


Article

Application of Machine Learning to Solid Particle Erosion of APS-TBC and EB-PVD TBC at Elevated Temperatures

Yuan Liu ¹, Ravi Ravichandran ¹, Kuiying Chen ^{2,*} and Prakash Patnaik ² ¹ Tecsisis Corporation, Ottawa, ON K2E 7L5, Canada; yliu@tecsisis.ca (Y.L.); ravi@tecsisis.ca (R.R.)² Aerospace Research Center, Structures, Materials and Performance Laboratory, National Research Council Canada, Ottawa, ON K1A 0R6, Canada; prakash.patnaik@gmail.com

* Correspondence: kuiying_chen@hotmail.ca; Tel.: +1-613-993-1247; Fax: +1-613-949-8165

Abstract: Machine learning (ML) and deep learning (DL) for big data (BD) management are currently viable approaches that can significantly help in high-temperature materials design and development. ML-DL can accumulate knowledge by learning from existing data generated through multi-physics modelling (MPM) and experimental tests (ETs). DL mainly involves analyzing nonlinear correlations and high-dimensional datasets implemented through specifically designed numerical algorithms. DL also makes it possible to learn from new data and modify predictive models over time, identifying anomalies, signatures, and trends in machine performance, develop an understanding of patterns of behaviour, and estimate efficiencies in a machine. Machine learning was implemented to investigate the solid particle erosion of both APS (air plasma spray) and EB-PVD (electron beam physical vapour deposition) TBCs of hot section components. Several ML models and algorithms were used such as neural networks (NNs), gradient boosting regression (GBR), decision tree regression (DTR), and random forest regression (RFR). It was found that the test data are strongly associated with five key factors as identifiers. Following test data collection, the dataset is subjected to sorting, filtering, extracting, and exploratory analysis. The training and testing, and prediction results are analysed. The results suggest that neural networks using the BR model and GBR have better prediction capability.

Keywords: machine learning; solid particle erosion; APS-TBC; EB-PVD TBC; elevated temperatures



Citation: Liu, Y.; Ravichandran, R.; Chen, K.; Patnaik, P. Application of Machine Learning to Solid Particle Erosion of APS-TBC and EB-PVD TBC at Elevated Temperatures. *Coatings* **2021**, *11*, 845. <https://doi.org/10.3390/coatings11070845>

Academic Editor: Changheui Jang

Received: 11 May 2021

Accepted: 1 July 2021

Published: 13 July 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 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

Thermal barrier coatings (TBCs) are applied on the surface of hot hardware parts of gas turbine engines to increase the turbine efficiency by providing thermal insulation and protection from the harsh environment. The industry generally uses TBCs processed by two technologies: namely, electron beam physical vapour deposition (EB-PVD) or air plasma spray (APS). The EB-PVD is widely used to deposit ceramic coatings on combustor cans, ductwork, platforms, and other hot gas path components. The EB-PVD, due to its unique columnar structure, offers satisfactory levels of resistance to damaging and failure for aerofoil applications. TBCs exhibit two primary modes of failures, namely by oxidation (TGO, i.e., thermally grown oxide growth at the bond coat–top coat interface) and by erosion (impact by projectiles ingested into the gas stream) [1]. For erosion, turbine blades have a tip velocity in the order of 300 m/s and suffer rapid erosive wear upon impact by small hard particles (1–30 µm) entrained within the combustion gases of the turbine. In turbomachinery, particle impacts are known to increase tip clearances and blade surface roughness and produce changes in the blade profiles (leading and trailing edges) [2].

Erosion refers to the progressive loss of material from the surface of TBCs while maintaining the integrity of the components. Damage occurs under small particle impacts where the near-surface region (5 to top 20 µm) of the individual columns are cracked, as it is very difficult to remove all solid particles from the gas stream without taxing the performance of gas turbine engines [3,4]. Cracks generally initiate at the elastic/plastic

interface under the impacting particles [5]. Foreign object damage (FOD), on the other hand, involves gross plastic damage, the bending of columns, and ultimately the propagation of shear bands down to the ceramic bond coat interface. TBC erosion is detrimental to the thermal protection of rotating hardware, e.g., turbine blades [2]. TBC damage could be very significant and life-limiting for highly loaded and rotating turbine components when the engine operates under erosive environments such as flying over a desert area or near an active volcanic or offshore ocean environment.

The top ceramic coating is typically made of yttria-stabilised zirconia (YSZ) for low thermal conductivity and high thermal-expansion coefficient. The bond coat is typically made of MCrAlY (M: Ni, Co and Ni + Co) for its good oxidation resistance. It is well recognised that the erosion rate (ER) of TBCs is affected by many factors that could be broadly classified into three types: impingement, particle, and materials variables. The impingement factors mainly consist of particle velocity, particle concentration, and the angle of incidence. The particle variables include particle shape, size, and hardness. Materials variables involve target material properties, such as hardness, microstructure, and porosity. The higher the TBC micro-hardness, the lower the erosion rate. The room temperature erosion rate of an EB-PVD thermal barrier ceramic using $100\ \mu\text{m}\ \text{Al}_2\text{O}_3$ was observed to be an order of magnitude lower than the ER for APS [6]. Experimental studies show that the erosion rates increase with an increased impingement angle, impact velocity, and temperature. The laser-glazing process increases the micro-hardness from about 550 HV for the as-sprayed layer to about 1550 HV for the as-glazed layer [7]. The impingement angle increased the ER for both plasma-sprayed and laser-glazed TBCs. Laser glazing enhanced the erosion resistance (lower ER) of plasma-sprayed TBCs by about 1.5 to 3 times with the impingement angle ranging between 30° and 75° , while the erosion resistance did not significantly improve when the impingement angle reached 90° . Hence, the higher fracture toughness favors superior erosion resistance [3]. Many methods were proposed to improve the erosion resistance of TBCs, such as laser glazing and re-melting of ceramic coating and doping of rare earth material such as gadolinia [8].

Early research focused on the effect of variables such as velocity, impact angle, and erodent properties on ER, while recent work has examined erosion mechanisms [5] and the effects of ageing and dopant additions. Microstructures and ageing conditions are known to greatly affect the erosion behaviour of TBCs. The ability of the columnar boundaries in the EB-PVD ceramic microstructure (and the vertical micro-cracks in the segmented APS TBC) limits crack propagation and accounts for its improved erosion resistance over the conventional APS microstructure. The sintering of columns occurs in EB-PVD TBCs, giving rise to larger 'columns', resulting in greater material removal per impact as cracks are initiated in the coating. Sintered column boundaries no longer inhibit crack growth, and cracks propagate further across the near-surface region of the TBC. In contrast, the removal of material for plasma-sprayed coatings occurs through poorly bonded splat boundaries, and hence, larger volumes of material are easily lost, resulting in higher erosion rates. The ability of the columnar boundaries in the EB-PVD ceramic microstructure (and the vertical micro-cracks in the segmented APS TBC) to limit crack propagation is thought to account for its improved erosion resistance over the conventional APS microstructure.

Recent observations indicate that the erosion resistance of many new lower thermal conductivity TBCs is worse than the commercial 7YSZ materials [8], and that during service, the erosion resistance of EB-PVD TBC-coated parts can degrade by up to a factor of $\times 4$ due to the ceramic sintering process [8].

For life-critical applications, erosion of the ceramic topcoat is perceived as a potential problem, whether for aero- or industrial applications. TBCs are more susceptible to erosion than fully dense ceramics because the coating microstructures contain many crack-like features. Erosion rates are highly dependent on a number of material properties, which include Young's modulus, hardness, and fracture toughness, all of which are temperature dependent. In contrast to TBC oxidation failure mechanisms, the erosion problem has received far less attention with regard to experimental studies, analytical and computation

modelling, prediction, and life analysis. With the advent of artificial intelligence technology, this problem needs to be researched for more in-depth understanding and erosion damage prediction for conventional and new advanced TBCs. A prediction model for erosion rate as a function of key variables will be a designing asset for reliable lifetime analysis.

Loss of material from gas turbines hardware surfaces by a solid-particle erosion mechanism is considered as one of the principal causes for limiting the durability of TBCs. However, no serious efforts towards the deeper understanding of the erosion behaviour and towards the development of any prediction modelling have been attempted. The research project in collaboration with NRC, Ottawa has been undertaken on this topic in order to realise the following major objectives:

1. Identifying key variables having dominant influences on the erosion loss during the gas turbine operation at high temperature and in the presence of corrosive and hazardous environments. For instance, such factors include solid erosive particle size, velocity, as well as angle of incidence on the TBC surface.
2. Compiling, classifying, and processing erosion damage data for TBCs through extensive survey of published literatures. The dataset collected will consist of both mechanical material loss in terms of weight or thickness, and secondly, microstructural degradation and changes due to erosion.
3. Establishing reliable prediction modelling and methodology using the erosion rate data and the significant variables identified. Artificial intelligence (AI) technology comprising of machining learning (ML) and deep learning (DL) has emerged as the most powerful prediction tool in recent times. Various AI-based models will be identified and used to determine the most valid and suitable ones for the prediction of the erosion rate.

In order to realise the objectives, the work needs to focus on the following:

1. Coating—6%–8% conventional YSZ TBC processed by electron beam physical vapour deposition (EB-PVD) or air plasma spray (APS).
2. Literature survey—compilation and classification of relevant data and information on the hard particle erosion process, behaviour, mechanisms, and rate.
3. Exploratory data analysis.
4. Principal component analysis for feature extraction.
5. Correlation coefficient analysis for model performance evaluation.
6. Neural network modelling approach and analysis.
7. Gradient boosting regression approach.
8. Decision tree regression approach.
9. Random forest regression approach and analysis.
10. Microstructural image collections for various erosive conditions and surfaces.
11. DL-based modelling approach and analysis for structural features e.g., porosity content, size, crack size, grain size, hardness, etc.

2. Literature Review on Current Research Status of Erosion Rate of TBCs and Data Collecting

2.1. Literature Review

The main focus of the first task in this project is to review existing literature on the current status of erosion rate of TBCs, and the outcomes of this review are outlined in Table 1.

Table 1. Summarised experimental research studies on erosion rates of TBCs.

Ref. No.	Year	Author	Material	Research Topic
[9]	1998	Davis	6.6%–20% Y ₂ O ₃ -ZrO ₂ and of four and eight layers of Al ₂ O ₃ -ZrO ₂	Erosion rate vs. TBC Thickness, TBC Surface roughness, Impact particle size, Particle velocity, Impingement angle, Test temperature (Room temp)
[10]	1998	Bruce	7% Y ₂ O ₃ -ZrO ₂	Erosion rate vs. TBC Thickness, Impact particle size, Particle velocity, Impingement angle, Test temperature; Equivalent equations

Table 1. Cont.

