	224
Very Good	85–90	408
Excellent	>90	62

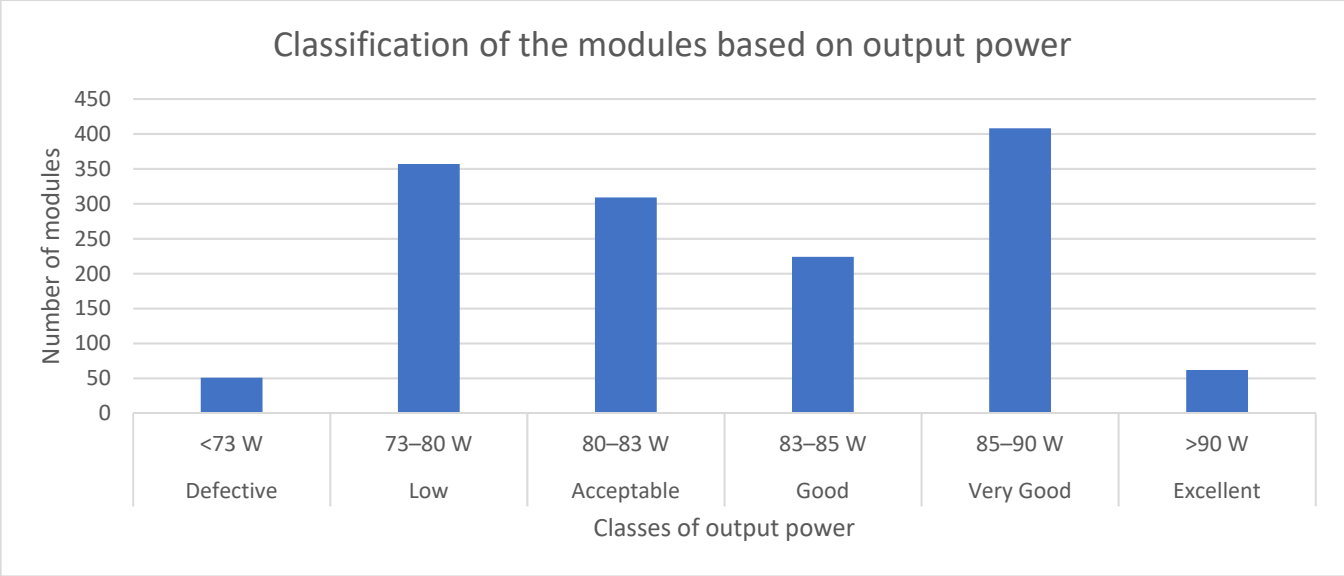

Figure 4. Classification of the d-serie laser modules based on output power.

As can be seen from Figure 4, the dataset is imbalanced. For the "Defective" category, we only have 51 modules out of 1411 (approximately 3.6% of the whole dataset), which might prove problematic for the prediction models as it will make the training on defective products harder. The same applies to the "Excellent" category, as it only has 62 modules (approximately 4.4% of the whole dataset). However, this is not seen as a significant issue from the company's point of view, as the distinction between "Very Good" and "Excellent" is not regarded as being as significant as the distinction between "Low" and "Defective."

This is due to the fact that the laser module maintains high quality requirements in both the "Very Good" and "Excellent" categories; however, the distinction between "Low" and "Defective" determines whether or not the module will properly function. This presents a challenge from a theoretical point of view due to the fact that the prediction models will have a bias toward the "Defective" and "Excellent" classes. The easiest and safest way to tackle this problem is to gather more data on these classes, although this takes time and cannot always be applied, especially in industrial settings where solutions need to be provided in a timely manner by utilising the information available at the time. For this study, this could not be applied for the reason that we had at our disposal the latest datasets from the assembly line and thus needed to wait for many months to collect new data, which was not feasible. Another way to tackle this problem is to generate "fake" data that resemble the "true" data through data upsampling techniques such as generative adversarial networks (GAN) [36] and deep convolutional generative adversarial networks (DCGAN) [37]. For instance, in 2022, Abu Ebayyeh et al. [38] faced the problem of an imbalanced dataset and solved it by creating synthetic images through the deployment of the DCGAN data upsampling technique. Although this appears to be the most convenient way to address the imbalance in our dataset, this was not feasible due to the fact that there was not an accurate way to validate whether the created synthetic data would resemble the true data. As our dataset consisted of many input variables that are all numerical, it was deemed impossible to create "fake" data based only on the sample of 51 modules of the "Defective" class. The relations between the inputs is complex; thus, altering some variables to resemble the ones of defective products is not possible as there is no "guidance" to conduct that process and the experts in the field cannot validate the "fake" data with high confidence. On the contrary, in image datasets, this can happen as the "fake" data are visualised (images), and thus an expert can confidently verify that these images resemble real ones. For instance, a defect can be recreated (low beams' intensity, dust etc.) and then validated by an expert in the field. In the numerical dataset, this is not the case. For that reason, the solution was based on the quality standards of the specific process. Modules in the "Acceptable" class and above preserve the high quality standards whilst the "Low" class modules do not. For example, if the framework predicted that a module would yield less than 73 W of output power, the company would immediately abort the assembly of that module. At the same time, if the framework predicted that the module would produce less than 80 W of output power, the company would consider aborting the assembly of the specific module, as this module would be of low quality. This indicates that the company's goal is to produce modules that are ranked in classes "Acceptable" and above in terms of their quality. As a consequence of this, a new categorisation of the modules was developed, which may be shown in Table 2 and Figure 5.

As can be seen from Figure 5, the new classification introduces more balance to the dataset and a statistically sufficient amount of data in each category, thus reducing the bias towards specific classes.

Table 2. New classification of the modules based on their output power.

Class	Output Beam Power (W)	Number of Modules
Low Quality	<80 W	408
Good Quality	80–85 W	533
High Quality	>85 W	470

Before the assembly begins, the emitters' power is tested in order to see whether they are working or not (laser voltage). After each step of the assembly process, measurements are taken regarding the geometry of the beams, their power, and their convergence to the centroid. Each assembly process (FAC, SAC, mirror, turning mirror, and polarisation) introduce new inputs to the problem. For example, during the FAC assembly, some of the important inputs that are measured are the width of the fast-axis beam alongside the centroid of the beam. During the SAC assembly, the pointing of the beam alongside the

width of both the fast-axis and slow-axis beams are measured. In addition to these, the power of the beam is also recorded during the SAC placement. A list of major inputs on each process, along with their respective units, is presented in Table 3.

Table 3. Important input variables of the d-serie laser module.

Variable Name	Assembly Process Step	Units
Laser Voltage	-	Volt
Centroid of the Beam	FAC	Pixel
Fast Axis Pointing Beam	SAC, Mirror	Pixel
Slow Axis Pointing Beam	SAC	Pixel
Fast Axis Width	SAC, Mirror	Pixel
Slow Axis Width	SAC, Mirror	Pixel
Centroid	Turning Mirror	Pixel
Power with Polariser	Polariser	Pixel
Sum Power of the Module (Output power)	-	Watt

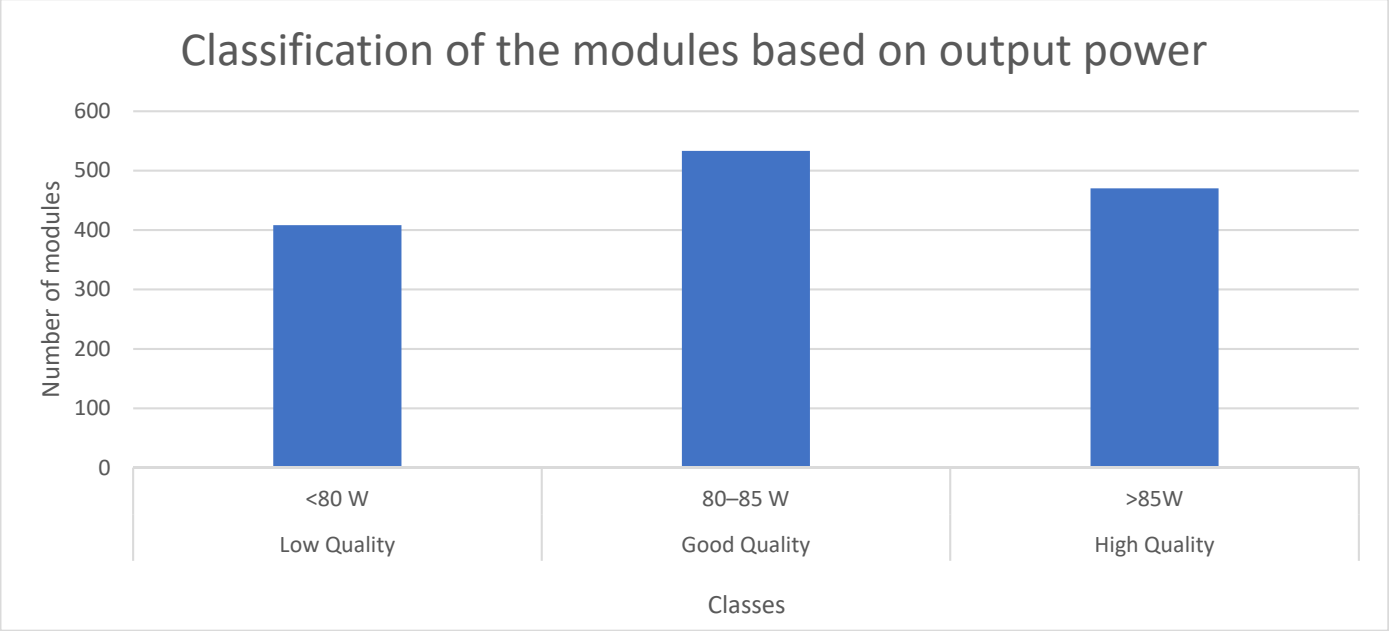

Figure 5. New classification of the modules based on their output power.

Before continuing to the next steps of the research, the scaling of inputs that was applied (between zero and one) as the difference in the range of values of the input variables was big. Variables of different ranges do not equally contribute to the outcome of the models (e.g., Input 1 in the range of 100–150 will outweigh Input 2, which is in the range of 0.01–0.1).

The data come from five assembly machines, and for that reason, shuffling was needed to avoid biased training (e.g., train the model with data from Machine 1 only). Scaling and shuffling improve the performance and reduce the bias of the model, respectively. From the 1411 modules that were selected, 1130 were used for training and testing (around 80% of the data). The remaining 281 modules were cut from the dataset and were treated as unseen data for the later validation of the models.

3.2.2. S-Serie Laser Module

The s-serie laser module is assembled using the same assembling philosophy as the d-serie laser module. In the s-serie module assembly, turning mirror and PCB steps are missing because the polarisation combination is not used for the product. There are a total of 31 stages in the assembly. After a semi-manual inspection of the dataset, 1628 modules

and 530 variables were deemed suitable for this research. The factors of appropriateness are the same as for the d-serie laser module. Due to the fact that the majority of the variables are identical to those that were measured for the d-serie module, the nature of the variables is the same. The ones relating to the turning mirror are the only variables that are not currently included, as they do not exist for that particular product. We utilised 1300 modules for training (about 80% of the dataset), and 328 modules were used for validation. These numbers were determined so that the training and validation processes could be carried out. Before continuing to the next steps of the research, the scaling of the inputs was applied (between zero and one), as the difference in the range of values of the input variables was big. The quality of the output beam power of the modules produced were classified as previously described into defective, low, acceptable, good, very good and excellent; this was instead of the binary mode of pass and fail. This stratification helped us to evaluate the risk of removing jobs from the production line or devise real-time corrective actions to reduce waste and maximise recyclability of material. Table 4 and Figure 6 show the spread of the 1628 modules in the five categories.