Ref. No.	Year	Author	Material	Research Topic
[11]	1998	Nicholls	APS and EB-PVD 7% Y ₂ O ₃ -ZrO ₂	Erosion rate vs. TBC Thickness, TBC Surface roughness, Impact particle size, Particle velocity, Impingement angle, Test temperature, and Erodent types
[12]	1999	Eaton	APS 7.5% Y ₂ O ₃ -ZrO ₂	Erosion rate vs. TBC Thickness, Impact particle size, Particle velocity, Impingement angle, Test temperature Function of erosion rate as velocity
[5]	1999	Janos	APS 7.5% Y ₂ O ₃ -ZrO ₂	Erosion rate vs. TBC Thickness, Hardness, Impact particle size, Particle velocity, Impingement angle, Test temperature, Erodent types, Porosity, Ageing temp and time. Function of erosion rate as micro-hardness
[13]	1999	Nicholls	EB-PVD 8% Y ₂ O ₃ -ZrO ₂	Erosion rate vs. Impact particle size, Particle velocity, Impingement angle, Test temperature, Erodent types
[14]	2003	Nicholls	EB-PVD 8% Y ₂ O ₃ -ZrO ₂	Erosion rate vs. Impact particle size, Particle velocity, Impingement angle, Test temperature, Erodent types
[15]	2009	Wellman	APS and EB-PVD 7% Y ₂ O ₃ -ZrO ₂	Erosion rate vs. Impact particle size, Particle velocity, Impingement angle, Test temperature, Erodent types
[16]	2011	Cernuschi	APS and EB-PVD 7.5~8% Y ₂ O ₃ -ZrO ₂	Erosion rate vs. TBC Thickness, hardness, Impact particle size, Particle velocity, Impingement angle, Test temperature, Erodent types, Porosity, Ageing temp and time.; Function of erosion rate as micro-hardness
[17]	2018	Shin and Hamed	APS 7% Y ₂ O ₃ -ZrO ₂	Erosion rate vs. TBC Thickness, hardness, Impact particle size, Particle velocity, Impingement angle, Test temperature, Erodent types, Porosity, Ageing temp and time.; Function of erosion rate as micro-hardness

2.2. Data Collection

2.2.1. Basic Information Gathering

Accurate data collection and accurate information classification are the key to maintain the integrity of the research. The first step of data collecting was reviewing the papers and gathering basic information about their research, including the material information, manufacturing methods, measurement of erosion rate with variation, and different test conditions, including temperature, or other parameters.

Taking the work of Shin [17] as an example, the procedure used for this research is explained below. This paper presents the results of an experimental investigation of the effect of a topcoat microstructure on the erosion resistance of air plasma-sprayed 7 wt.% YSZ thermal barrier coating (TBC) at high temperature. Two sets of air plasma-sprayed YSZ TBCs of differing microstructures (porosities of 12.9% ± 0.5% and 19.5% ± 1.2%) were tested in a high-temperature erosion tunnel to simulate a modern gas turbine engine's operating conditions. The tests were conducted over a range of temperatures between 537 and 980 °C, gas velocities between 122 and 305 m/s, and impingement angles between 20° and 90°. The two types of erodent powders used in the erosion tests were nominal 27 µm white-fused alumina and nominal 15 µm A3 test dust. The results demonstrated reduced erosion resistance associated with increased porosity in the topcoat layer.

Erosion tests were conducted in a high-temperature erosion test facility at the University of Cincinnati. The particle feed rates are controlled by a volumetric single screw-type particle feeder with a high-pressure air supply and kept constant during each test. After the erosive particles are introduced in the tunnel, they are dispersed to achieve a uniform distribution by the time they reach the test section. Particle velocities were experimentally measured for the nominal 27 µm alumina particles at room temperature. It was determined that the particle velocities reach 96% of the flow velocity of 91 m/s at the test section. Further analysis was conducted to estimate particle velocities under high-temperature gas

flow test conditions. The analysis was based on the particles being accelerated by drag force associated with velocity differences between particles and gas flow. The tests were conducted at gas temperatures ranging between 537 and 980 °C, gas velocities of 122, 231, and 305 m/s, and a 20° impingement angle.

2.2.2. Data Extracting

The second step was extracting data from the plots. For most of the published research papers, the data are provided as plots of erosion rate (ER) with variation of the impact velocity, impact angle, measurement temperature, or other parameters. In our present work, the data extracting was performed using PlotDigitizer, which is an open-source Java program that allows the user to digitise data points off of scanned plots, scaled drawings, or orthographic photographs. The source codes of the tool are available at <http://plotdigitizer.sourceforge.net/> since 2001.

Information of the authors are listed as follows: Joseph A. Huwaldt (jhuwaldt@users.sourceforge.net) and Scott Steinhorst (scottsteinhorst@users.sourceforge.net). The process of data extracting using PlotDigitizer includes four steps:

Step 1: Importing plot

Step 2: Calibrating x - and y -axis

Step 3: Digitising data points set

Step 4: Exporting dataset

All plots related to the erosion rates in the reference papers listed in Table 1 are digitised using the PlotDigitizer tool. Then, all the data points are put in a big table to be sorted based on all possible measurements information. The total number of the data points is 247. For all these 247 data points extracted, the values of ER and impact velocity and measured temperature are recorded. A lot of effort has been made to obtain important parameters related to the ER of TBC, such as thickness of TBC, hardness of TBC, erodent type, and all those available information is stored in a large table.

2.2.3. Dataset Used in the Present Study

It must be noted here that not all parameters (TBC thickness of TBC, hardness, total test time, erodent information) are available for all of the 17 reference papers. In order to prepare the datasets to be used for the machine learning test, one dataset was collated based on the big table data and is summarised and explained in Tables 2 and 3. This dataset contains six variables (wt.% of Y₂O₃, impact angle, impact velocity, particle size, measurement temperature, and erosion rate) and the total number of samples is 245. In order to study the different patterns of erosion rates of APS and EB-PVD TBC, the samples are separated into two different groups to perform prediction separately.

Table 2. Summarised data of EB-PVD YSZ TBC.

Ref. No.	Year	Author	APS Sample No.	EB-PBD Sample No.	Total
[9]	1998	Davis	12	1	13
[10]	1998	Bruce	0	45	45
[11]	1998	Nicholls	14	14	28
[12]	1999	Eaton	28	0	28
[5]	1999	Janos	12	0	12
[13]	1999	Nicholls	10	0	10
[14]	1999	Nicholls	16	14	30
[15]	2009	Wellman	1	20	21
[16]	2011	Cernuschi	5	9	14
[17]	2017	Shi	44	0	44
total	-	-	142	103	245

Table 3. Summarised experimental research on erosion rates of TBCs.

Variables	Unit	Description	Function
Material	W/(m·K)	wt.% of Y ₂ O ₃	
Impact Angle	Degree	Impact angle of particles	
Impact Velocity	m/s	Impact velocity of particles	Inputs
Particle Size	micron	Diameter of erodent particles	
TestTemp	°C	Temperature during measurement	
Target ER	g/kg	Erosion rate of TBC	Output

The source data fed into the machine learning model are called input, with the goal of making a decision or prediction about the data. The data are broken down into binary signals to allow it to be processed by single neurons—for example, an image is input as individual pixels. In the present work, five variables are adopted as inputs. The outputs (or called target) of the machine learning framework can be a discrete value (for example, a category ID). In the present work, ER is the subject investigated and the output. In summary, the following set of input and output variables was prepared:

- Input variables for prediction of erosion rates:
 - wt.% of Y₂O₃
 - Impact angle
 - Impact velocity
 - Particle size
 - Measurement temperature
- Output: erosion rate

3. Prediction of Erosion Rates of TBCs Using Machine Learning

3.1. Exploratory Data Analysis

Before machine learning modelling was used for predictions, exploratory data analysis (EDA) was used. In statistics and scientific research, EDA is a way to analyse the collected dataset to summarise their main characteristics and to discover patterns to foster hypothesis development and refinement [18]. In addition, EDA is useful to uncover the underlying structure, to extract important variables, to detect outliers and anomalies, and so on.

The major objective in many applications of PCA is to replace the m elements of the original data by a much smaller number p of principal components (PCs) while discarding very little information. In the present work, the number of inputs is not large (five). The main purpose of using PCA here is to get statistic information of all variables and get an idea of how many original input variables are needed to retain much of the variation within the original data.

For the PCA analysis in this study, the five inputs along with the output of ER are included. The resulting matrix of correlation coefficients are found in Tables 4–6. It is observed from Table 4 that for all data points together, the input variables impact velocity, particle size, and impact angle have enough correlations with the output variable ER such that they can be identified as relevant inputs. For APS TBC data, besides impact velocity, particle size, and impact angle, the test temperature also shows enough correlation with the output variable ER such that the four can be identified as relevant inputs. As for the EB-PVD data, test temperature shows the largest correlation with the ER, and the impact velocity and particle size also show enough correlation with the output.

Table 4. Correlation coefficients for all TBC data.

Parameters	Material	Impact Angle	Impact V	Particle Size	Test Temp	Erosion Rate
Material	1	−0.03194	−0.09196	−0.04592	−0.26285	−0.02612
Impact Angle	−0.03194	1	−0.42818	0.356322	−0.04395	−0.13179
Impact V	−0.09196	−0.42818	1	−0.48263	0.374805	0.54068
Particle Size	−0.04592	0.356322	−0.48263	1	0.186174	−0.24137
Test Temp	−0.26285	−0.04395	0.374805	0.186174	1	0.049198
Erosion Rate	−0.02612	−0.13179	0.54068	−0.24137	0.049198	1

Table 5. Correlation coefficients for APS TBC data.

Title	Material	Impact Angle	Impact V	Particle Size	Test Temp	Erosion Rate
Material	1	0.142241	−0.18374	0.332938	−0.24079	−0.08089
Impact Angle	0.142241	1	−0.32357	0.310126	−0.2549	0.229697
Impact V	−0.18374	−0.32357	1	−0.47894	0.640075	0.495648
Particle Size	0.332938	0.310126	−0.47894	1	−0.30445	−0.21112
Test Temp	−0.24079	−0.2549	0.640075	−0.30445	1	0.145308
Erosion Rate	−0.08089	0.229697	0.495648	−0.21112	0.145308	1

Table 6. Correlation coefficients for EB-PVD TBC data.

Parameters	Material	Impact Angle	Impact V	Particle Size	Test Temp	Erosion Rate
Material	1	−0.30787	0.034516	−0.15291	−0.30402	0.061988
Impact Angle	−0.30787	1	−0.12055	0.205277	0.257427	0.004347
Impact V	0.034516	−0.12055	1	−0.70291	0.013128	0.242658
Particle Size	−0.15291	0.205277	−0.70291	1	0.358358	−0.30279
Test Temp	−0.30402	0.257427	0.013128	0.358358	1	−0.59386
Erosion Rate	0.061988	0.004347	0.242658	−0.30279	−0.59386	1

3.2. Preliminary Quick Test on Prediction of Erosion Rate Using Different ML Models

There are usually several suitable candidate models for a particular type of problem. In order to select the suitable learning methods from a set of eligible machine learning methods, a preliminary quick test on the prediction of ER of TBC was performed using the Tecsis Model Selection Tool (MST).

The Model Selection Tool (MST) is built using Python (Version 3.7, in Spyder/Anocoda Navigator), and the main Graphical User Interface (GUI) can be seen in Figure 1. The following are the five major modules on the interface:

- i. Data: Load data file into the tool (.csv file)
- ii. Input and Output: Choose inputs and output; add extended terms; plot data
- iii. Pre-process: Normalise inputs
- iv. Data Analysis: Predict target (output) by inputs using different models
- v. Results Ranking: Rank results using different methods (single metric, aggregated, MCDM) and provide the best model.

In the Data Analysis module, ten regression models (algorithms) are imbedded into the model library. The list of the models is shown in Figure 1. As a preliminary work, the ER data (including all TBC, APS, and EB-PVD, summarised and explained in Tables 2 and 3) are input into the MST to perform the quick test. The results of the quick test are listed in Tables 7–9, for all, APS, and EB-PVD TBC data, respectively. The gradient boosting regression (GB), random forest regression (RF), decision tree regression (DT), and AdaBoost classifier (AC) approaches provide relatively good prediction. The neural network (NN) approach is a promising model, which is mainly used in many applications. A case study is

presented in the following section to highlight the ML approach to determine and predict the erosion rates of TBC using the following methodology:

1. Neural network
2. Gradient boosting regression
3. Decision tree regression
4. Random forest regression

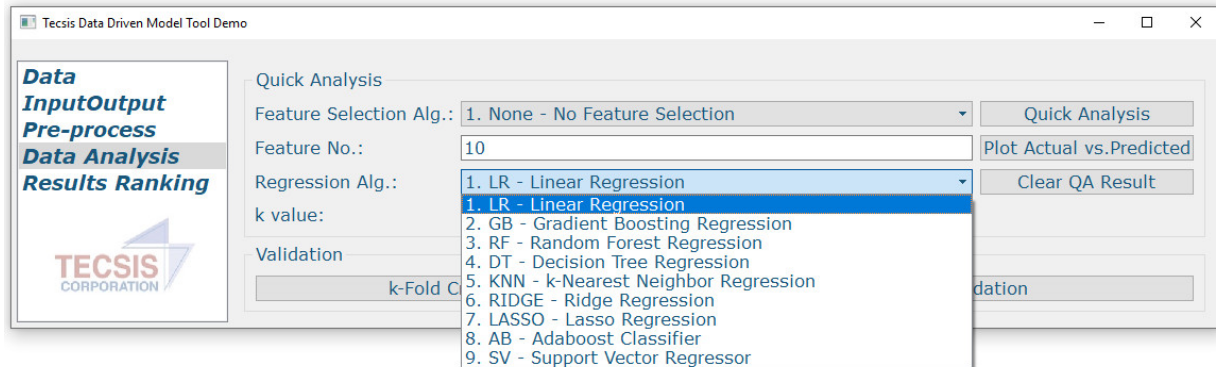


Figure 1. Tectis Model Selection Tool (MST) main window.

Table 7. Quick test results for all TBC data.

Features	RAlg	MSE	MAE	R2	AdjR2	PRESS	BIC	GCV
5	LR	5599.136	51.0778	0.3463	0.3327	52.9465	2164.757	1,452,706
5	GB	4143.612	41.9211	0.5162	0.5062	43.4084	2090.399	1,075,068
5	RF	3403.455	31.0987	0.6026	0.5944	32.1065	2041.795	883,032.7
5	DT	3184.008	30.2029	0.6282	0.6205	31.187	2025.333	826,096.7
5	KNN	4591.058	40.6564	0.464	0.4528	42.1395	2115.727	1,191,158
5	RIDGE	6333.306	55.1329	0.2605	0.2452	57.0639	2195.19	1,643,188
5	LASSO	7581.492	63.9384	0.1148	0.0964	66.0237	2239.622	1,967,032
5	AB	4197.602	41.752	0.5099	0.4997	43.1949	2093.597	1,089,075
5	SV	7451.062	42.7329	0.13	0.112	44.313	2235.336	1,933,192

Table 8. Quick test results for APS TBC data.

Features	RAlg	MSE	MAE	R2	AdjR2	PRESS	BIC	GCV
5	LR	6315.652	60.1596	0.4588	0.439	62.7441	1281.14	983,977.6
5	GB	3552.395	45.6964	0.6956	0.6845	47.4223	1198.856	553,462.5
5	RF	1454.941	24.0177	0.8753	0.8708	24.9086	1071.206	226,679.5
5	DT	1115.415	17.1148	0.9044	0.9009	17.7995	1033.206	173,781.5
5	KNN	5812.361	55.3393	0.5019	0.4837	57.6708	1269.264	905,564.9
5	RIDGE	7960.008	68.9798	0.3178	0.293	71.7985	1314.23	1,240,168
5	LASSO	9680.04	78.8275	0.1704	0.1402	82.1582	1342.206	1,508,149
5	AB	3142.537	46.3897	0.7307	0.7209	48.2247	1181.325	489,606.7
5	SV	7627.288	58.6225	0.3464	0.3225	60.8968	1308.124	1,188,330

Table 9. Quick test results for EB-PBD TBC data.

Features	RAIlg	MSE	MAE	R2	AdjR2	PRESS	BIC	GCV
5	LR	84.191	6.5402	0.4797	0.4526	7.5648	479.9248	9694.465
5	GB	58.5397	5.0387	0.6382	0.6194	5.6239	442.8598	6740.761
5	RF	42.2348	3.4883	0.739	0.7254	3.6952	409.5608	4863.272
5	DT	34.6876	2.4219	0.7856	0.7745	2.5633	389.4808	3994.224
5	KNN	70.8368	5.2107	0.5622	0.5394	5.5297	462.3084	8156.746
5	RIDGE	105.8371	7.8759	0.3459	0.3119	9.432	503.2637	12,186.97
5	LASSO	161.8108	10.4959	0	−0.0521	12.5985	546.5655	18,632.27
5	AB	55.6216	4.5628	0.6563	0.6384	4.8273	437.6442	6404.744
5	SV	68.6382	4.0034	0.5758	0.5537	4.2498	459.0924	7903.579

The following performance measures are adopted in the present study to evaluate prediction models using different datasets:

- Coefficient of Determination (R^2)

In statistics, the coefficient of determination (R^2) is a measure of how well the regression predictions approximate the real data points. An R^2 of 1 indicates that the regression predictions perfectly fit the data.

The most general definition of the coefficient of determination is:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \quad (1)$$

$$SS_{res} = \sum_{i=1}^n (y_i - f_i)^2 \quad (2)$$

$$SS_{tot} = \sum_{i=1}^n (y_i - \bar{y})^2 \quad (3)$$

where n represents the total number of observations, y_i are the real data points, and f_i are the predicted values. The values close to 1 for the coefficient of determination indicate a good deal of predictive power of selected inputs for the output variable, while values close to 0 indicate little predictive power.

- Mean Squared Error (MSE)

In statistics, Mean Squared Error (MSE) is a common measure of the difference between the predictor (value predicted by the model), or an estimator and the observed value. MSE is a measure of the quality of an estimator, and the values closer to zero are better. Usually, MSE is good to use if we have a lot of outliers in the data. MSE is given by:

$$MSE = \frac{\sum_{i=1}^n (f_i - y_i)^2}{n} \quad (4)$$

- Maximum Absolute Error (MAXE)

Absolute error is defined as the absolute value of the difference between the measured value and the measured true value, and it has the same as the unit of measurement. The maximum absolute error usually can be considered as the maximum possible error given a measuring tool's degree of accuracy:

$$MAXE = \max(|f_i - y_i|). \quad (5)$$

3.3. Neural Network Model

3.3.1. Neural Network Description

Artificial neural networks are computational models, which work similar to the functioning of a human nervous system. There are several kinds of artificial neural networks. These types of networks are implemented based on the mathematical operations and a set of parameters required to determine the output.

The neural network model was comprised of three parts: one input layer, one (sometimes two or three) hidden layer (layers), and an output layer, as illustrated in Figure 2. Multiple-layer networks are quite powerful. For instance, a network of two layers, where the first layer is sigmoid and the second layer is linear, can be trained to approximate any function (with a finite number of discontinuities) arbitrarily well. This kind of two-layer network is used extensively in backpropagation algorithms [19].

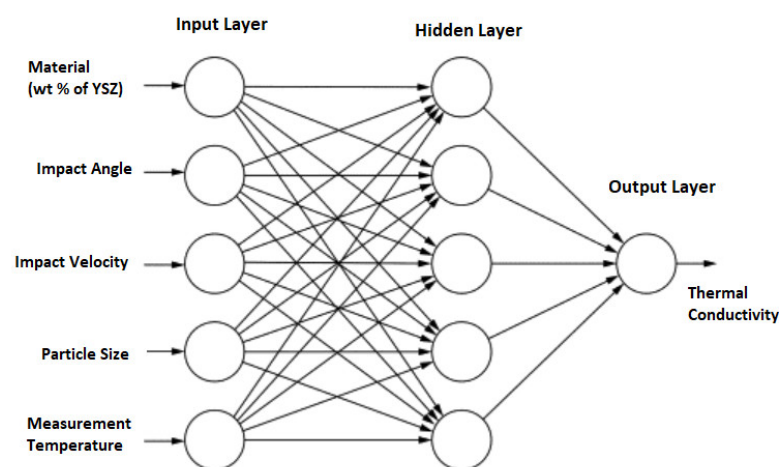


Figure 2. Structure of neural network in the present work.

In the present study, the input layer has five neurons, which conform to five variables extracted from previous BR measurement research: material, impact angle, impact velocity, particle size, and test temp, as shown in Figure 2. The output layer contains one neuron representing the target of the prediction: erosion rate. For the hidden layer, we tried single-layer and two-layer settings with different numbers of neurons. The number of layers and number of nodes in each layer were used to determine the topology of the ANN model. More details will be provided in the “Training Parameters Setting” section.

3.3.2. Training Algorithms

In function optimisation problems with neural networks, the training algorithm is designed to determine the best network parameters in order to minimise network error. Various function optimisation methods can be applied to neural network model training. Considering a multi-layer feedforward network, $p_i = (p_{i1}, p_{i2}, \dots, p_{ii})$ contained values for I input (independent) variables from individual i . The input variables are associated with each of N neurons in a hidden layer by using weights (w_{kj} , $k = 1, 2, \dots, N$), which are specific to each independent variable (j) to neuron (k) connection. The mapping has two forms for the relationship between output \hat{t} and independent input variables [20]:

$$\text{Hidden Layer } n_k^{(1)} = \sum_{j=1}^R \omega_k^{(1)} p_j + b_k^{(1)}; a_k^1 = f_{level-one}(n_k^{(1)}) \quad (6)$$

$$\text{Output Layer } n_k^{(2)} = \sum_{j=1}^R \omega_k^{(2,1)} a_k^1 + b_1^{(2)}; \hat{t}_i = a_k^{(2)} = f_{level-two}(n_k^{(2)}) \quad (7)$$

For an M layer network with N neurons, the biases are $b_k^{(1)}$ ($k = 1 \sim N$). After the end of the training process, this activated quantity is carried out again with function g , which can be expressed as:

$$g \left[\sum_{j=1}^N \omega'_k f_k(\cdot) + b_k^{(2)} \right] = f_k(n_k^{(2)}) \quad (8)$$

Then, the estimated target variable value of t_i in the training dataset can be expressed as:

$$\hat{t}_i = g \left\{ \sum_{j=1}^N \omega'_k f \left(\sum_{j=1}^R \omega_{kj} p_j + b_k^{(1)} \right) + b_k^{(2)} \right\} \quad (9)$$

in which, $j = 1, 2, \dots, R$ and $k = 1, 2, \dots, N$.

The task of the network is to use an algorithm to learn the connection between the specified set of input–output pairs of a dataset. In machine learning, specifically deep learning, backpropagation is a widely used algorithm in training feedforward neural networks for supervised learning. Backpropagation is shorthand for “the backward propagation of errors”, since an error is computed at the output and distributed backwards throughout the network’s layers.

In the training process, a common performance function is used for computing the distance between real and predicted data. This function can be expressed as follows [16,17]:

$$F = E_D(D|\omega, M) = \frac{1}{N} \sum_{i=1}^n (\hat{t}_i - t_i)^2 \quad (10)$$

in which E_D is the mean sum of squares of the network error, D is the training set with input–target pairs, and M is the number of neural networks.

Backpropagation was created by generalising the Widrow–Hoff learning rule to multiple-layer networks and non-linear differentiable transfer functions [18]. Input vectors and the corresponding target vectors are used to train a network until it can approximate a function, associate input vectors with specific output vectors, or classify input vectors in an appropriate way as defined. Backpropagation is closely associated with the Gauss–Newton algorithm and is part of continuing research in neural backpropagation. Backpropagation is a special case of a more general technique called automatic differentiation.