Table 4. Quality classes for the produced modules.

Class	Output Beam Power (W)	Number of Modules
Defective	<19	7
Low	19–21	82
Acceptable	21–23	24
Good	23–25	231
Very Good	25–26	974
Excellent	>26	310

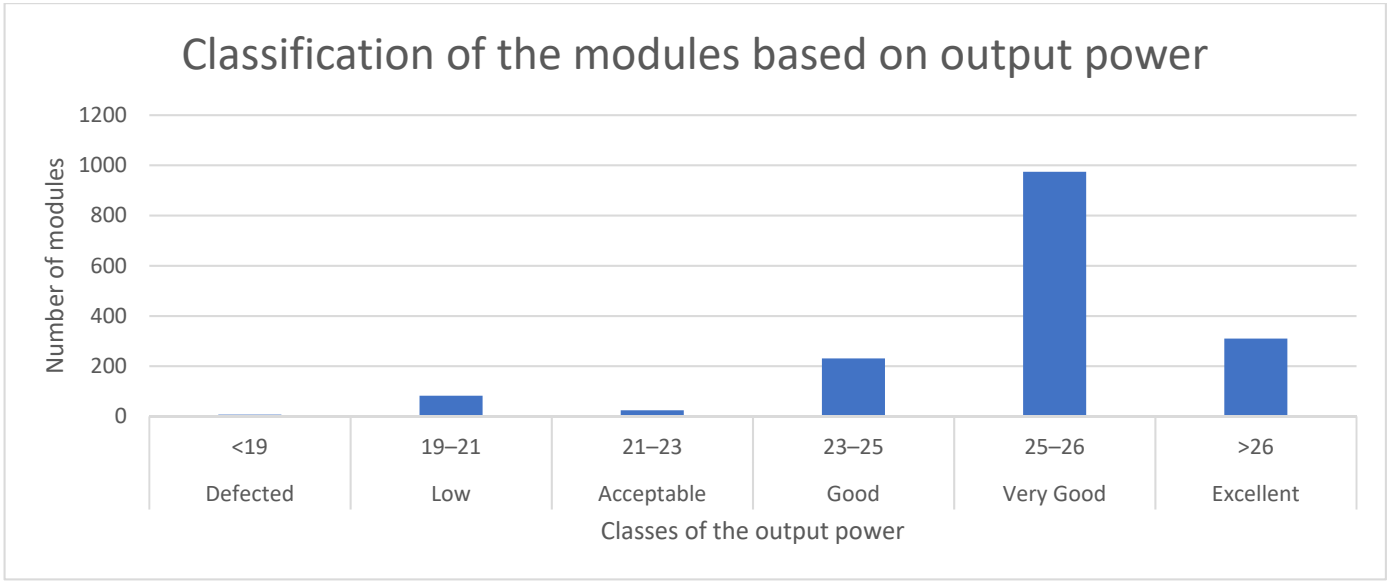

Figure 6. Classification of the s-serie laser modules based on output power.

The dataset is imbalanced, as can be seen from Figure 6 above. Out of a total of 1628 modules, only 113 were classified as belonging to the "Defective", "Low", and "Acceptable" categories. The "Very Good" class had 974 modules, making it the category with the greatest number of samples. Following the same steps as for the d-serie laser module, the six quality classes were transformed into three new ones, called *Low Quality*, *Good Quality*, and *High Quality*. For the d-serie laser module, the new classification introduced more balance to the dataset and a statistically sufficient amount of data in each category, thus reducing the bias towards specific classes. This is not something that applies to the s-series module, however. Even after the new classification of the modules into the three

newly introduced classes, the dataset remains imbalanced, as can be seen from Table 5 and Figure 7. This is interesting, as the data that are collected in many industrial scenarios are not balanced, which poses a great challenge for the generalisation of the proposed framework. In real situations, the proposed framework should be able to efficiently work and adapt to imbalanced datasets in order to consider it as a framework that can be generalised for products of similar assembly philosophy.

Table 5. New classification of the s-serie laser modules based on their output power.

Class	Output Beam Power (W)	Number of Modules
Low Quality	<21	89
Good Quality	21–25	255
High Quality	>25	1284

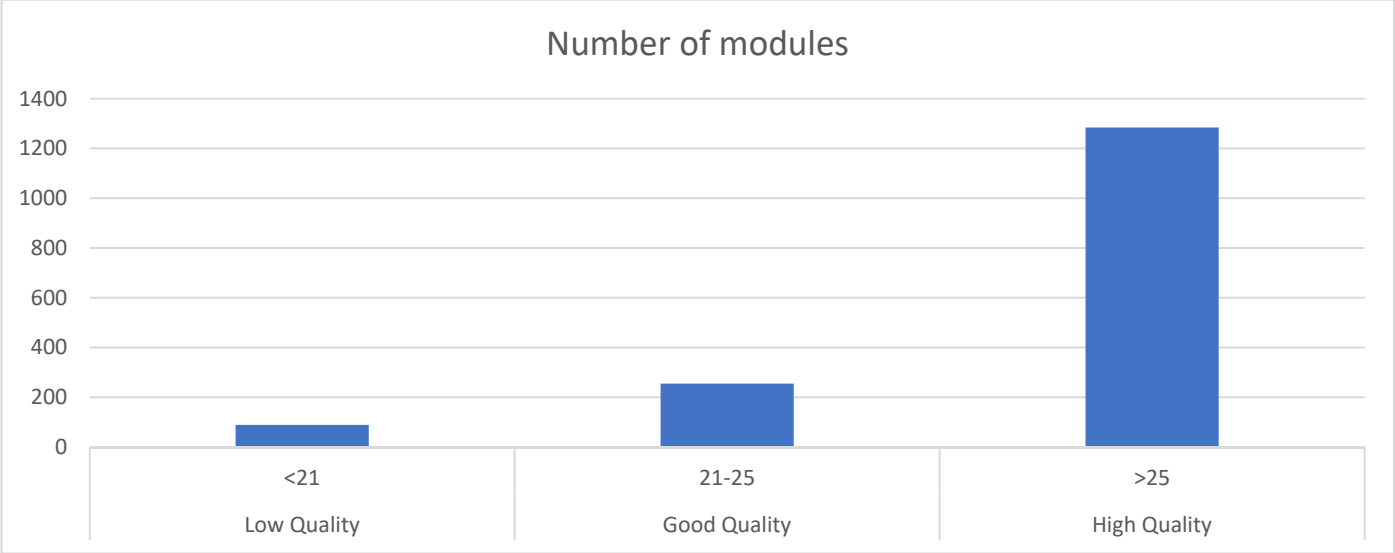

Figure 7. New classification of the s-serie laser modules based on their output power.

4. Implementation

The framework was completed with the deployment of the prediction models after the stage of feature selection had been concluded. There is a total of 62 steps involved in the assembly process of the d-serie laser module; however, there are some steps that have been determined to be particularly important, and a predictive model was implemented for each of these steps. The important steps of the procedure were determined by: (i) analysing the information that is added to the system at each stage of the assembly process because of the potential impact that this may have on the predictive capabilities of the model, (ii) analysing the nature of the assembly process and the steps, and (iii) consulting the company's experts regarding their view on the importance of each step. For instance, from Step 3 to Step 4, there was the addition of just one significant variable. This indicates that the predicted accuracy of the model would remain roughly equivalent to what it was in the stage before this one. Therefore, it would be inefficient to have one predictive model for each stage of the assembly process.

Instead, it was concluded that the information that has been provided at about every ten assembly steps is sufficient to make the process more accurate based on the amount of newly added important variables. Additionally, the nature of the process shifts every ten steps (for example, the installation of FACs and then placement of SACs), which indicates that the important steps act like indicators to the shifts that occur in the process. Furthermore, experts agreed that with every ten steps, a better overview of the process is given, as the different processes (FAC placement, SAC placement, etc.) are fully completed. Thus,

the crucial steps are: Step 10, Step 20, Step 30, Step 40, Step 50, Step 60, Step 61, and Step 62. For this reason, eight predictive models were built to cover the whole assembly process.

The inputs of each step were identified as the inputs of the current step combined with the inputs of all previous steps. This occurred in order to be able to accurately depict the evolution of the laser module throughout the assembly process and also efficiently explain the impact of the inputs to the output power. The output is common between all steps, and it is the final output power of the laser module.

A diagram of the assembly process alongside the stages that the predicted models were applied can be seen in Figure 8. As the assembly process proceeds forward, there are more inputs. For example, in Step 1, inputs are considered the variables that were identified as important during the feature selection and have to do with Step 1 (e.g., centroid of the beam). The output power of the module is considered as the output of Step 1. In Step 10, the inputs are considered the inputs of Step 10 combined with the inputs of all previous steps (Step 1–Step 9). The output in Step 10 is also the output power of the module.

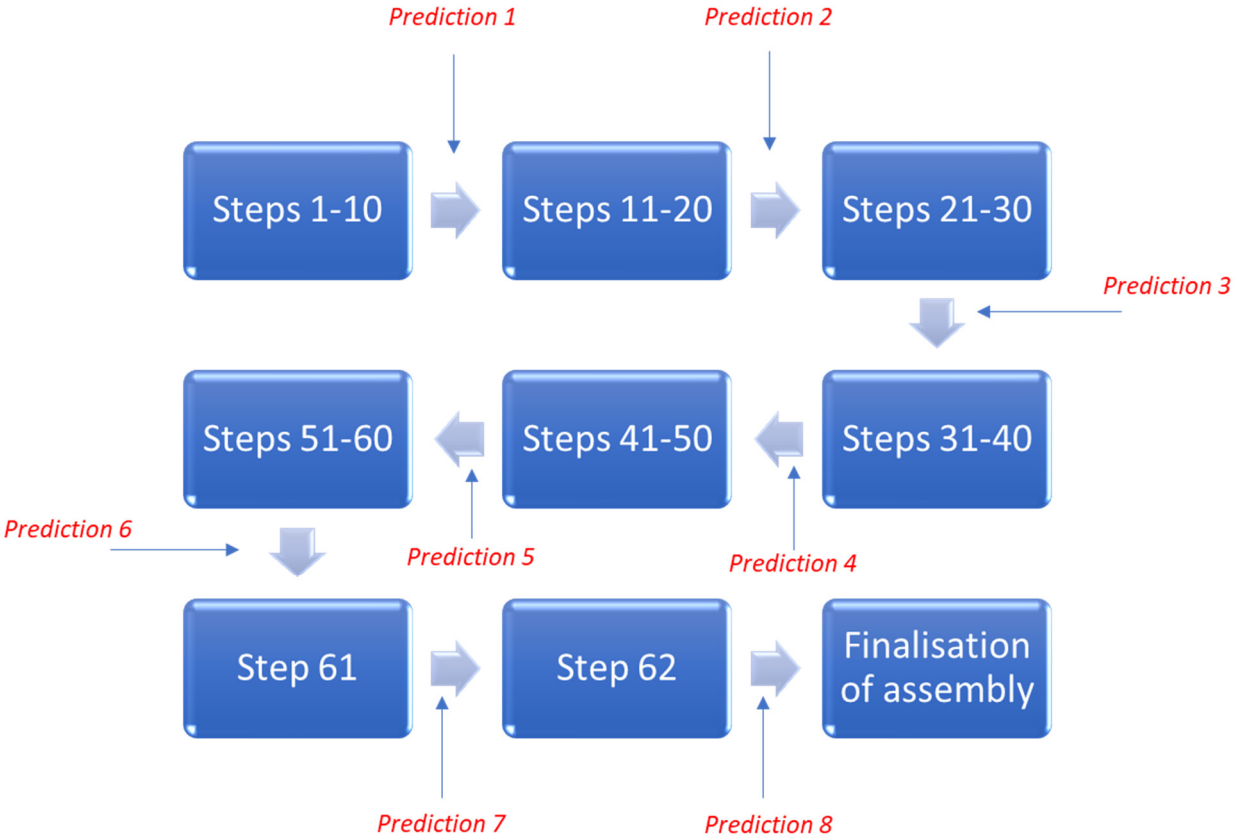

Figure 8. Assembly process flow and identification of crucial steps of the process.