Properly trained backpropagation networks tend to give reasonable answers when presented with inputs that they have never seen. Typically, a new input leads to an output similar to the correct output for input vectors used in training that are similar to the new input being presented. This generalisation property makes it possible to train a network on a representative set of input/target pairs and get good results without training the network on all possible input/output pairs [21].

There are many variations of the backpropagation algorithm. In the present study, Bayesian regularisation was adopted, which is a backpropagation algorithm. In this training process, an adaptive parameter called mu (Marquardt adjustment parameter) [22] was used with a default value of 0.005, and this parameter is adapted (by decreasing and increasing its value) using two other parameters: mu decrease ratio = 0.1 and mu increase ratio = 10. We used the tanh activation function (implemented using the ‘tansig’ function in MATLAB) in the hidden layers and the linear activation function in the output layer. This choice is motivated by the fact that using the tanh activation provides a greater activation output range (compared to the case of using the sigmoid activation function), which leads to larger derivatives, thus helping with faster convergence to the minimum cost function. More details of Bayesian regularisation can be found in the previous Tecsis-NRC report (2020).

In the present study, no validation subset is needed for the BR training algorithm. Thus, the training set makes up approximately 70% of the full dataset, testing making up approximately 30% of the dataset.

3.3.3. Neural Network Results

The performance of ER prediction using a single-layer NN model with a BR algorithm can be found in Tables 10–12. For the APS + EB-PVD TBC data case, the R^2 values are around 0.49, which is deficient. The best prediction performance is obtained for APS TBC data cases, which provide overall good prediction performance, whose values of R^2 are all around 0.84. The EB-PVD TBC case also obtains relatively good results like in Figures 3–5.

Table 10. Performances of ER prediction using the NN approach for APS + EB-PVD TBC data.

Hyper-Parameters		Train			Test			All		
Number of Nodes	Repeat	R^2	MSE	MAXE	R^2	MSE	MAXE	R^2	MSE	MAXE
10	100	0.4740	4570.4	327.79	0.3329	6031	261.04	0.4463	4791	348.23
20	100	0.4636	4649.6	334.95	0.3240	6786.9	269.54	0.4306	4972.3	357.01
30	100	0.5627	3735.4	290.14	0.2789	9322	322.22	0.4847	4579.1	372.07
40	100	0.5310	4013	306.53	0.3377	6865.9	289.63	0.4871	4443.8	355.04
50	100	0.5436	3891.7	298.03	0.3211	8374.8	317.36	0.4838	4568.7	373.41
60	100	0.5371	3988.7	300.58	0.3021	7824.4	298.21	0.4767	4567.9	364.35

Table 11. Performances of ER prediction using the NN approach for APS TBC data.

Hyper-Parameters		Train			Test			All		
Number of Nodes	Repeat	R^2	MSE	MAXE	R^2	MSE	MAXE	R^2	MSE	MAXE
5	100	0.8340	1958.3	159.21	0.6473	4388	178.58	0.8034	2317.6	192.14
10	100	0.8540	1732.2	155.81	0.6524	4196.8	181.08	0.8219	2096.7	190.42
15	100	0.8419	1851.3	157.04	0.6554	4394	180.7	0.8108	2227.4	190.86
20	100	0.8687	1535.6	149.59	0.6991	3946.7	170.84	0.8400	1892.2	184.01
25	100	0.8510	1750.6	158.41	0.6701	4022.3	171.5	0.8227	2086.5	187.81
30	100	0.8428	1849.8	158.17	0.6625	4391.2	178.8	0.8116	2225.6	194.69
35	100	0.8549	1702.6	152.25	0.6816	4211.6	172.59	0.8245	2073.6	188.45
40	100	0.8459	1843.8	157.39	0.6146	4737.9	191.73	0.8078	2271.8	204.34

Table 12. Performances of ER prediction using the NN approach for EB-PVD TBC data.

Hyper-Parameters		Train			Test			All		
Number of Nodes	Repeat	R^2	MSE	MAXE	R^2	MSE	MAXE	R^2	MSE	MAXE
5	100	0.7106	46.684	27.339	0.6039	83.216	22.887	0.6802	52.004	28.305
10	100	0.7147	46.416	27.294	0.5868	81.478	22.99	0.6827	51.522	28.253
15	100	0.7118	46.457	27.529	0.5893	86.439	23.182	0.6786	52.28	28.627
20	100	0.7027	47.918	27.771	0.5909	79.554	22.685	0.6765	52.525	28.602
25	100	0.7025	48.234	27.534	0.6146	75.217	21.724	0.6782	52.164	28.384
30	100	0.6983	48.857	27.83	0.6253	73.212	21.032	0.6769	52.403	28.321
35	100	0.6983	48.895	27.917	0.6162	72.445	21.073	0.6767	52.324	28.502
40	100	0.6938	50.019	27.534	0.6465	70.822	20.763	0.6729	53.049	28.303

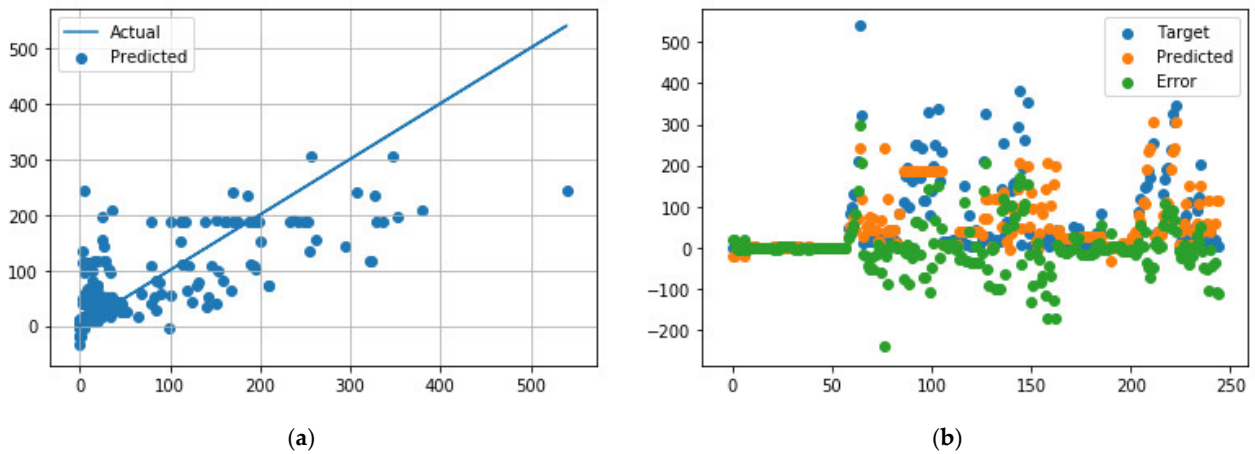


Figure 3. Comparison of predicted and measured ER values for APS + EB-PVD data points using the best NN model parameters. (a) Predicted values vs. actual values; (b) Distribution of actual values, predicted values, and errors.

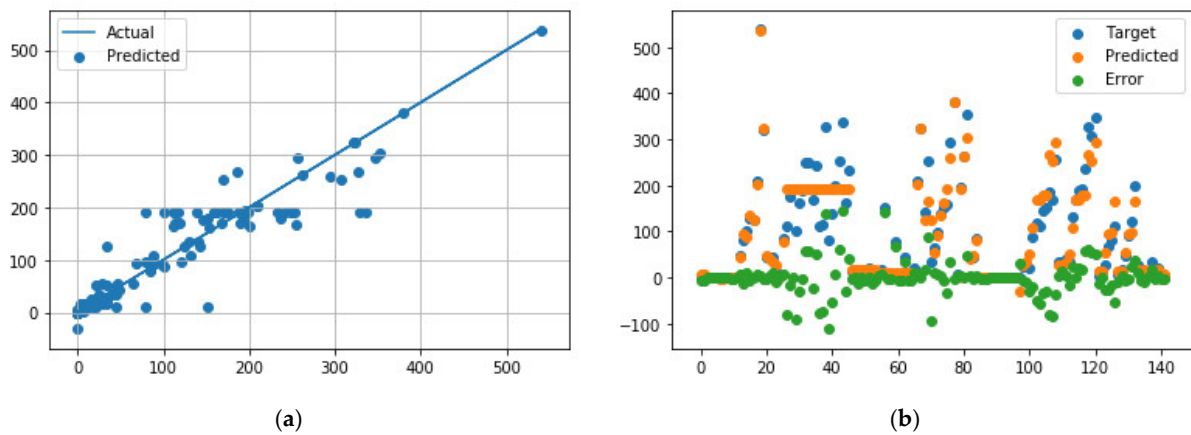


Figure 4. Comparison of predicted and measured ER values for APS data points using the best NN model parameters. (a) Predicted values vs. actual values; (b) Distribution of actual values, predicted values, and errors.

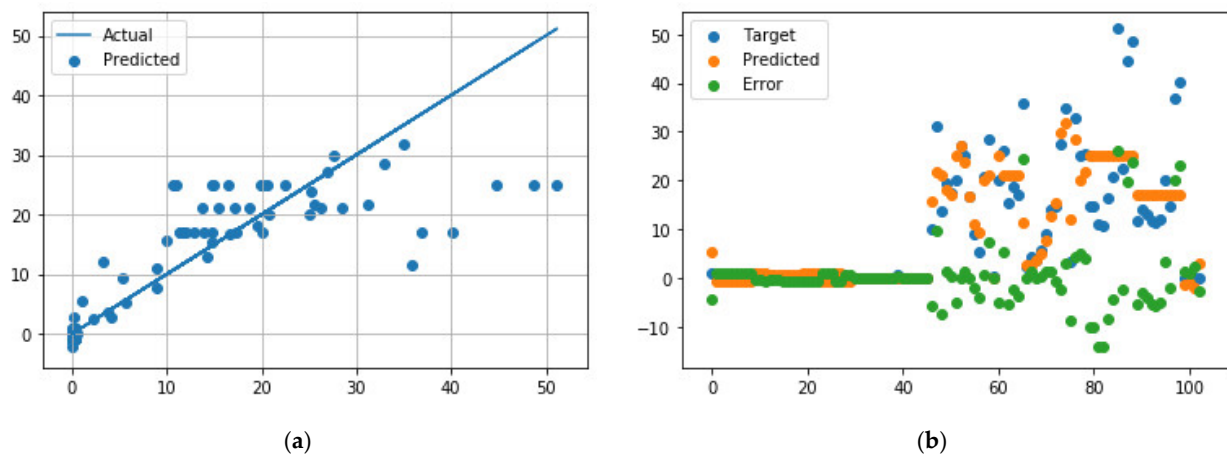


Figure 5. Comparison of predicted and measured ER values for EB-PVD data points using the best NN model parameters. (a) Predicted values vs. actual values; (b) Distribution of actual values, predicted values, and errors.

3.4. Gradient Boosting Approaches

3.4.1. Gradient Boosting Algorithm and Settings

Gradient boosting is a powerful machine-learning algorithm, which is widely used in application of regression, classification, and ranking. As its name implies, gradient boosting is the combination of gradient descent with boosting [23]. Boosting is an effective procedure to blend multiple base classifiers to produce a new form of committee, which can result in significantly better performance than any base classifier. The key concept of boosting is adding new models to the ensemble sequentially. At each specific iteration, a new base-learner model is trained regarding the error of the entire group learner so far.

Here, we present the basic methodology and learning algorithms of the GBR, which is derived first by Friedman [24]. The following three key components are involved in GBR:

1. A loss function to be optimised.

The loss functions applied can be arbitrary, but to give better intuition, if the error function is the classic squared-error loss, the learning procedure would result in consecutive error-fitting. In general, the choice of the loss function is up to the researcher, with both a rich variety of loss functions derived so far and the possibility of implementing one's own task-specific loss.

2. A weak-learner or base-learner model to make prediction.

It is common to constrain the base learners in specific ways, such as a maximum number of layers, nodes, splits, or leaf nodes. This is to ensure that the learners remain weak but can still be constructed in a greedy manner.

3. An additive model to add base learners to minimise the loss function.

The basic idea behind this algorithm is to build the new base learners to be maximally correlated with the negative gradient of the loss function, which is associated with the whole ensemble [Brownlee, web reference]. Generally, this approach is called functional gradient descent or gradient descent with functions.

Friedman's gradient boost algorithm can be summarised as the following steps [24]:

Inputs:

- Input data $(x, y)_{i=1}^N$
- Number of iterations M
- Choice of the loss-function $\psi(y, f)$
- Choice of the base-learner model $h(x, \theta)$

Algorithm:

1. Initialise \hat{f}_0 with a constant
2. For $t = 1$ to M ,
 - a. Compute the negative gradient $g_t(x)$
 - b. Fit a new base-learner function $h(x, \theta_t)$ for this iteration
 - c. Find the best gradient descent step-size ρ_t :

$$\rho_t = \underset{\rho}{\operatorname{argmin}} \sum_{i=1}^N \psi \left[y_i, \hat{f}_{t-1}(x_i) + \rho h(x_i, \theta_t) \right]$$

- d. Update the function estimate $\hat{f}_t \leftarrow \hat{f}_{t-1} + \rho_t h(x_i, \theta_t)$
3. End for

The main advantages of GBR includes:

- Natural treatment of data of varied types
- High predictive power
- Robustness to outliers in output space (via robust loss functions),
- Supports different loss functions.

The main disadvantage of GBR is scalability; due to the sequential nature of boosting, it can hardly be parallelised. Furthermore, GBR is slow to train.

3.4.2. Gradient Boosting Regression (GBR) Topology and Details

In the present work, the prediction of ER using GBR is performed in Python using scikit-learn library. First, an import of the library needs to be done:

```
import numpy as np
import numpy.ma as ma
import matplotlib.pyplot as plt
import csv
import xlswriter
import pandas as pd
from sklearn.ensemble import GradientBoostingRegressor as gbr
from sklearn import datasets
from sklearn.utils import shuffle
from sklearn.metrics import mean_squared_error as mse, r2_score
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split, GridSearchCV, RandomizedSearchCV,
cross_val_score
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_validate
from sklearn.model_selection import ShuffleSplit
from sklearn.preprocessing import StandardScaler.
```

Importing data and setting up the input and output dataset is the second step. In this study, we tried two different inputs settings for the GBR approach. The first one is the original five input variables (T Material, ImpactAngle, ImpactVelocity, ParticleSize, and Test-Temp). The other input setting is using five input variables and their second-degree polynomial terms. Codes regarding the data reading and setup are given as follows:

```
data = pd.read_csv("NRC2020_Data_ML_EBPVD_Corrected_Nor.csv", header = 0)
X0 = data.iloc[range(103),0:4] # five inputs
y = data.iloc[range(103),5]
sc = StandardScaler()
Xn = sc.fit_transform(X0)
from sklearn.preprocessing import PolynomialFeatures
poly_features = PolynomialFeatures(degree = 2, include_bias = False)
X = poly_features.fit_transform(Xn)
X, y = shuffle(X, y, random_state = 13)
X = X.astype(np.float32)
```

Then, the gradient boosting regression model can be done using the following codes:

```
params = {'n_estimators': 500, 'max_depth': 4, 'min_samples_split': 6, 'learning_rate': 0.1,
         'loss': 'ls'}
clf = gbr(**params)
clf = clf.fit(X_train,y_train)
scoring = ['r2']
```

More information about hyper-parameters settings is in the following Table 13. It has to be noted that there are more hyper-parameters for the model than are listed in the table; the details can be found in the scikit-learn 0.23.2 help document: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html> since 2007.

Table 13. Values of parameters of GBR.

Parameters	Description	Options or Values	Default Values
loss	Loss function to be optimised	String, 'ls', 'lad', 'huber', 'quantile'	ls
learning_rate	Learning rate shrinks the contribution of each tree	float, optional	0.055
n_estimators	The number of boosting stages to perform	int	400
subsample	The fraction of samples to be used for fitting the individual base learners	float	1.0
min_samples_split	The minimum number of samples required to split an internal node	int, float, optional	4
min_weight_fraction_leaf	The minimum weighted fraction of the sum total of weights (of all the input samples) required to be at a leaf node.	float, optional	0
max_depth	Maximum depth of the individual regression estimators	integer, optional	4
min_impurity_decrease	A node will be split if this split induces a decrease of the impurity greater than or equal to this value	float, optional	0
min_impurity_split	Threshold for early stopping in tree growth	float	1×10^{-7}
validation_fraction	The proportion of training data to set aside as the validation set for early stopping	float	0.1
tol	Tolerance for the early stopping	float, optional	1×10^{-4}
ccp_alpha	Complexity parameter used for minimal cost-complexity pruning	non-negative float, optional	0.0

3.4.3. Prediction Results Using Gradient Boosting Regression (GBR)

In our present study, three cases of prediction of ER using the GBR approach were performed for all, APS, and EB-PVD TBC data, respectively. Similar to neural network, the performance of ER prediction was evaluated using R^2 , MAXE, and MSE. For model validation, a holdout cross-validation is used by randomly splitting the data into 80% for training and 20% for testing, and this holdout cross-validation is repeated 10 times. The training and testing results are averaged for evaluating all the model candidates.

The performance results of ER prediction using GBR with different hyper-parameter settings are listed in Tables 14–16. The best prediction performance (using all subset results for each case) is obtained for APS TBC data cases, which provide overall good prediction performance, whose values of R^2 are all around 0.85. The EB-PVD TBC case also obtains relatively good results. However, for all the TBC data cases, the R^2 values are around 0.55, which are not satisfactory enough.

Table 14. Performances of ER prediction using the GBR approach for all TBC data.

N of Est.	Hyper-Parameters				Train			Test			All		
	Max Dep.	Min Sample Split	Learning Rate	Repeat	R^2	MSE	MAXE	R^2	MSE	MAXE	R^2	MSE	MAXE
500	4	6	0.01	10	0.6499	2906.54	265.85	0.2498	7390.64	370.12	0.5547	3814.25	378.42
500	4	2	0.01	10	0.6509	2898.25	263.29	0.2359	7531.86	377.20	0.5521	3836.23	383.77
100	4	6	0.01	10	0.5136	4029.38	322.12	0.2876	7036.15	347.88	0.4585	4638.04	397.26
1000	4	6	0.01	10	0.6531	2880.90	258.56	0.2116	7760.85	380.09	0.5483	3868.74	384.41
500	4	6	0.1	10	0.6533	2878.55	256.78	0.1973	7879.10	380.96	0.5457	3890.80	384.03
500	2	6	0.01	10	0.5846	3447.75	314.67	0.3552	6359.93	333.17	0.5286	4037.26	386.36

Table 15. Performances of ER prediction using the GBR approach for APS TBC data.

Hyper-Parameters					Train			Test			All		
N of Est.	Max Dep.	Min Sample Split	Learning Rate	Repeat	R ²	MSE	MAXE	R ²	MSE	MAXE	R ²	MSE	MAXE
10	4	6	0.2	10	0.8484	1785.11	157.87	0.6208	3959.35	170.14	0.8093	2229.15	179.94
500	4	6	0.2	10	0.8815	1395.88	146.29	0.6595	3447.56	161.86	0.8447	1814.88	163.96
500	4	6	0.01	10	0.8792	1422.32	147.88	0.6679	3403.43	159.31	0.8437	1826.91	166.05
500	4	2	0.2	10	0.8815	1395.88	146.29	0.6548	3534.03	167.08	0.8432	1832.54	170.08
500	6	6	0.2	10	0.8815	1395.88	146.29	0.6229	3853.18	175.80	0.8376	1897.72	176.89
500	2	6	0.2	10	0.8815	1395.90	146.29	0.6632	3518.42	168.74	0.8435	1829.37	171.89

Table 16. Performances of ER prediction using the GBR approach for EB-PVD TBC data.

Hyper-Parameters					Train			Test			All		
N of Est.	Max Dep.	Min Sample Split	Learning Rate	Repeat	R ²	MSE	MAXE	R ²	MSE	MAXE	R ²	MSE	MAXE
500	4	2	0.01	10	0.7705	34.98	29.52	0.4736	100.51	27.88	0.7000	48.34	30.83
500	6	2	0.01	10	0.7709	34.91	29.51	0.4707	103.48	28.04	0.6966	48.89	31.00
1000	4	2	0.01	10	0.7710	34.90	29.44	0.4709	100.89	27.82	0.6999	48.35	30.75
500	4	6	0.01	10	0.7702	35.02	29.55	0.4681	101.86	27.87	0.6981	48.65	30.81
500	4	6	0.1	10	0.7710	34.90	29.43	0.4652	102.33	27.72	0.6981	48.65	30.73
500	2	6	0.1	10	0.7420	39.30	30.62	0.4846	100.19	27.25	0.6791	51.72	31.12

Comparison of predicted and measured ER values and distribution of predicted error using GBR are illustrated in Figures 6–8 for all, APS, and EB-PVD TBC data, respectively. It appears that predictions using APS TBC data show good consistency with actual measurement data throughout the data range. For EB-PVD TBC, relatively better agreement between predicted and actual measurement data can be found for lower values of ER.

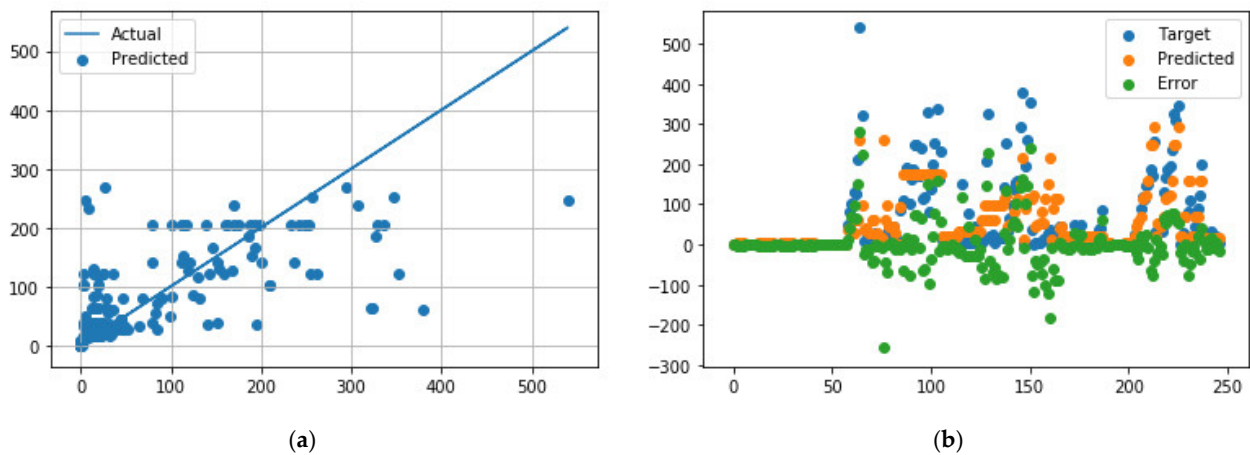


Figure 6. Comparison of predicted and measured ER values for APS + EB-PVD data points using the best GB model parameters. (a) Predicted values vs. actual values; (b) Distribution of actual values, predicted values, and errors.