The aforementioned way of identifying the crucial steps of the process may be applied to different cases, but the number of crucial steps alongside the stages on which they occur may vary. In the next chapter, where the generalisation of the model is discussed, the same philosophy for choosing the crucial steps will be followed.

In order to apply ML and AI solutions, the tuning of their hyperparameters needs to take place before the implementation of the models. Hyperparameters are parameters that are specified from the researcher before the training of the model, such as hidden layers, nodes, batch size, epochs, dropout, etc. Hyperparameters are different for each problem, and the choice of the proper combination lies on the researcher's experience and on trial and error.

Tuning is an essential process as it finds the optimal combination of the hyperparameters of the model for a specific case. These combinations are based on the case and nature

of the problem; therefore, for different cases, different hyperparameter combinations are needed. Hyperparameter tuning aims to improve the accuracy of the models by adjusting appropriate values to each hyperparameter through iterative attempts. A search space for the hyperparameters is provided, either by the experts in the field or through intuition and trial and error. In this research, the possible combinations were tested, and the combination that yielded better results based on two evaluation metrics, root-mean-square error (RMSE) and mean absolute error (MAE), was considered to be the optimal hyperparameter combination. Low RMSE and MAE scores indicate better performance. RMSE and MAE can be described by Equation (19) and Equation (20), respectively:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_{pred} - y_{real})^2} \tag{19}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_{pred} - y_{real}\right| \tag{20}$$

The optimal hyperparameter combinations for all of the prediction methods can be found in Appendix A for the d-serie module and in Appendix B for the s-serie module. A graphical representation of this process can be seen in Figure 9.

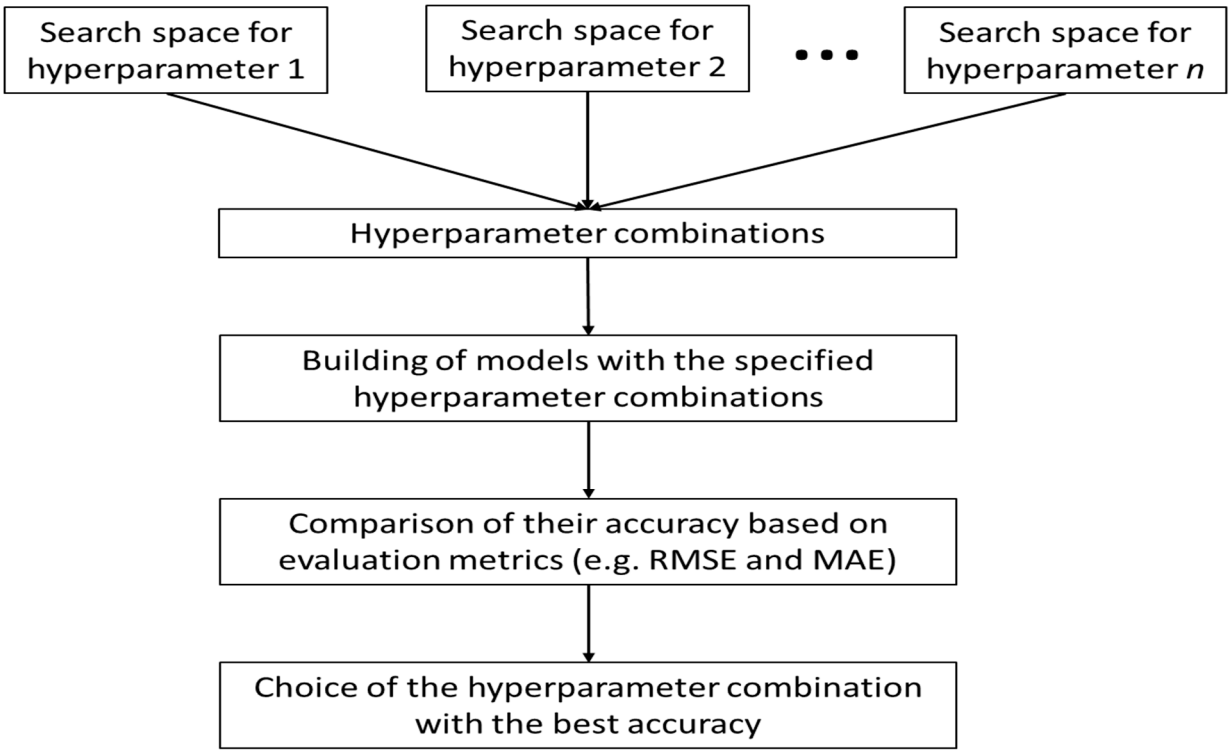

Figure 9. Hyperparameter tuning.

5. Results

The outcomes of using the XGBoost, RFR, and ANN algorithms on the specific subset are presented in the following paragraph. As explained in Section 3, there are eight crucial steps in the whole assembly process, and for that reason, eight prediction models were deployed in the framework. From now on, these models will be referred to as Model 1–Model 8. The detailed RMSE and MAE of the deployed methods can be found in Tables 6–8. A graphical representation for the RMSE and MAE for all methods, both for the training and the testing datasets, can be seen in Figures 10 and 11.

Table 6. Train and test RMSE and MAE for all RFR models.

RFR	Train RMSE	Train MAE	Test RMSE	Test MAE
Model 1	2.118	1.570	4.166	3.305
Model 2	2.453	1.811	3.920	3.061
Model 3	2.254	1.667	3.922	3.072
Model 4	1.182	0.760	3.881	3.014
Model 5	1.107	0.696	3.280	2.473
Model 6	0.913	0.577	3.198	2.434
Model 7	0.977	0.606	3.084	2.339
Model 8	0.338	0.149	2.953	2.214

Table 7. Train and test RMSE and MAE for all XGBoost models.

XGBoost	Train RMSE	Train MAE	Test RMSE	Test MAE
Model 1	3.215	2.505	4.142	3.320
Model 2	2.386	1.853	3.954	3.120
Model 3	2.459	1.896	3.882	3.067
Model 4	1.691	1.278	3.879	2.997
Model 5	1.558	1.173	3.120	2.330
Model 6	1.543	1.167	3.011	2.282
Model 7	1.448	1.091	2.834	2.109
Model 8	1.408	1.048	2.745	2.033

Table 8. Train and test RMSE and MAE for all ANN models.

ANN	Train RMSE	Train MAE	Test RMSE	Test MAE
Model 1	3.901	3.076	4.299	3.399
Model 2	3.794	2.949	4.066	3.162
Model 3	3.789	2.99	4.046	3.146
Model 4	3.457	2.69	3.782	2.926
Model 5	2.794	2.114	3.254	2.508
Model 6	2.499	1.914	3.171	2.387
Model 7	2.575	2.006	3.054	2.355
Model 8	2.558	1.991	2.858	2.223

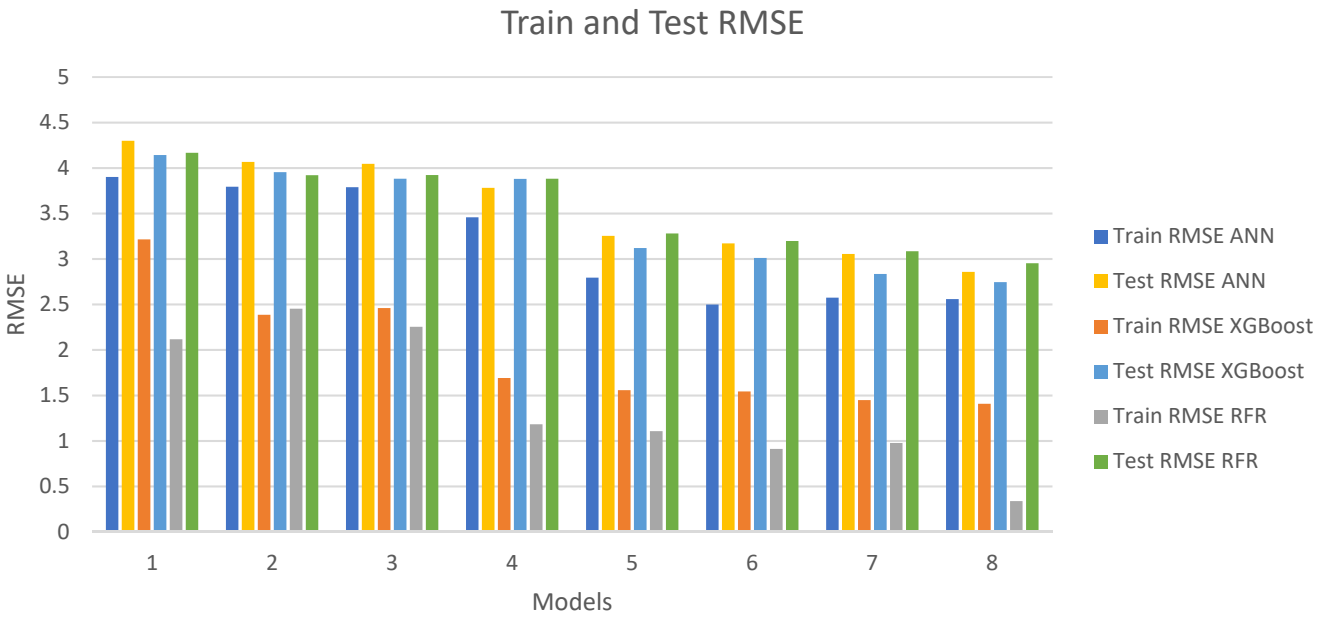

Figure 10. Train and test RMSE for all methods.

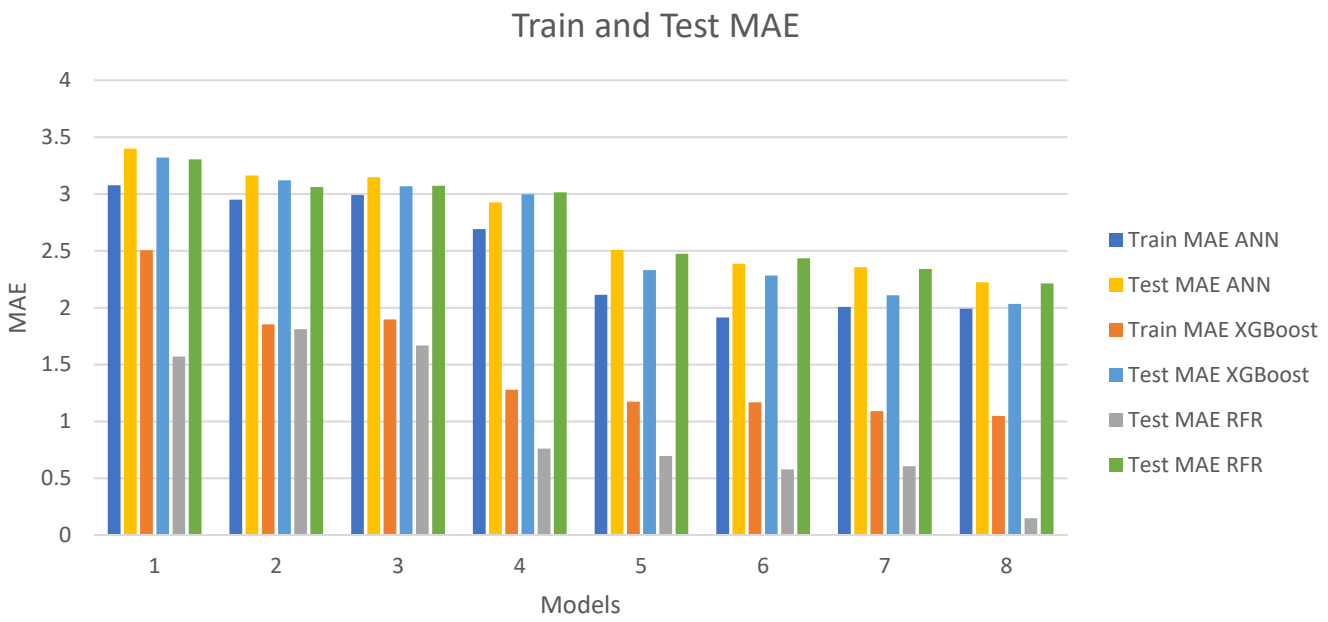

Figure 11. Train and test MAE for all methods.

In the specific dataset, the mean of the output variable are 82.330 W and 82.640 W for the train set and test set, respectively. To interpret the metrics, RMSE is convenient, as it has the same units as the output value and can be easily compared with the predicted value. This means that, practically, we could say that the final prediction will be expressed as "final prediction ± RMSE". For example, in Model 1, the ANN model had a test RMSE of 4.299. This means that the final prediction will have an error of approximately 5%. In Model 8, ANN has a test RMSE of 2.858. Accordingly, the final prediction will have an error of approximately 3.5%. The percentages of the errors for all of the deployed methods are presented in Figure 12. The results for all Models will be presented in the next paragraph.

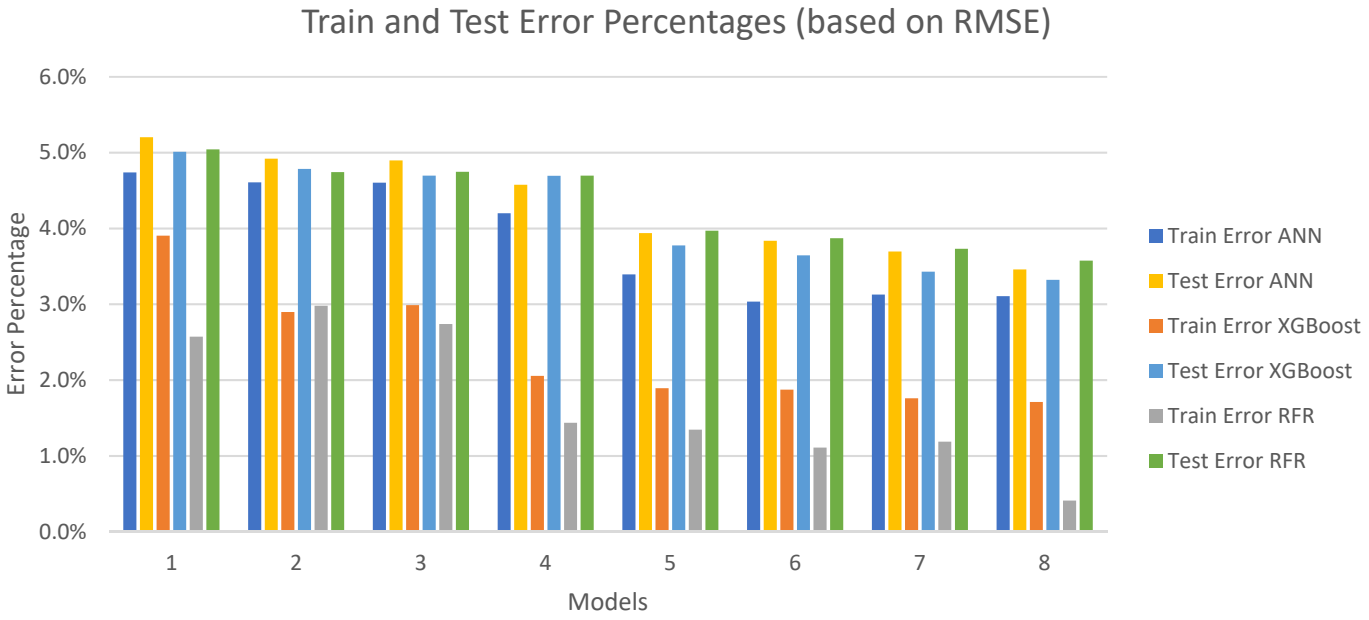

Figure 12. Train and test error percentage (based on the RMSE).

5.1. Model Analysis for D-Serie Module

5.1.1. Model 1

Model 1 is applied in Step 10 of the assembly process (finalisation of FAC placement in Bank A). Based on the RReliefF-RFE method, there are 15 important input variables that have been introduced so far in the process. Regarding the train RMSE and MAE, RFR scored the best values compared with the XGBoost and ANN algorithms. Regarding the Test RMSE and MAE, all three methods scored similar metrics. Regarding the train error percentage, RFR scored a 2.6% error while XGBoost and ANN scored 3.9% and 4.7%, respectively. In the test error percentage, RFR, XGBoost, and ANN scored 5%, 5%, and 5.2%, respectively.

In order to choose the most appropriate method to be used as a prediction algorithm for Model 1, the reliability of the model and how it can generalise in "unseen" data should be taken into consideration. This reliability of the model was also considered in the following seven models. A significant difference between the train and test metrics means that the model is overfitting. Overfitting is a common problem in machine learning problems. During the training of the model, instead of the model learning from the dataset, it "memorises" the dataset, hence the low train and high test metrics. That way, when "unseen" data are fed to the trained model, it cannot accurately predict (lack of generalisation).

More specifically, in Model 1, the only method that yielded close train and test error percentages was the ANN algorithm. In the XGBoost and RFR methods, the difference between the train and the test error percentages was 2.4% and 1.1%, respectively, whereas in ANN, the difference was 0.5%. This makes ANN trustworthy, as the small difference between train and test error percentages indicates that ANN can generalise well in "unseen" data. Furthermore, ANN scored test metrics that were similar to XGBoost and RFR. For these reasons, it seems the choice of ANN is satisfactory for Model 1. From a technical perspective, this model offers the company's expert an early-stage prediction for the output power of the module.

The error margin at this step was ± 4.299 *W*. This means that the prediction of the output power that is produced by the model will have an error of 4.299 W and the prediction in Step 10 will be:

$$FinalPrediction_{10} = (Prediction_{10} \pm 4.299),$$

where $FinalPrediction_{10}$ is the prediction that the expert should consider when making a choice whether to continue the assembly or not, $Prediction_{10}$ is the prediction that the model has produced in Step 10, and 4.299 is the Test RMSE of the model in Step 10.

5.1.2. Model 2

The application of Model 2 occurs at Step 20 of the assembly process (finalisation of SAC placement in Bank A). According to the RReliefF-RFE technique, there are a total of 31 significant input variables (15 of which are carried over from Model 1 and 16 of which are brand new) that have been brought into the equation up to this point in the assembly process. When compared with XGBoost and RFR, ANN scored slightly worse values for both the train RMSE and the train MAE. All three approaches achieved comparable results for the test RMSE and MAE metrics, with XGBoost and RFR achieving somewhat better results than ANN in this regard. RFR had a train error percentage of 3%, whereas XGBoost and ANN had 2.9% and 4.6%, respectively. RFR, XGBoost, and ANN scored 4.7%, 4.8%, and 4.9%, respectively, in the test error percentage.

As had occurred in Model 1, one can also observe that in Model 2, the only method that yielded close train and test error percentages was the ANN algorithm. In XGBoost and RFR, the difference between the train and test error percentages was 1.7% and 1.9%, respectively, whereas in ANN, the difference was 0.3%. This makes ANN trustworthy, as it is not overfitting. Moreover, ANN scored similar test metrics with XGBoost and RFR. For these reasons, it seems that the choice of ANN is satisfactory for Model 2.

The error margin at this step was ±4.066 W. This means that the prediction of the output power that is produced by the model will have an error of 4.066 W and the prediction in Step 20 will be:

$$FinalPrediction_{20} = (Prediction_{20} \pm 4.066),$$

where $FinalPrediction_{20}$ is the prediction that the expert should consider when making a choice of whether to continue the assembly or not; $Prediction_{20}$ is the prediction that the model has produced in Step 20, and 4.066 is the test RMSE of the model in Step 20.

5.1.3. Model 3

Model 3 is applied in Step 30 of the assembly process (finalisation of FAC placement in Bank B). Based on the RReliefF-RFE method, there are 43 important input variables (31 from Model 2 and 12 newly introduced) that have been introduced so far in the process. Regarding the train RMSE and MAE, the RFR method scored the best values compared with XGBoost and ANN. Regarding the test RMSE and MAE, all three methods scored similar metrics, with XGBoost and RFR scoring slightly better than ANN. Regarding the train error percentage, RFR scored a 2.7% error, whereas XGBoost and ANN scored 3% and 4.6%, respectively. In the test error percentage, RFR, XGBoost, and ANN scored 4.7%, 4.7%, and 4.9%, respectively.

As had happened in Model 1 and Model 2, one can also observe that in Model 3 the only method that yielded close train and test error percentages was ANN. In XGBoost and RFR, the difference between the train and the test error percentages was 2% and 1.7%, respectively, whereas in ANN, the difference was 0.3%. This makes ANN trustworthy, as it is not overfitting. Moreover, ANN scored similar test metrics with XGBoost and RFR. For these reasons, it seems the choice of ANN is satisfactory for Model 3.

The error margin at this step is ±4.046 W. This means that the prediction of the output power that is produced by the model will have an error of 4.046 W and the prediction in Step 30 will be:

$$FinalPrediction_{30} = (Prediction_{30} \pm 4.046),$$

where $FinalPrediction_{30}$ is the prediction that the expert should consider when making a choice of whether to continue the assembly or not. $Prediction_{30}$ is the prediction that the model has produced in Step 30, and 4.046 is the test RMSE of the model in Step 30.

5.1.4. Model 4

Model 4 is applied in Step 40 of the assembly process (finalisation of SAC placement in Bank B). Based on the RReliefF-RFE method, there are 52 important input variables (43 from Model 3 and 9 newly introduced) that have been introduced so far in the process. Regarding the train RMSE and MAE, the RFR method scored the best values compared with XGBoost and ANN. Regarding the test RMSE and MAE, all three methods scored similar metrics, with ANN scoring slightly better metrics than XGBoost and RFR. Regarding the train error percentage, RFR scored a 1.4% error, whereas XGBoost and ANN scored 2.1% and 4.2%, respectively. For the test error percentage, RFR, XGBoost, and ANN scored 4.7%, 4.7%, and 4.6%, respectively. In Model 4, RFR and XGBoost highly overfitted compared with the previous models. This is due to the fact that important information is introduced in Model 4, which probably contains "noise". XGBoost and RFR can train well, but without being able to distinguish the "noise". For that reason, they underperform when they are applied on "unseen" data. ANN, on the other hand, managed to distinguish the "noise" and generalise well in the test dataset.