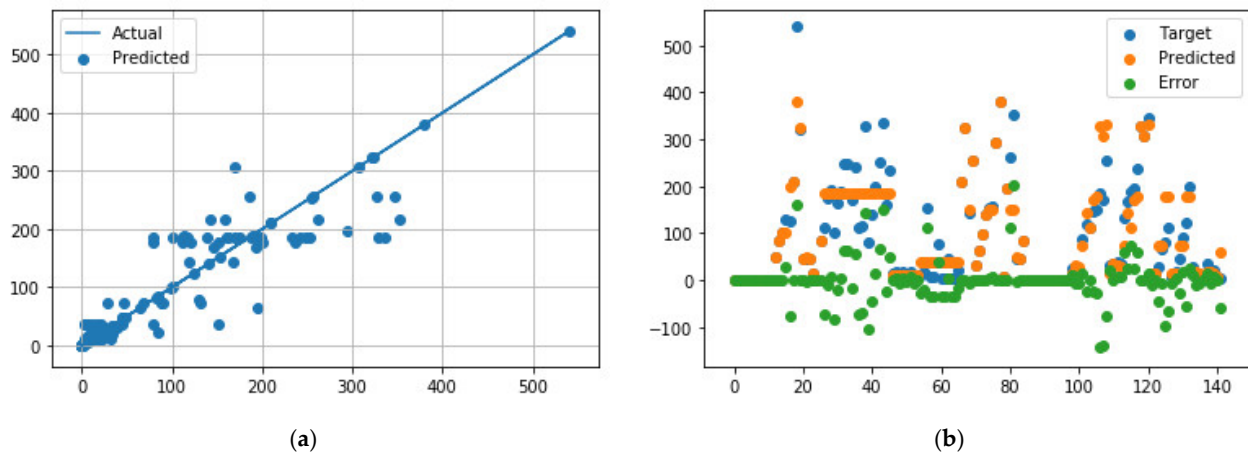


Figure 7. Comparison of predicted and measured ER values for APS TBC data points using the best GB model parameters. (a) Predicted values vs. actual values; (b) Distribution of actual values, predicted values, and errors.

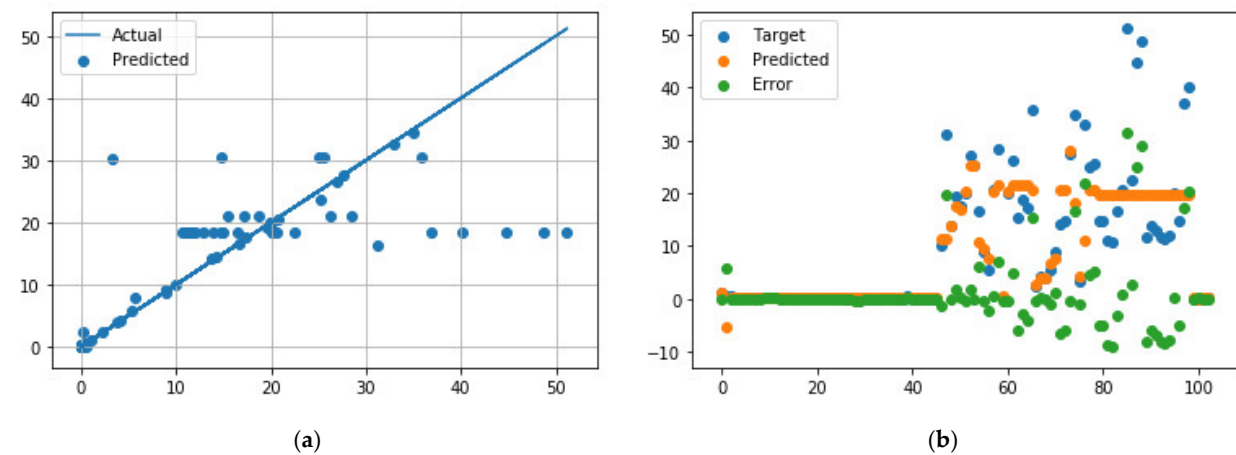


Figure 8. Comparison of predicted and measured ER values for EB-PVD TBC data points using the best GB model parameters. (a) Predicted values vs. actual values; (b) Distribution of actual values, predicted values, and errors.

3.5. Decision Tree Regression (DTR) Approaches

3.5.1. Decision Tree Regression Algorithm and Settings

Decision tree is a supervised machine learning model for predicting goals by learning decision rules from features. As the name implies, this model decomposes users' data by making decisions based on a series of questions. The decision tree algorithm typically consists of three steps: feature selection, decision tree generation, and decision tree pruning [25]. In a decision tree, each internal node is associated with a feature test also known as a split, and data falling into this node will be split into different subsets according to their different values on the feature test. Each terminal or leaf node is associated with a label, which will be assigned to data instances falling into this node. When new data are presented, the decision algorithm performs a series of feature tests starting from the root node, and the result is obtained when a leaf node is reached.

Although the concept of a decision tree is mainly based on the classification objective (classification), the same concept applies if our objective is real numbers (regression).

The process of solving regression problem with decision tree using Scikit Learn is performed by using the `DecisionTreeRegressor` class of the tree library:

```
from sklearn.tree import DecisionTreeRegressor
```

Similar to the GBR approach, after importing data, the gradient boosting regression model can be done using following codes:

```

clf = DecisionTreeRegressor(min_samples_split = 2, min_samples_leaf = 1, min_weight_fraction_leaf = 0,
                           min_impurity_decrease = 0, min_impurity_split = None, ccp_alpha = 0)
clf = clf.fit(X_train, y_train)
scoring = ['r2'].

```

More details of the hyper-parameters settings of DTR approach can be found in Table 17. In addition, it should be pointed out that there are more hyper-parameters for the model than are listed in the table; details can be found in the scikit-learn 0.23.2 help document: <https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html> since 2007.

Table 17. Hyper-parameters of DTR.

Parameters	Description	Options or Values	Default Values
criterion	The function to measure the quality of a split.	"mse", "friedman_mse", "mae"	"mse"
Splitter	The strategy used to choose the split at each node.	"best", "random"	"best"
max_depth	The maximum depth of the tree. If none, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.	integer, optional	None
min_samples_split	The minimum number of samples required to split an internal node.	int, float, optional	2
min_samples_leaf	The minimum number of samples required to be at a leaf node.	int, float, optional	1
min_weight_fraction_leaf	The minimum weighted fraction of the sum total of weights (of all the input samples) required to be at a leaf node.	float, optional	0
max_depth	Maximum depth of the individual regression estimators.	integer, optional	None
min_impurity_decrease	A node will be split if this split induces a decrease of the impurity greater than or equal to this value.	float, optional	0
min_impurity_split	Threshold for early stopping in tree growth.	float	0
random_state	Define the random number generator.	int, RandomState instance or none, optional	none
max_features	The number of features to consider when looking for the best split.	int, float, string or none, optional	none
max_leaf_nodes	Grow trees with max_leaf_nodes in best-first fashion.	int or none, optional	none
tol	Tolerance for the early stopping.	float, optional	1×10^{-4}
ccp_alpha	Complexity parameter used for Minimal Cost-Complexity Pruning.	non-negative float, optional	0.0

3.5.2. Decision Tree Regression (DTR) Results

The performance results of ER prediction using DTR with different hyper-parameter settings are found in Tables 18–20. Comparable with the BR model, it can be seen that the best prediction performance (using all subset results for each case) is obtained for APS TBC data cases, which provide overall good prediction performance, whose values of R^2 are all around 0.8377. The EB-PVD TBC case also obtains relatively good results. However, for the EB-PVD + APS TBC data case, the best R^2 values are around 0.54, which are mediocre.

Table 18. Performances of ER prediction using the DTR approach for all TBC data.

Hyper-Parameters					Train			Test			All		
Min Sample Split	Min Samples Leaf	Min Weight Fraction Leaf	Min Impurity Decrease	CCP Alpha	R^2	MSE	MAXE	R^2	MSE	MAXE	R^2	MSE	MAXE
2 (def)	1 (def)	0 (def)	0 (def)	0 (def)	0.6533	2878.55	256.77	0.1407	8428.65	385.56	0.5327	4002.05	388.63
4	1	0	0	0	0.6510	2897.31	259.24	0.1711	8149.88	385.56	0.5376	3960.58	391.09
6	1	0	0	0	0.6402	2987.76	263.82	0.1745	8114.65	359.30	0.5300	4025.59	363.85
2	2	0	0	0	0.6311	3063.11	258.91	0.2243	7638.51	355.67	0.5342	3989.31	361.50
2	4	0	0	0	0.5753	3524.22	304.93	0.2163	7730.95	354.78	0.4891	4375.78	396.79
2	1	0.1	0	0	0.4510	4556.44	332.56	0.2946	6887.12	328.96	0.4129	5028.24	398.04
2	1	0	0.1	0	0.6533	2878.77	256.77	0.1351	8484.42	385.36	0.5314	4013.52	388.43
2	1	0	0.5	0	0.6531	2880.49	256.77	0.1313	8547.40	387.86	0.5297	4027.64	390.83
2	1	0	0	0.1	0.6533	2878.72	256.77	0.1413	8423.97	386.98	0.5328	4001.24	390.05

Table 19. Performances of ER prediction using the DTR approach for APS TBC data.

Hyper-Parameters					Train			Test			All		
Min Sample Split	Min Samples Leaf	Min Weight Fraction Leaf	Min Impurity Decrease	CCP Alpha	R^2	MSE	MAXE	R^2	MSE	MAXE	R^2	MSE	MAXE
2 (def)	1 (def)	0 (def)	0 (def)	0 (def)	0.8815	1395.88	146.29	0.6165	3846.94	178.08	0.8377	1896.45	178.55
4	1	0	0	0	0.8660	1570.00	146.00	0.6180	3840.00	175.00	0.8260	2040.00	176.00
6	1	0	0	0	0.8475	1801.79	148.55	0.5798	4204.34	181.52	0.8039	2292.45	184.13
2	2	0	0	0	0.8511	1752.10	146.45	0.6365	3637.31	170.61	0.8172	2137.11	173.96
2	4	0	0	0	0.7605	2825.72	180.90	0.5668	4430.75	184.93	0.7302	3153.51	204.16
2	1	0.1	0	0	0.5477	5354.25	302.01	0.3449	6983.93	211.96	0.5134	5687.07	311.16
2	1	0	0.1	0	0.8814	1395.96	146.29	0.6001	3983.24	180.32	0.8354	1924.35	180.85
2	1	0	0.5	0	0.8814	1396.75	146.29	0.6152	3846.68	177.98	0.8377	1897.09	178.52
2	1	0	0	0.1	0.8814	1395.96	146.29	0.6027	3973.13	172.68	0.8355	1922.29	174.17

Table 20. Performances of ER prediction using the DTR approach for EB-PVD TBC data.

Hyper-Parameters				Train			Test			All			
Min Sample Split	Min Samples Leaf	Min Weight Fraction Leaf	Min Impurity Decrease	CCP Alpha	R^2	MSE	MAXE	R^2	MSE	MAXE	R^2	MSE	MAXE
2 (def)	1 (def)	0 (def)	0 (def)	0 (def)	0.7710	34.90	29.43	0.4369	110.31	28.39	0.6880	50.27	31.07
4	1	0	0	0	0.7655	35.76	29.43	0.4118	113.58	29.02	0.6796	51.62	31.07
6	1	0	0	0	0.7512	37.85	29.43	0.4335	110.41	28.67	0.6733	52.64	31.07
2	2	0	0	0	0.7382	39.97	29.45	0.4635	104.07	28.30	0.6709	53.04	30.94
2	4	0	0	0	0.6951	46.55	29.35	0.4574	106.00	27.65	0.6359	58.67	30.89
2	1	0	0.5	0	0.7619	36.27	29.34	0.3928	118.04	28.51	0.6715	52.94	31.01
2	1	0	0	0.1	0.7695	35.13	29.41	0.4147	112.96	28.79	0.6835	51.00	31.07

Comparison of predicted and measured ER values and distribution of predicted error using decision tree are illustrated in Figures 9–11 for all, APS, and EB-PVD TBC data, respectively. It appears that the predictions using three sets of data using DTR are very similar to those using GBR.