One can observe in Model 4 that the only method that yielded close train and test error percentages was ANN. In XGBoost and RFR, the difference between the train and test error percentages was 3.3% and 2.6%, respectively, whereas in ANN, the difference was 0.4%. Furthermore, ANN performed well under the introduction of important information, which includes "noise". This makes ANN trustworthy, as it is not overfitting. Furthermore,

ANN scored better test metrics compared with XGBoost and RFR. For these reasons, it seems the choice of ANN is satisfactory for Model 4.

The error margin at this step is ±3.782 W. This means that the prediction of the output power that is produced by the model will have an error of 3.782 W and the prediction in Step 40 will be:

$$FinalPrediction_{40} = (Prediction_{40} \pm 3.782),$$

where $FinalPrediction_{40}$ is the prediction that the expert should consider when making a choice of whether to continue the assembly or not. $Prediction_{40}$ is the prediction that the model has produced in Step 40, and 3.782 is the test RMSE of the model in Step 40.

5.1.5. Model 5

Model 5 is applied in Step 50 of the assembly process (finalisation of mirror placement in Bank A). Based on the RReliefF-RFE method, there are 72 important input variables (52 from Model 4 and 20 newly introduced) that have been introduced so far in the process. As can be deducted, after Step 40, the information that is introduced is significant, as the amount of important variables being introduced is becoming larger. This is something that was also validated from the company's experts. Regarding the train RMSE and MAE, the RFR algorithm scored the best values compared with XGBoost and ANN.

Regarding the test RMSE and MAE, all three methods scored similar metrics, with XGBoost scoring slightly better metrics than ANN and RFR. Regarding the train error percentage, the RFR method scored a 1.3% error while XGBoost and ANN scored 1.9% and 3.4%, respectively. In the test error percentage, RFR, XGBoost, and ANN scored 4%, 3.8%, and 3.9%, respectively. In Model 5, RFR and XGBoost demonstrated severe overfitting, thus making these methods unreliable.

ANN demonstrated its ability to avoid overfitting by efficiently interpreting the newly information both on train and test datasets. Moreover, ANN scored similar test metrics to those obtained by XGBoost and better test metrics than RFR. For these reasons, it seems the choice of ANN is satisfactory for Model 5. As can be concluded so far, after Model 4, the only algorithm that can efficiently generalise is ANN. The fact is that it can interpret the "noise" in the dataset in a manner that does not affect its performance on "unseen" data.

The error margin at this step is ±3.254 W. This means that the prediction of the output power that is produced by the model will have an error of 3.254 W and the prediction in Step 50 will be:

$$FinalPrediction_{50} = (Prediction_{50} \pm 3.254),$$

where $FinalPrediction_{50}$ is the prediction that the expert should consider when making a choice of whether to continue the assembly or not. $Prediction_{50}$ is the prediction that the model has produced in Step 50, and 3.254 is the test RMSE of the model in Step 50.

5.1.6. Model 6

Model 6 is applied in Step 60 of the assembly process (finalisation of mirror placement in Bank B). Based on the RReliefF-RFE method, there are 91 important input variables (72 from Model 5 and 19 newly introduced) that have been introduced so far in the process. Regarding the train RMSE and MAE, the RFR algorithm scored the best values compared with XGBoost and ANN.

Regarding the test RMSE and MAE, all three methods scored similar metrics, with XGBoost scoring slightly better metrics than ANN and RFR. Regarding the train error percentage, RFR scored a 1.1% error while XGBoost and ANN scored 1.9% and 3% errors, respectively. In the test error percentage, RFR, XGBoost, and ANN scored 3.9%, 3.6%, and 3.8%, respectively. In Model 6, RFR and XGBoost demonstrate severe overfitting, thus making these methods unreliable.

ANN demonstrated its ability to avoid overfitting by efficiently interpreting the newly information both on train and test datasets. Moreover, ANN scored slightly worse test metrics than XGBoost but slightly better test metrics than RFR. The differences, however,

are small (approximately 0.17 worse than the XGBoost and 0.02 better than the RFR), making these differences insignificant. For these reasons, it seems the choice of ANN is satisfactory for Model 6. Again, as it was concluded in Model 5, the only algorithm that can efficiently generalise is ANN.

The error margin at this step is ±3.171 W. This means that the prediction of the output power that is produced by the model will have an error of 3.171 W and the prediction in Step 60 will be:

$$FinalPrediction_{60} = (Prediction_{60} \pm 3.171),$$

where $FinalPrediction_{60}$ is the prediction that the expert should consider when making a choice of whether to continue the assembly or not. $Prediction_{60}$ is the prediction that the model has produced in Step 60, and 3.171 is the test RMSE of the model in Step 60.

5.1.7. Model 7

Model 7 is applied in Step 61 of the assembly process (placement of the turning mirror). Based on the RReliefF-RFE method, there are 102 important input variables (91 from Model 6 and 11 newly introduced) that have been introduced so far in the process. Regarding the train RMSE and MAE, the RFR algorithm scored the best values compared with XGBoost and ANN.

Regarding the test RMSE and MAE, all three methods scored similar metrics, with XGBoost scoring slightly better metrics than ANN and RFR. ANN scored better test metrics than RFR. Regarding the train error percentage, RFR scored a 1.2% error while XGBoost and ANN scored 1.8% and 3.1%, respectively. In the test error percentage, RFR, XGBoost and ANN scored 3.7%, 3.4%, and 3.7%, respectively. In Model 7, RFR and XGBoost continue to demonstrate severe overfitting, thus making these methods unreliable.

In model 7, ANN was the only method that avoided overfitting and scored slightly worse test metrics than XGBoost but better test metrics than RFR. For these reasons, it seems that the choice of ANN is satisfactory for Model 7. Again, as it was concluded in Model 5 and Model 6, the only algorithm that can efficiently generalise is ANN.

The error margin at this step is ±3.054 W. This means that the prediction of the output power that is produced by the model will have an error of 3.054 W and the prediction in Step 61 will be:

$$FinalPrediction_{61} = (Prediction_{61} \pm 3.054),$$

where $FinalPrediction_{61}$ is the prediction that the expert should consider when making a choice of whether to continue the assembly or not. $Prediction_{61}$ is the prediction that the model has produced in Step 61, and 3.054 is the test RMSE of the model in Step 61.

5.1.8. Model 8

Model 8 is applied in Step 62 of the assembly process (placement of the polariser). Based on the RReliefF-RFE method, there are 123 important input variables (102 from Model 7 and 21 newly introduced) that have been introduced so far in the process. Regarding the train RMSE and MAE, the RFR method scored the best values compared with XGBoost and ANN.

Regarding the test RMSE and MAE, all three methods scored similar metrics, with XGBoost scoring slightly better metrics than ANN and RFR. ANN scored a better test RMSE than RFR and had a test MAE similar to RFR. Regarding the train error percentage, RFR scored a 0.4% error while XGBoost and ANN scored 1.7% and 3.1%, respectively. In the test error percentage, RFR, XGBoost, and ANN scored 3.6%, 3.3%, and 3.5%, respectively. In Model 8, RFR and XGBoost continue to demonstrate severe overfitting, thus making these methods unreliable.

As in Models 4–7, ANN is the only method that avoided overfitting in Model 8. ANN scored slightly worse test metrics than XGBoost and slightly better test metrics than RFR, but the differences again are considered insignificant. For these reasons it seems the choice of ANN is satisfactory for Model 8.

The error margin at this step is ±2.858 W. This means that the prediction of the output power that is produced by the model will have an error of 2.858 W and the prediction in Step 62 will be:

$$FinalPrediction_{62} = (Prediction_{62} \pm 2.858),$$

where $FinalPrediction_{62}$ is the prediction that the expert should consider when making a choice of whether to continue the assembly or not. $Prediction_{62}$ is the prediction that the model has produced in Step 62 and 2.858 is the test RMSE of the model in Step 62.

5.2. Summary of the Models' Outputs (D-Serie Module)

In Table 9, the results regarding the train and test errors of ANN, XGBoost, and RFR are presented. In the cases that ANN scored slightly worse metrics, the difference is considered insignificant, and for that reason, the criteria that were used to decide which prediction method addresses the studied case better was the criteria of overfitting. As can be concluded from Table 9, only ANN avoided overfitting while simultaneously scoring metrics that were similar to XGBoost and RFR.

Table 9. Train and test errors of ANN, XGBoost, and RFR.

	ANN				***XGBoost***				***RFR***			
	Train Error ANN	Test Error ANN	Difference	Overfitting	Train Error XGBoost	Test Error XGBoost	Difference	Overfitting	Train Error RFR	Test Error RFR	Difference	Overfitting
Model 1	4.7%	5.2%	0.5%	NO	3.9%	5.0%	1.1%	YES	2.6%	5.0%	2.5%	YES
Model 2	4.6%	4.9%	0.3%	NO	2.9%	4.8%	1.9%	YES	3.0%	4.7%	1.8%	YES
Model 3	4.6%	4.9%	0.3%	NO	3.0%	4.7%	1.7%	YES	2.7%	4.7%	2.0%	YES
Model 4	4.2%	4.6%	0.4%	NO	2.1%	4.7%	2.6%	YES	1.4%	4.7%	3.3%	YES
Model 5	3.4%	3.9%	0.5%	NO	1.9%	3.8%	1.9%	YES	1.3%	4.0%	2.6%	YES
Model 6	3.0%	3.8%	0.8%	NO	1.9%	3.6%	1.8%	YES	1.1%	3.9%	2.8%	YES
Model 7	3.1%	3.7%	0.6%	NO	1.8%	3.4%	1.7%	YES	1.2%	3.7%	2.5%	YES
Model 8	3.1%	3.5%	0.4%	NO	1.7%	3.3%	1.6%	YES	0.4%	3.6%	3.2%	YES

Table 10 shows the equations that the operator of the assembly machine should take into consideration to assess the accuracy of the prediction and determine whether the assembly process shall continue or not.

Table 10. Final prediction equations for Models 1–8.

	ANN
	Final Prediction (W)
Model 1	$FinalPrediction_{10} = (Prediction_{10} \pm 4.299)$
Model 2	$FinalPrediction_{20} = (Prediction_{20} \pm 4.066)$
Model 3	$FinalPrediction_{30} = (Prediction_{30} \pm 4.046)$
Model 4	$FinalPrediction_{40} = (Prediction_{40} \pm 3.782)$
Model 5	$FinalPrediction_{50} = (Prediction_{50} \pm 3.254)$
Model 6	$FinalPrediction_{60} = (Prediction_{60} \pm 3.171)$
Model 7	$FinalPrediction_{61} = (Prediction_{61} \pm 3.054)$
Model 8	$FinalPrediction_{62} = (Prediction_{62} \pm 2.858)$

5.3. Model Analysis for S-Serie Module

So far, an adaptable quality prediction framework for automated optoelectronics assembly lines has been presented. One main aspect of the industrial automation is to develop solutions that can be transferred to similar cases. For that reason, the proposed framework has been tested on a different product of the same company (s-serie laser module). This product is assembled in the same assembly machine but follows a different assembly recipe. The key change is that rather than having eight prediction models, there will be only four of them. This is as a result of the fact that the procedure of assembling the s-serie module requires just half as many steps as the assembly that was previously investigated. The methodology for determining which steps are most important adheres to the same format as in the past. Each significant stage marks the completion of a certain procedure (e.g., FAC placement). The following steps are especially important for this assembly: Step 10, Step 20, Step 25, and Step 30. Because the significant information gathered between Steps 20 and 30 is sufficient to make predictions even before all of the mirrors have been installed, two models were implemented instead of just one during the installation of the mirrors.