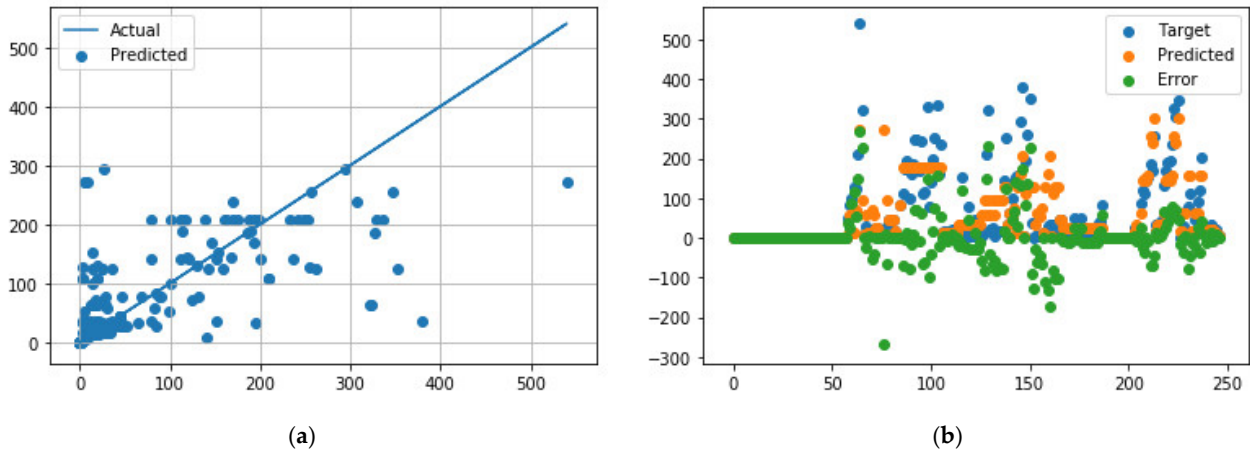


Figure 9. Comparison of predicted and measured ER values for all TBC data points using the best TDR model parameters. (a) Predicted values vs. actual values; (b) Distribution of actual values, predicted values, and errors.

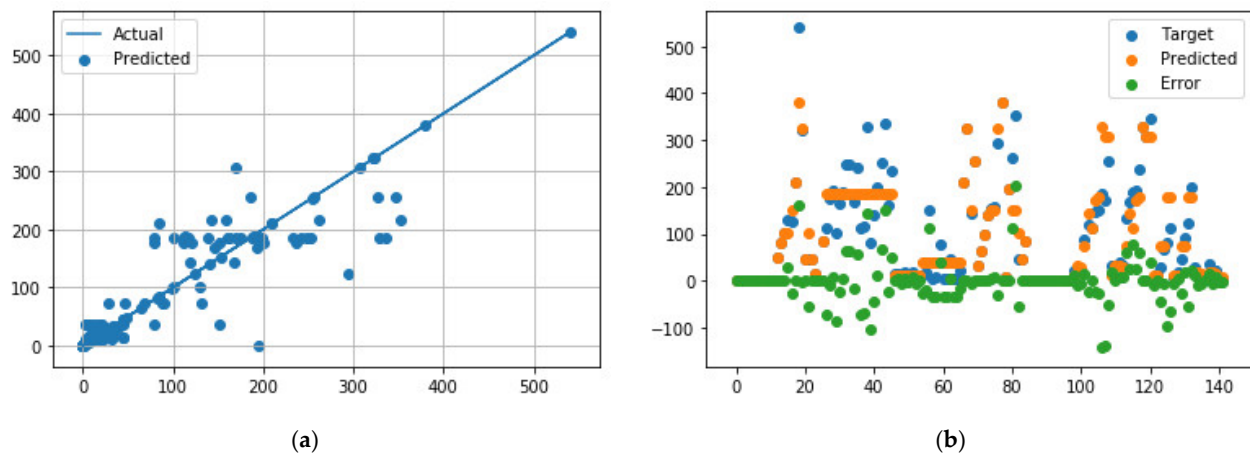


Figure 10. Comparison of predicted and measured ER values for APS TBC data points using the best TDR model parameters. (a) Predicted values vs. actual values; (b) Distribution of actual values, predicted values, and errors.

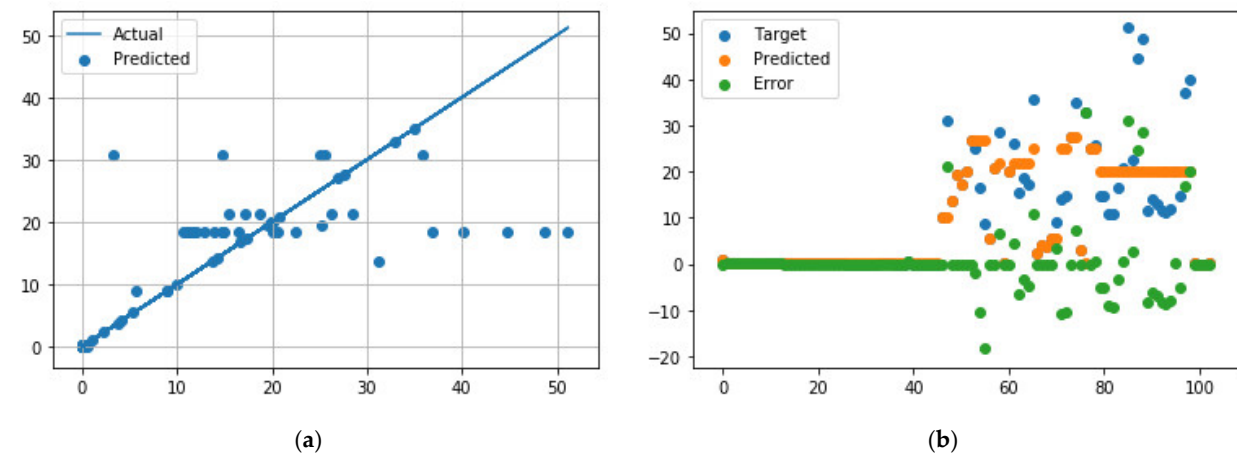


Figure 11. Comparison of predicted and measured ER values for EB-PVD TBC data points using the best TDR model parameters. (a) Predicted values vs. actual values; (b) Distribution of actual values, predicted values, and errors.

3.6. Random Forest Approaches

3.6.1. Random Forest Regression (RFR) Algorithm and Settings

A major drawback of the decision tree algorithm is that it leads to overfitting. This problem can be reduced or limited by implementing random forest regression instead of decision tree regression. It can be called the “forest” of trees and called “random forest” [26]. The term “random” is due to the fact that this algorithm is a forest of “randomly created decision trees”. A random forest is an integrated technique that is capable of performing regression and classification tasks using multiple decision trees and a technique called bootstrap aggregation.

Similar to the GBR and DTR approach, the prediction of ER using the RFR approach was performed using Python codes. The process of solving a regression problem with random forest using Scikit Learn is performed by using the RandomForestRegressor class of the ensemble library:

```
from sklearn.ensemble import RandomForestRegressor as rfr
```