These four models will be called Model 1–Model 4. A graphical representation for the RMSE and MAE for all methods, both for the training and the testing datasets, can be seen in Figures 13 and 14. The detailed RMSE and MAE of the deployed methods can be found in Tables 11–13.

Figure 13. Train and test RMSE for all prediction methods.

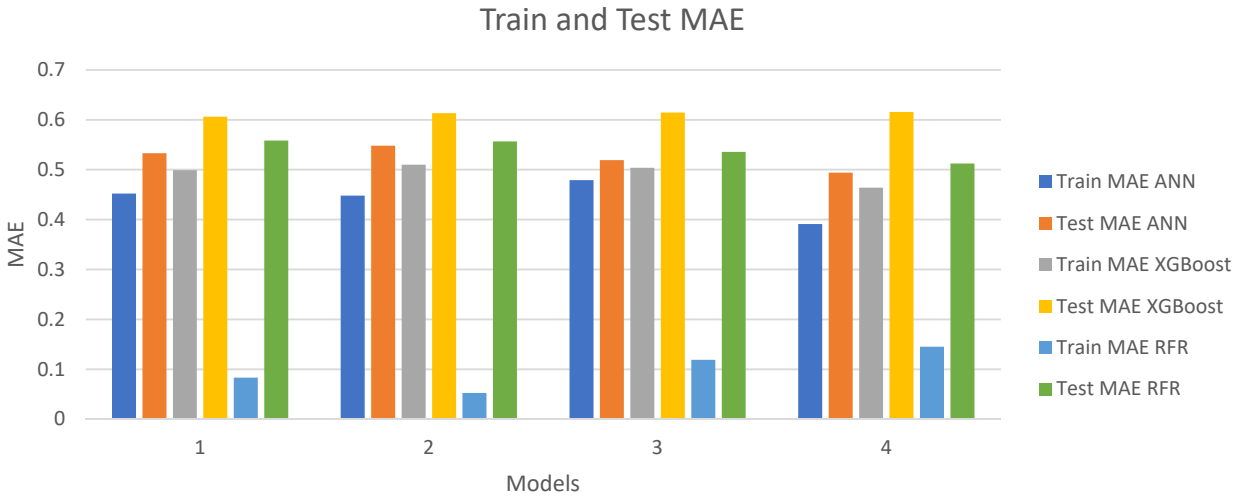

Figure 14. Train and test MAE for all prediction methods.

Table 11. Train and test RMSE and MAE for all ANN models.

ANN	Train RMSE	Train MAE	Test RMSE	Test MAE
Model 1	0.704	0.452	1.208	0.533
Model 2	0.643	0.448	1.139	0.548
Model 3	0.768	0.479	1.123	0.519
Model 4	1.367	0.391	1.106	0.494

Table 12. Train and test RMSE and MAE for all XGBoost models.

XGBoost	Train RMSE	Train MAE	Test RMSE	Test MAE
Model 1	0.896	0.499	1.227	0.606
Model 2	0.901	0.510	1.223	0.613
Model 3	0.889	0.504	1.238	0.614
Model 4	0.773	0.464	1.241	0.616

Table 13. Train and test RMSE and MAE for all RFR models.

Random Forest Regression	Train RMSE	Train MAE	Test RMSE	Test MAE
Model 1	0.395	0.083	1.188	0.558
Model 2	0.200	0.052	1.168	0.557
Model 3	0.388	0.119	1.156	0.535
Model 4	0.501	0.145	1.137	0.512

In the specific dataset, the mean of the output power of the modules are 25.230 W and 25.096 W for the train set and test set, respectively. To interpret the metrics, the error percentages based on the RMSE were used as was the case for the d-serie laser module. In Figure 15, the percentages of the errors for all deployed methods are presented.

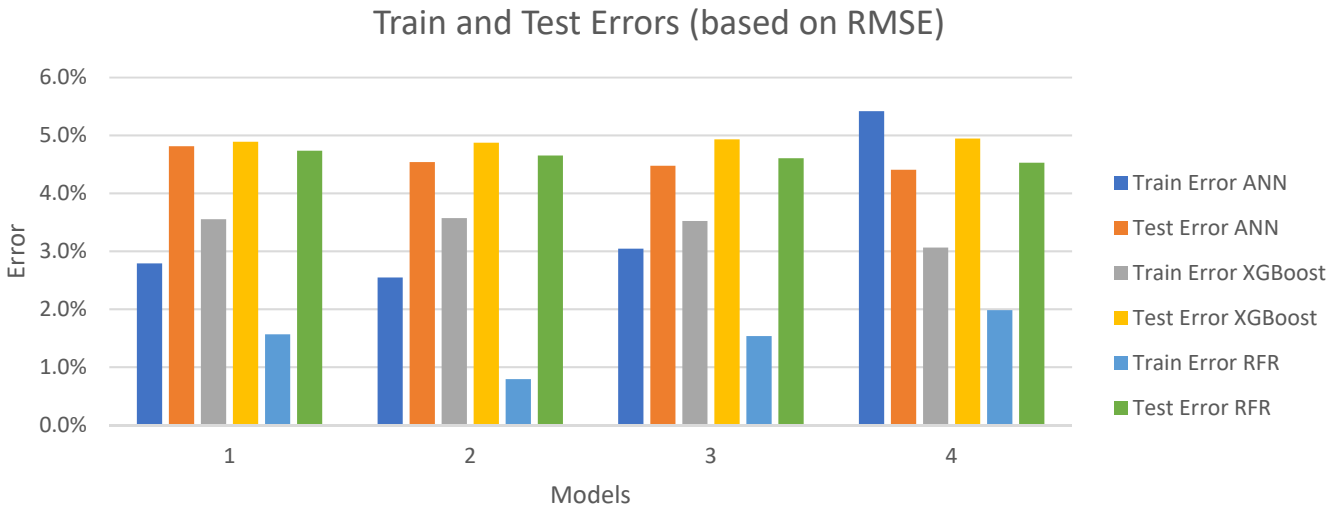

Figure 15. Train and test errors for all prediction methods.

5.3.1. Model 1

Model 1 is applied in Step 10 of the assembly process (finalisation of FAC placement). Based on the RReliefF-RFE method, there are 58 important input variables that have been introduced so far in the process. Regarding the train RMSE and MAE, the RFR algorithm scored the best values compared with XGBoost and ANN. Regarding the test RMSE and

MAE, all three methods scored similar metrics. Regarding the train error percentage, RFR scored 1.6% error while XGBoost and ANN scored 3.6% and 2.8%, respectively. In the test error percentage, RFR, XGBoost, and ANN scored 4.7%, 4.9%, and 4.8%, respectively.

In order to choose the most appropriate method to be used as a prediction algorithm for Model 1, the reliability of the model and how it can generalise in "unseen" data should be taken into consideration. This reliability of the model was also considered in the following three models.

More specifically, in Model 1, it can be seen that the RFR algorithm demonstrated severe overfitting. XGBoost and ANN slightly overfitted. This makes ANN and XGBoost trustworthy, as the small difference between the train and test metrics indicates that both methods can generalise well in "unseen" data. As ANN and XGBoost demonstrated similar overfitting, their test metrics were compared to decide which method to choose. ANN scored slightly better test metrics. For these reasons, it seems that the choice of ANN is satisfactory for Model 1.

The error margin at this step is ±1.208 W. This means that the prediction of the output power that is produced by the model will have an error of 1.208 W and the prediction in Step 10 will be:

$$FinalPrediction_{10} = (Prediction_{10} \pm 1.208),$$

where $FinalPrediction_{10}$ is the prediction that the expert should consider when making a choice of whether to continue the assembly or not. $Prediction_{10}$ is the prediction that the model has produced in Step 10, and 1.139 is the test RMSE of the model in Step 10.

5.3.2. Model 2

Model 2 is applied in Step 20 of the assembly process (finalisation of SAC placement). Based on the RReliefF-RFE method, there are 65 important input variables (58 from Model 1 and 7 newly introduced) that have been introduced so far in the process. Regarding the train RMSE and MAE, the RFR algorithm scored the best values compared with XGBoost and ANN. Regarding the test RMSE and MAE, ANN performed slightly better than XGBoost and RFR. Regarding the train error percentage, RFR scored a 0.8% error while XGBoost and ANN scored 3.6% and 2.5%, respectively. In the test error percentage, RFR, XGBoost, and ANN scored 4.7%, 4.9%, and 4.5%, respectively.

More specifically, in Model 2, it can be seen that RFR again demonstrated severe overfitting. XGBoost and ANN slightly overfitted. XGBoost scored similar test metrics as it did in Model 1, indicating that it cannot efficiently interpret the newly added information. For these reasons, it seems that the choice of ANN is satisfactory for Model 2.

The error margin at this step is ±1.139 W. This means that the prediction of the output power that is produced by the model will have an error of 1.139 W and the prediction in Step 20 will be:

$$FinalPrediction_{20} = (Prediction_{20} \pm 1.139),$$

where $FinalPrediction_{20}$ is the prediction that the expert should consider when making a choice of whether to continue the assembly or not. $Prediction_{20}$ is the prediction that the model has produced in Step 20, and 1.139 is the test RMSE of the model in Step 20.

5.3.3. Model 3

Model 3 is applied in Step 25 of the assembly process (half of the mirrors are placed). Based on the RReliefF-RFE method, there are 119 important input variables (65 from Model 2 and 54 newly introduced) that have been introduced so far in the process. As can be deducted, after Step 20, the information that is introduced is significant as the amount of important variables introduced is becoming larger; this is something that was also validated from the company's experts. Furthermore, it agrees with the research carried out for the 20 emitter laser module where, also during the placement of the Mirrors, the introduction of a significant amount of information regarding the assembly process was observed. Regarding the train RMSE and MAE, the RFR algorithm score the best values

compared with XGBoost and ANN. Regarding the test RMSE and MAE, ANN performs better than XGBoost and RFR. Regarding the train error percentage, RFR scored a 1.5% error while XGBoost and ANN scored 3.5% and 3%, respectively. In the test error percentage, RFR, XGBoost, and ANN scored 4.6%, 4.9%, and 4.5%, respectively.

More specifically, in Model 3, it can be seen that RFR again demonstrated severe overfitting. XGBoost and ANN slightly overfitted. XGBoost scored similar test metrics as it did in Model 1 and Model 2, indicating that after Model 1, it cannot efficiently interpret the newly added information. For these reasons, it seems that the choice of ANN is satisfactory for Model 3.

The error margin at this step is ±1.123 W. This means that the prediction of the output power that is produced by the model will have an error of 1.123 W and the prediction in Step 25 will be:

$$FinalPrediction_{25} = (Prediction_{25} \pm 1.123),$$

where $FinalPrediction_{25}$ is the prediction that the expert should consider when making a choice of whether to continue the assembly or not. $Prediction_{25}$ is the prediction that the model has produced in Step 25, and 1.123 is the test RMSE of the model in Step 25.

5.3.4. Model 4

Model 4 is applied in Step 30 of the assembly process (finalisation of mirrors placement). Based on the RReliefF-RFE method, there are 185 important input variables (119 from Model 3 and 66 newly introduced) that have been introduced so far in the process. Regarding the train RMSE and MAE, the RFR algorithm score the best values compared with XGBoost and ANN. Regarding the test RMSE and MAE, ANN outperformed XGBoost and RFR. Regarding the train error percentage, RFR scored a 2% error while XGBoost and ANN scored 3.1% and 5.4%, respectively. In the test error percentage, RFR, XGBoost, and ANN scored 4.5%, 4.9%, and 4.4%, respectively.

Regarding the train and test RMSE of Model 4, it is important to emphasise that there is no cause for concern despite the fact that the train RMSE is larger than the test RMSE. This might be supported as follows: during the training phase, many different regularisation layers were used in the model (batch normalisation and dropout), which made it "harder" for the model to train so that it could subsequently generalise in unseen data more effectively.

In Model 4, it can be seen that RFR again demonstrated severe overfitting. XGBoost also demonstrated slight overfitting. XGBoost scored similar test metrics as it did in Model 1, Model 2, and Model 3, indicating that after Model 1, it cannot efficiently interpret the newly added information. ANN performed well in train and test metrics and avoided overfitting. For these reasons, it seems that the choice of ANN is satisfactory for Model 4.

The error margin at this step is ±1.106 W. This means that the prediction of the output power that is produced by the model will have an error of 1.106 W and the prediction in Step 30 will be:

$$FinalPrediction_{30} = (Prediction_{30} \pm 1.106),$$

where $FinalPrediction_{30}$ is the prediction that the expert should consider when making a choice of whether to continue the assembly or not. $Prediction_{30}$ is the prediction that the model has produced in Step 30, and 1.106 is the test RMSE of the model in Step 30.

5.4. Summary of the Models' Outputs (S-Serie Module)

In Table 14, the results regarding the train and test errors of the ANN, XGBoost, and RFR methods are presented. As can be concluded, all methods overfitted, with XGBoost having the lowest overfitting problem, whereas RFR demonstrated the largest overfit. For this reason, RFR cannot be chosen as the best candidate for the specific problem. Although XGBoost overfitted less than ANN, it did not capture the introduced information at each model and thus did not interpret the assembly efficiently. This can be seen by the *XGBoost test error* not decreasing, even when new information was introduced into the assembly.

For the aforementioned reasons, the ANN algorithm was chosen as the optimal candidate for quality prediction on the specific problem.

Table 14. Train and test errors of ANN, XGBoost, and RFR.

	ANN				***XGBoost***				***RFR***			
	Train Error ANN	Test Error ANN	Difference	Overfitting	Train Error XGBoost	Test Error XGBoost	Difference	Overfitting	Train Error RFR	Test Error RFR	Difference	Overfitting
Model 1	2.8%	4.8%	2.0%	YES	3.6%	4.9%	1.3%	YES	1.6%	4.7%	3.2%	YES
Model 2	2.5%	4.5%	2.0%	YES	3.6%	4.9%	1.3%	YES	0.8%	4.7%	3.9%	YES
Model 3	3.0%	4.5%	1.4%	YES	3.5%	4.9%	1.4%	YES	1.5%	4.6%	3.1%	YES
Model 4	5.4%	4.4%	1.0%	YES	3.1%	4.9%	1.9%	YES	2.0%	4.5%	2.5%	YES

Table 15 shows the equations that the operator of the assembly machine should take into consideration to assess the accuracy of the prediction and determine whether the assembly process shall continue or not.

Table 15. Final prediction equations for Models 1–4.

	ANN
	Final Prediction (W)
Model 1	$FinalPrediction_{10} = (Prediction_{10} \pm 1.208)$
Model 2	$FinalPrediction_{20} = (Prediction_{20} \pm 1.139)$
Model 3	$FinalPrediction_{25} = (Prediction_{25} \pm 1.123)$
Model 4	$FinalPrediction_{30} = (Prediction_{30} \pm 1.106)$

5.5. Conclusions on Transferability of the Proposed Framework

Regarding the test RMSE, the ANN model outperforms RFR and XGBoost in all models except for Model 1, where the RFR algorithm scores slightly better RMSE than ANN. The difference is small (0.020), however, so it can be considered as insignificant. Regarding the test MAE, ANN outperforms XGBoost and RFR in all four models. Regarding the train metrics, RFR scored the lowest metrics in all four models; ANN came second. While ANN and XGBoost demonstrated just a reasonable degree of overfitting, RFR had severe overfitting problems. Due to the fact that RFR cannot be relied upon as a trustworthy model to implement, this means that it is no longer a viable option to explore as a potential choice. Additionally, RFR had test metrics that were higher than ANN's.

Concerning XGBoost, its test metrics did not change throughout the process, which indicates that XGBoost was unable to interpret the newly added information that was introduced in each step of the assembly. XGBoost will not be selected as the prediction technique for this framework because of this reason, as well as the fact that XGBoost's test metrics were higher than those of ANN and RFR. Because ANN overfitted to a reasonable degree, adequately captured and interpreted the newly added information, and scored lower test metrics during the whole assembly process, it was selected to be deployed in our proposed framework.

As it can be seen from the previous sections, the same research methodology was applied to a different product that follows the same assembly philosophy. The most successful outcomes were achieved by using the same techniques for both the feature selection and prediction components of the proposed framework. To be more explicit,

when it came to the feature selection process, RReliefF-RFE scored the best metrics and generated the best possible subset of variables to be utilised in the prediction phase later on. In the prediction component, the ANN algorithm outperformed XGBoost and RFR in terms of test metrics, process interpretation, and avoiding severe overfitting. This demonstrates that the suggested framework may be used for similar products; however, the architecture of the prediction model will need to be adjusted accordingly based on the case.

The generalisation of the solution is thus proven. This indicates that the framework that has been proposed has the potential to serve as the foundation for further research on laser modules' sophisticated and multi-step assembly lines.

6. Conclusions and Future Work

The purpose of this study was to design a framework for the quality prediction of automated optoelectronic assembly lines. These assembly lines are complex and contain a large number of assembly steps. As we have deduced up to this point, the most significant challenge that modern industries face is the requirement to offer a high level of customisation while still retaining a high level of quality standards. Because of this, early-stage quality prediction is required so that defects can be avoided and their impact reduced as much as possible. Our investigation centred on a multi-step manufacturing process for laser module components. Our aim was to predict the output power of the produced module throughout the assembly process. We utilised the RReliefF-RFE subset to make predictions regarding the output power of the module during the entirety of the assembly process by applying ML and AI algorithms. Throughout the entirety of the laser module assembly process, ANN, XGBoost, and RFR were utilised in an effort to ascertain which approach was capable of producing a more precise forecast of the output power of the laser module. According to the findings of our investigation, the application of ANN results in an accurate forecast of the output power, even in the early phases of the assembly, when there is a limited amount of information available. In addition to this, ANN avoids overfitting, which makes it a credible option for incorporation into our framework. Due to the fact that the method that has been deployed is capable of accurately linking the causal relationship that exists between inputs and outputs, it is possible to use it for efficient and real-time accurate predictions while the assembly process is taking place. This is evidenced by the low metrics that are recorded throughout this complex and sophisticated assembly process. Even at the early stages of the assembly process, the suggested framework showed promising results regarding the quality prediction for the given scenario. The error percentages assessed by the framework dropped from 5% in the earlier stages to 3.5% in the latter ones. This demonstrates that the framework is effective as, in an assembly line consisting of 62 steps, it only had a 5% mistake rate at Step 10, which is relatively low; hence, it can be trusted to make a decision on the future of the assembly, even from the early steps. Though the framework showed promising results, one important attribute of every industrial framework that also needs to be tested was the transferability of the framework. Because of this, we have applied the proposed framework to a different product made by the same company and evaluated the results. This product is put together using the same assembly machine; however, the assembly recipe that is used is different. It was found that using the same methods for both the feature selection and prediction aspects of the proposed framework yielded the most positive outcomes. The ANN algorithm scored better than both XGBoost and RFR in terms of test metrics, process interpretation, and the avoidance of severe overfitting. This illustrates that our proposed framework might be utilised for other products that are similar; however, the architecture of the prediction model will need to be appropriately modified depending on the specifics of the situation. This suggests that the framework that has been developed has the potential to act as the foundation for additional study on the complex assembly lines of laser modules, which include many steps. As future work, we intend to improve the framework in two aspects: the implementation of both numerical and image datasets and the further validation of the transferability of the framework.

Author Contributions: Conceptualisation, N.G.M. and A.M.; data curation, N.G.M.; formal analysis, N.G.M.; investigation, N.G.M.; methodology, N.G.M.; resources, G.P. and R.P.; software, N.G.M.; supervision, A.M.; validation, N.G.M., G.P., and R.P.; visualisation, N.G.M.; writing—original draft, N.G.M.; writing—review and editing, N.G.M. and A.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been carried out in the framework of the IQONIC project, which received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 820677.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing not applicable. The datasets are not publicly available due to confidentiality.

Acknowledgments: The authors wish to express their sincere thanks to Giulia Pippione and Roberto Paoletti for providing the databases to start with and for their expert knowledge on the subject.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Appendix A

Table A1. Optimal hyperparameters for the ANN models (d-serie laser module).

ANN	Learning Rate	Input Layer Nodes (Features)	Hidden Layer 1 Nodes	Dropout 1	Hidden Layer 2 Nodes	Dropout 2	Hidden Layer 3 Nodes	Dropout 3	Hidden Layer 4 Nodes	Dropout 4	Hidden Layer 5 Nodes	Dropout 5	Hidden Layer 6 Nodes	Dropout 6	Hidden Layer 7 Nodes
Model 1	*0.01*	*15*	*96*	*0.3*	*96*	*0.1*	*448*	*0.4*	*32*	*0.4*	*96*	//	//	//	//
Model 2	*0.01*	*31*	*256*	*0.4*	*448*	*0.4*	*384*	*0*	*192*	*0.1*	*288*	//	//	//	//
Model 3	*0.01*	*43*	*64*	*0.3*	*128*	*0.2*	*128*	*0.1*	*960*	*0.5*	*128*	//	//	//	//
Model 4	*0.01*	*52*	*64*	*0.5*	*544*	*0*	*768*	*0.4*	*608*	*0.3*	*64*	//	//	//	//
Model 5	*0.01*	*72*	*224*	*0.2*	*192*	*0*	*256*	*0.3*	*224*	*0.5*	*32*	*0.5*	*192*	//	//
Model 6	*0.01*	*91*	*160*	*0.3*	*544*	*0.2*	*832*	*0.3*	*640*	*0.3*	*480*	*0.5*	*512*	//	//
Model 7	*0.01*	*102*	*512*	*0.5*	*768*	*0.2*	*992*	*0.1*	*224*	*0.4*	*320*	*0.3*	*64*	*0.4*	*736*
Model 8	*0.01*	*123*	*480*	*0.4*	*128*	*0.3*	*512*	*0.3*	*32*	*0.3*	*32*	*0.3*	*352*	//	//

Table A2. Optimal hyperparameters for the XGBoost models (d-serie laser module).

XGBoost	Objective	eval_metric	colsample_bytree	Subsample	eta	max_depth	min_child_weight	reg_alpha	reg_lambda	Gamma
Model 1	reg:squarederror	RMSE	0.3	0.5	0.100	13	19	0.2	1.0	20
Model 2	reg:squarederror	RMSE	0.8	0.8	0.005	10	18	0.6	0.7	20
Model 3	reg:squarederror	RMSE	0.2	0.8	0.010	16	19	0.1	0.7	20
Model 4	reg:squarederror	RMSE	0.7	0.9	0.010	14	12	0.0	1.0	20
Model 5	reg:squarederror	RMSE	0.5	0.9	0.005	11	17	0.0	1.0	20
Model 6	reg:squarederror	RMSE	0.2	0.9	0.005	17	18	0.3	0.2	20
Model 7	reg:squarederror	RMSE	0.6	0.7	0.005	10	15	1.0	1.0	20
Model 8	reg:squarederror	RMSE	0.8	0.9	0.010	12	16	0.8	1.0	20

Table A3. Optimal hyperparameters for the RFR models (d-serie laser module).

RFR	Bootstrap	Max Depth	Max Features	Min Samples Leaf	Min Samples Split	n_Estimators
Model 1	TRUE	40	sqrt	2	5	1800
Model 2	TRUE	None	sqrt	4	2	800
Model 3	TRUE	110	sqrt	2	10	800
Model 4	FALSE	30	sqrt	2	10	1600
Model 5	FALSE	60	sqrt	4	5	1000
Model 6	FALSE	90	sqrt	2	10	1000
Model 7	FALSE	90	sqrt	4	5	1000
Model 8	FALSE	90	sqrt	2	2	200

Appendix B

Table A4. Optimal hyperparameters for the ANN models (s-serie laser module).

ANN	Learning Rate	Input Layer Nodes (Features)	Hidden Layer 1 Nodes	Dropout 1	Hidden Layer 2 Nodes	Dropout 2	Hidden Layer 3 Nodes	Dropout 3	Hidden Layer 4 Nodes	Dropout 4	Hidden Layer 5 Nodes	Dropout 5	Hidden Layer 6 Nodes	Dropout 6	Hidden Layer 7 Nodes
Model 1	0.01	58	362	0.3	10	0.1	298	//	//	//	//	//	//	//	//
Model 2	0.01	65	234	0.5	522	0.5	10	0.1	714	//	//	//	//	//	//
Model 3	0.01	119	234	0.5	522	0.5	10	0.1	714	//	//	//	//	//	//
Model 4	0.01	185	1386	0.5	618	0.1	618	0.5	298	//	//	//	//	//	//

Table A5. Optimal hyperparameters for the RFR models (s-serie laser module).

RFR	Bootstrap	Max Depth	Max Features	Min Samples Leaf	Min Samples Split	n_Estimators
Model 1	FALSE	70	sqrt	2	2	1800
Model 2	FALSE	40	sqrt	1	5	400
Model 3	FALSE	100	sqrt	1	10	400
Model 4	FALSE	60	sqrt	2	10	400

Table A6. Optimal hyperparameters for the XGBoost models (s-serie laser module).

XGBoost	Objective	eval_metric	colsample_bytree	Subsample	eta	max_depth	min_child_weight	reg_alpha	reg_lambda	Gamma
Model 1	reg:squarederror	RMSE	0.4	0.9	0.005	10	17	1	0	20
Model 2	reg:squarederror	RMSE	0.4	0.9	0.05	17	16	0	1	20
Model 3	reg:squarederror	RMSE	0.7	0.9	0.005	14	17	0	0	20
Model 4	reg:squarederror	RMSE	0.7	0.9	0.05	10	10	0.5	0	20

References

1. IQONIC. European Project. Available online: https://www.iqonic-h2020.eu/ (accessed on 24 April 2022).
2. Shah, D.; Wang, J.; He, Q.P. Feature engineering in big data analytics for IoT-enabled smart manufacturing—Comparison between deep learning and statistical learning. *Comput. Chem. Eng.* **2020**, *141*, 106970. [CrossRef]
3. Dimitriou, N.; Leontaris, L.; Vafeiadis, T.; Ioannidis, D.; Wotherspoon, T.; Tinker, G.; Tzovaras, D. A Deep Learning framework for simulation and defect prediction applied in microelectronics. *Simul. Model. Pract. Theory* **2019**, *100*, 102063. [CrossRef]
4. Petri, K.L.; Billo, R.E.; Bidanda, B. A neural network process model for abrasive flow machining operations. *J. Manuf. Syst.* **1998**, *17*, 52–64. [CrossRef]
5. Warnecke, G.; Kluge, R. Control of tolerances in turning by predictive control with neural networks. *J. Intell. Manuf.* **1998**, *9*, 281–287. [CrossRef]
6. Institute of Electrical and Electronics Engineers. Advanced Process Defect Monitoring Model and Prediction Improvement by Artificial Neural Network in Kitchen Manufacturing Industry: A Case of Study. In Proceedings of the 2019 IEEE Workshop on Metrology for Industry 4.0 and Internet of Things, Naples, Italy, 4–6 June 2019.
7. Burggräf, P.; Wagner, J.; Heinbach, B.; Steinberg, F.; Pérez M., A.R.; Schmallenbach, L.; Garcke, J.; Steffes-Lai, D.; Wolter, M. Predictive analytics in quality assurance for assembly processes: Lessons learned from a case study at an industry 4.0 demonstration cell. *Procedia CIRP* **2021**, *104*, 641–646. [CrossRef]
8. Stock, S.; Pohlmann, S.; Günter, F.J.; Hille, L.; Hagemeister, J.; Reinhart, G. Early Quality Classification and Prediction of Battery Cycle Life in Production Using Machine Learning. *J. Energy Storage* **2022**, *50*, 104144. [CrossRef]
9. Brik, B.; Bettayeb, B.; Sahnoun, M.; Duval, F. Towards Predicting System Disruption in Industry 4.0: Machine Learning-Based Approach. *Procedia Comput. Sci.* **2019**, *151*, 667–674. [CrossRef]
10. Xu, Y.; Sun, Y.; Liu, X.; Zheng, Y. A Digital-Twin-Assisted Fault Diagnosis Using Deep Transfer Learning. *IEEE Access* **2019**, *7*, 19990–19999. [CrossRef]
11. Chou, P.-H.; Wu, M.-J.; Chen, K.-K. Integrating support vector machine and genetic algorithm to implement dynamic wafer quality prediction system. *Expert Syst. Appl.* **2010**, *37*, 4413–4424. [CrossRef]
12. Wang, G.; Ledwoch, A.; Hasani, R.M.; Grosu, R.; Brintrup, A. A generative neural network model for the quality prediction of work in progress products. *Appl. Soft Comput.* **2019**, *85*, 105683. [CrossRef]
13. Javadpour, R.; Knapp, G.M. A fuzzy neural network approach to machine condition monitoring. *Comput. Ind. Eng.* **2003**, *45*, 323–330. [CrossRef]
14. Liang, R.; Yu, R.; Luo, Y.; Zhang, Y. Machine learning of weld joint penetration from weld pool surface using support vector regression. *J. Manuf. Process.* **2019**, *41*, 23–28. [CrossRef]

15. Valizadeh, M.; Wolff, S.J. Convolutional Neural Network applications in additive manufacturing: A review. *Adv. Ind. Manuf. Eng.* **2022**, *4*, 100072. [CrossRef]
16. Fu, Y.; Downey, A.R.; Yuan, L.; Zhang, T.; Pratt, A.; Balogun, Y. Machine learning algorithms for defect detection in metal laser-based additive manufacturing: A review. *J. Manuf. Process.* **2022**, *75*, 693–710. [CrossRef]
17. Wang, P.; Yang, Y.; Moghaddam, N.S. Process modeling in laser powder bed fusion towards defect detection and quality control via machine learning: The state-of-the-art and research challenges. *J. Manuf. Process.* **2022**, *73*, 961–984. [CrossRef]
18. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
19. Markatos, N.G.; Mousavi, A.; Pippione, G.; Paoletti, R. *Industry 4.0 and Hybrid Feature Selection for Interpreting Quality Loss in the Assembly of Multi-Emitter Laser Modules*; Elsevier: Amsterdam, The Netherlands, 2022.
20. Kiangala, S.K.; Wang, Z. An effective adaptive customization framework for small manufacturing plants using extreme gradient boosting-XGBoost and random forest ensemble learning algorithms in an Industry 4.0 environment. *Mach. Learn. Appl.* **2021**, *4*, 100024. [CrossRef]
21. Benali, L.; Notton, G.; Fouilloy, A.; Voyant, C.; Dizene, R. Solar radiation forecasting using artificial neural network and random forest methods: Application to normal beam, horizontal diffuse and global components. *Renew. Energy* **2019**, *132*, 871–884. [CrossRef]
22. Çolakoğlu, N.; Akkaya, B. Comparison of Multi-Class Classification Algorithms on Early Diagnosis of Heart Diseases Recommendation System for Spotify View Project. 2019. Available online: https://www.researchgate.net/publication/338950098 (accessed on 22 December 2022).
23. NewTechDojo, Learn Random Forest using Excel. 2017. Available online: https://www.newtechdojo.com/learn-random-forest-using-excel/ (accessed on 9 September 2022).
24. Statnikov, A.; Wang, L.; Aliferis, C.F. A comprehensive comparison of random forests and support vector machines for microarray-based cancer classification. *BMC Bioinform.* **2008**, *9*, 319. [CrossRef]
25. Svozil, D.; Kvasnieka, V.; Pospichal, J. Chemometrics and intelligent laboratory systems Introduction to multi-layer feed-forward neural networks. *Chemom. Intell. Lab. Syst.* **1997**, *39*, 43–62.
26. Ciaburro, G.; Venkateswaran, B. *Neural Network with R: Smart Models Using CNN, RNN, Deep Learning, and Artificial Intelligence Principles*; Packt Publishing Ltd.: Birmingham, UK, 2017; Volume 91.
27. Hemmati-Sarapardeh, A.; Amar, M.N.; Soltanian, M.R.; Dai, Z.; Zhang, X. Modeling CO_2 Solubility in Water at High Pressure and Temperature Conditions. *Energy Fuels* **2020**, *34*, 4761–4776. [CrossRef]
28. Temel, F.A.; Yolcu, C.; Kuleyin, A. A multilayer perceptron-based prediction of ammonium adsorption on zeolite from landfill leachate: Batch and column studies. *J. Hazard. Mater.* **2021**, *410*, 124670. [CrossRef]
29. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the KDD'16, 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794. [CrossRef]
30. Feng, Q.; Maier, W.; Stehle, T.; Möhring, H.-C. Optimization of a clamping concept based on machine learning. *Prod. Eng.* **2021**, *16*, 9–22. [CrossRef]
31. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
32. Oukawa, G.Y.; Krecl, P.; Targino, A.C. Fine-scale modeling of the urban heat island: A comparison of multiple linear regression and random forest approaches. *Sci. Total. Environ.* **2022**, *815*, 152836. [CrossRef]
33. Fouedjio, F. Exact Conditioning of Regression Random Forest for Spatial Prediction. *Artif. Intell. Geosci.* **2020**, *1*, 11–23. [CrossRef]
34. Segal, M.R. *Machine Learning Benchmarks and Random Forest Regression*; Center for Bioinformatics and Molecular Biostatistics, University of California: San Francisco, CA, USA, 2004.
35. Geng, H.; Wang, Z.; Zou, L.; Mousavi, A.; Cheng, Y. Protocol-Based Tobit Kalman Filter Under Integral Measurements and Probabilistic Sensor Failures. *IEEE Trans. Signal Process.* **2020**, *69*, 546–559. [CrossRef]
36. Goodfellow, I.J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Nets. *Commun. ACM* **2020**, *63*, 139–144. [CrossRef]
37. Radford, A.; Metz, L.; Chintala, S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. In Proceedings of the 4th International Conference on Learning Representations, ICLR 2016—Conference Track Proceedings, San Juan, Puerto Rico, 2–4 May 2016.
38. Ebayyeh, A.A.R.M.A.; Danishvar, S.; Mousavi, A. An Improved Capsule Network (WaferCaps) for Wafer Bin Map Classification Based on DCGAN Data Upsampling. *IEEE Trans. Semicond. Manuf.* **2021**, *35*, 50–59. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.