After importing data, the gradient boosting regression model can be done using the following codes:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = seed)
clf = rfr(n_estimators = 500, min_samples_split = 2, min_samples_leaf = 1, ccp_alpha = 0)
clf = clf.fit(X_train, y_train)
scoring = ['r2'].
```

More details of the hyper-parameters settings of the RFR approach can be found in Table 21. In addition, it should be pointed out that there are more hyper-parameters for the model than are listed in the table; details can be found in the scikit-learn 0.23.2 help document: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html> since 2007.

Table 21. Hyper-parameters of RFR.

Parameters	Description	Options or Values	Default Values
n_estimators	The number of trees in the forest.	int	0
min_samples_split	The minimum number of samples required to split an internal node.	int, float	2
min_samples_leaf	The minimum number of samples required to be at a leaf node.	int, float	1
min_weight_fraction_leaf	The minimum weighted fraction of the sum total of weights (of all the input samples) required to be at a leaf node.	float, optional	0
max_depth	The maximum depth of the individual regression estimators.	integer, optional	None
min_impurity_decrease	A node will be split if this split induces a decrease of the impurity greater than or equal to this value.	float, optional	0
min_impurity_split	Threshold for early stopping in tree growth.	float	0
random_state	Define the random number generator.	int, RandomState instance or none, optional	none
max_samples	If bootstrap is true, the number of samples to draw from X to train each base estimator.	int, float	none
max_leaf_nodes	Grow trees with max_leaf_nodes in best-first fashion.	int or none, optional	none
tol	Tolerance for the early stopping.	float, optional	1×10^{-4}
ccp_alpha	Complexity parameter used for Minimal Cost-Complexity Pruning.	non-negative float, optional	0.0

3.6.2. Random Forest Regression (RFR) Results

The performance of ER prediction using RFR using different hyper-parameter settings can be found in Tables 22–24. Comparable with the BR model, it can be seen that the best prediction performance (using all the subset results for each case) is obtained for APS TBC data cases, which provide overall good prediction performance, whose values of R^2 are all around 0.8209. Meanwhile, this value is a little bit lower than the best R^2 values provided by the GBR and DTR models.

Table 22. Performances of ER prediction using the RFR approach for all TBC data.

Hyper-Parameters					Train			Test			All		
N of Est.	Max Depth	Min Sample Split	Min Samples Leaf	CCP Alpha	R^2	MSE	MAXE	R^2	MSE	MAXE	R^2	MSE	MAXE
10	None	2	1	0	0.6192	3160.65	285.89	0.2861	6938.43	322.91	0.5417	3925.38	351.92
100	None	2	1	0	0.6370	3013.81	281.62	0.2951	6921.69	342.92	0.5557	3804.88	369.92
200	None	2	1	0	0.6388	2998.72	274.95	0.2884	6981.54	345.30	0.5557	3804.96	366.62
500	None	2	1	0	0.6392	2995.33	271.86	0.2952	6913.27	344.76	0.5577	3788.44	361.61
500	4	2	1	0	0.5982	3330.92	273.38	0.3112	6760.09	342.04	0.5300	4025.08	357.77
500	None	4	1	0	0.6344	3035.32	281.25	0.3063	6801.32	339.83	0.5566	3797.67	365.95
500	None	2	2	0	0.6173	3176.67	296.32	0.3232	6643.01	332.11	0.5472	3878.35	374.16
500	None	2	1	0.1	0.6393	2995.18	270.86	0.2904	6975.11	345.43	0.5562	3800.83	362.98

Table 23. Performances of ER prediction using the RFR approach for APS TBC data.

Hyper-Parameters					Train			Test			All		
N of Est.	Max Depth	Min Sample Split	Min Samples Leaf	CCP Alpha	R^2	MSE	MAXE	R^2	MSE	MAXE	R^2	MSE	MAXE
10	None	2	1	0	0.8392	1895.39	161.97	0.6250	3866.68	161.25	0.8034	2297.98	179.14
10	4	2	1	0	0.7973	2388.49	158.98	0.5914	4259.93	164.94	0.7629	2770.68	180.05
100	None	2	1	0	0.8540	1721.07	145.01	0.6503	3637.27	162.41	0.8193	2112.40	172.57
200	None	2	1	0	0.8544	1715.10	146.13	0.6582	3583.94	163.83	0.8206	2096.76	173.64
500	None	2	1	0	0.8543	1717.74	146.05	0.6598	3558.09	159.80	0.8209	2093.59	170.63
500	None	4	1	0	0.8406	1881.33	155.41	0.6531	3621.78	161.01	0.8086	2236.77	177.53
500	None	2	2	0	0.8167	2162.93	186.94	0.6549	3664.26	163.54	0.7887	2469.54	205.84
500	None	2	1	0.1	0.8548	1710.61	146.14	0.6548	3617.42	160.30	0.8203	2100.03	171.64

Table 24. Performances of ER prediction using the RFR approach for EB-PVD TBC data.

Hyper-Parameters					Train			Test			All		
N of Est.	Max Depth	Min Sample Split	Min Samples Leaf	CCP Alpha	R^2	MSE	MAXE	R^2	MSE	MAXE	R^2	MSE	MAXE
10	None	2	1	0	0.7369	39.98	29.59	0.5000	99.22	26.31	0.6770	52.05	30.17
100	None	2	1	0	0.7468	38.54	29.60	0.5189	95.16	26.37	0.6892	50.09	30.45
200	None	2	1	0	0.7465	38.58	29.42	0.5171	94.80	26.49	0.6894	50.04	30.18
500	None	2	1	0	0.7473	38.47	29.46	0.5175	94.83	26.40	0.6900	49.96	30.25
500	4	2	1	0	0.7274	41.47	29.41	0.5244	93.29	26.37	0.6771	52.04	30.18
500	None	4	1	0	0.7405	39.49	29.39	0.5196	94.17	26.27	0.6857	50.64	30.18
500	None	2	2	0	0.7236	42.03	29.45	0.5265	92.58	26.60	0.6752	52.33	30.26
500	None	2	1	0.1	0.7468	38.54	29.45	0.5184	94.54	26.42	0.6900	49.96	30.25

Comparisons of predicted and measured ER values and distribution of predicted error using decision tree are illustrated in Figures 12–14 for all, APS, and EB-PVD TBC data, respectively. It appears that predictions using three sets of data using RFR are very similar to those using GBR and DTR. However, it can be clearly seen that for APS data,

the predicted value for the largest valued data point is much lower than the actual data, which reflects that RFR has no advantage over the other three models.

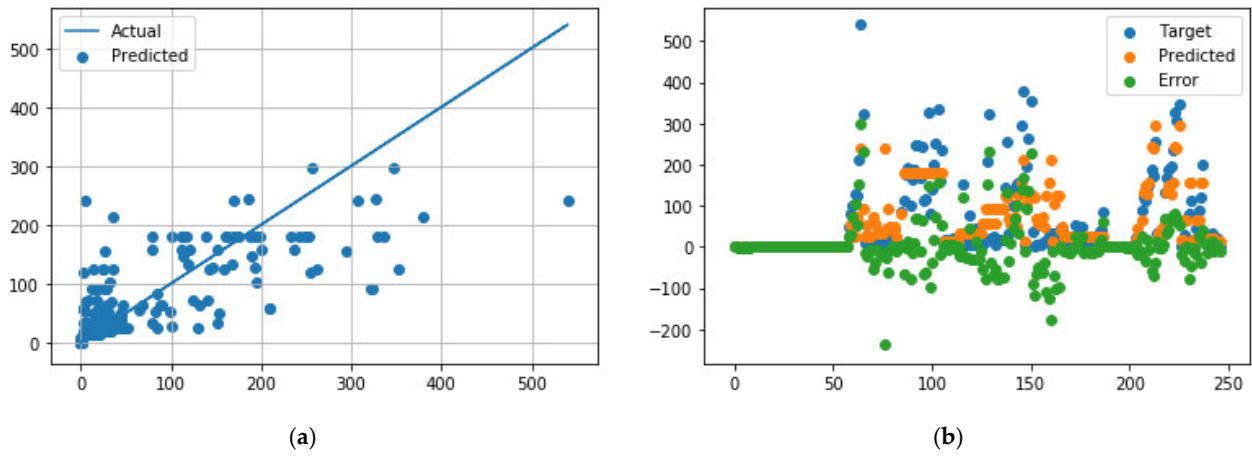


Figure 12. Comparison of predicted and measured ER values for all TBC data points using the best RFR model parameters. (a) Predicted values vs. actual values; (b) Distribution of actual values, predicted values, and errors.

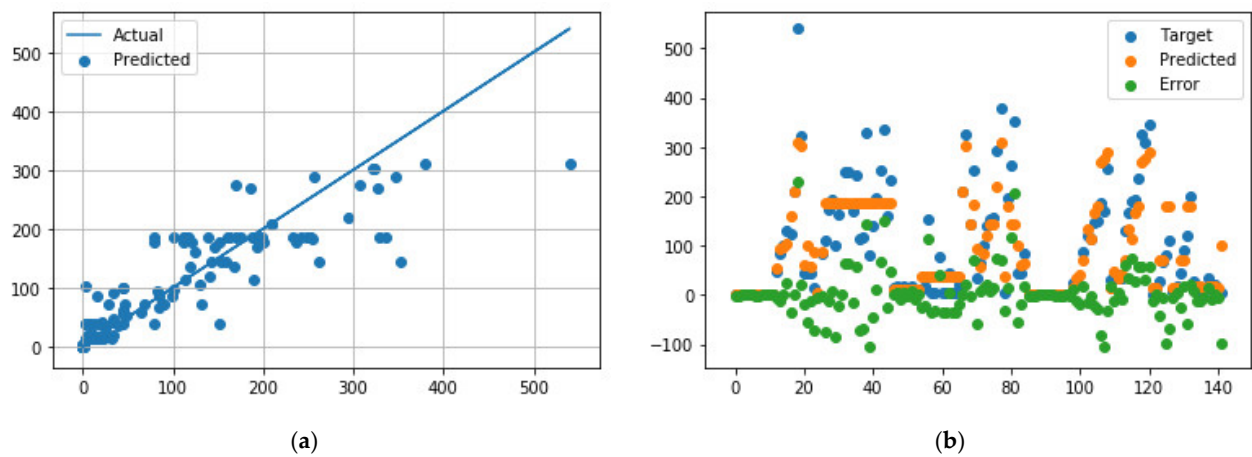


Figure 13. Comparison of predicted and measured ER values for APS TBC data points using the best RFR model parameters. (a) Predicted values vs. actual values; (b) Distribution of actual values, predicted values, and errors.

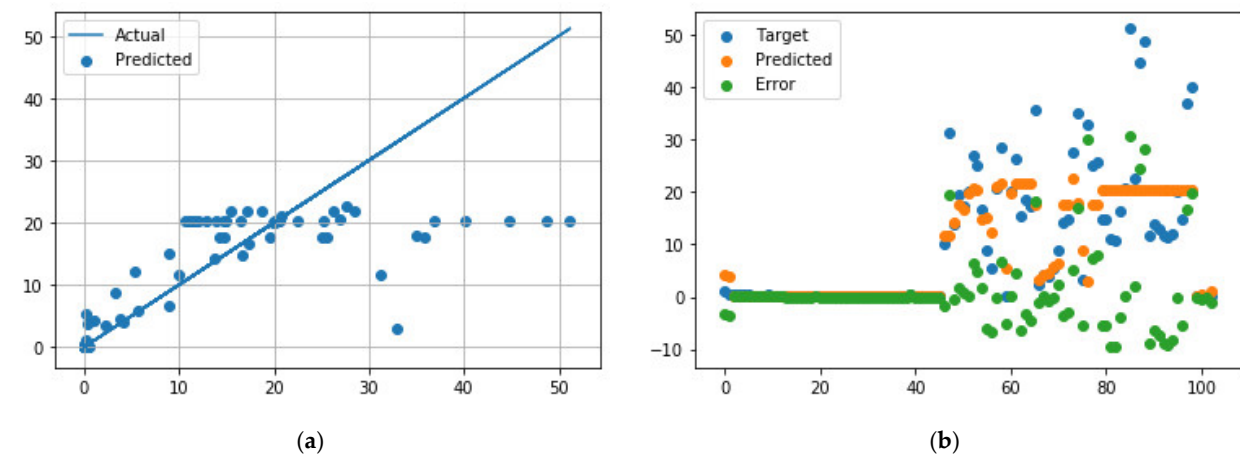


Figure 14. Comparison of predicted and measured ER values for APS TBC data points using the best RFR model parameters. (a) Predicted values vs. actual values; (b) Distribution of actual values, predicted values, and errors.

3.7. Summary of Prediction of Erosion Rates Using Machine Learning

In the present study, the prediction of erosion rate of APS and EB-PVD TBC from five input variables was investigated using four different machine learning approaches. The original dataset was gathered from experimental measurements of previous related research in the past three decades. This dataset contains six variables (impact angle, impact velocity, particle size, measurement temperature, and erosion rate) and the total number of samples are 245 for APS and EB-PVD TBC together.

Four machine learning algorithms were adopted, including Bayesian regularisation (BR) neural network (NN), gradient boosting regression (GBR), decision tree regression (DTR), and random forest regression (RFR). The comparisons of four approaches adopted in the present work are summarised in Tables 25–27 for the APS + EB-PVD, APS, and EB-PVD TBC cases, respectively.

Table 25. Comparison of ER prediction using different approaches for APS + EB-PVD TBC.

All		Train			Test			All	
Model	R^2	MSE	MAXE	R^2	MSE	MAXE	R^2	MSE	MAXE
NN	0.5436	3891.7	298.03	0.3211	8374.8	317.36	0.4838	4568.7	373.41
GBR	0.6499	2906.54	265.85	0.2498	7390.64	370.12	0.5547	3814.25	378.42
DTR	0.6510	2897.31	259.24	0.1711	8149.88	385.56	0.5376	3960.58	391.09
RFR	0.6392	2995.33	271.86	0.2952	6913.27	344.76	0.5577	3788.44	361.61

Table 26. Comparison of ER prediction using different approaches for APS TBC.

APS		Train			Test			All	
Model	R^2	MSE	MAXE	R^2	MSE	MAXE	R^2	MSE	MAXE
NN	0.8687	1535.6	149.59	0.6991	3946.7	170.84	0.8400	1892.2	184.01
GBR	0.8815	1395.88	146.29	0.6595	3447.56	161.86	0.8447	1814.88	163.96
DTR	0.8815	1395.88	146.29	0.6165	3846.94	178.08	0.8377	1896.45	178.55
RFR	0.8392	1895.39	161.97	0.6250	3866.68	161.25	0.8034	2297.98	179.14

Table 27. Comparison of ER prediction using different approaches for EB-PVD TBC.

EB-PVD		Train			Test			All	
Model	R^2	MSE	MAXE	Model	R^2	MSE	MAXE	Model	R^2
NN	0.7147	46.42	27.294	0.5868	81.48	22.99	0.6827	51.52	28.25
GBR	0.7705	34.98	29.52	0.4736	100.51	27.88	0.7000	48.34	30.83
DTR	0.7710	34.90	29.43	0.4369	110.31	28.39	0.6880	50.27	31.07
RFR	0.7473	38.47	29.46	0.5175	94.83	26.40	0.6900	49.96	30.25

The results suggest that for APS TBC, all four approaches show overall good prediction performance on the erosion rates. In addition, the GBR approach needs a relatively short computing time and better results. For EB-PVD TBC, the results of the GBR approach also showed better agreement with real measurement data. When APS and EB-PVD data are considered together, the results of prediction are unsatisfactory, indicating the importance of the TBC types on the erosion rate. In summary, machine learning is a promising tool for material science predictions, especially for the highly complex TBC system.

4. Exploration of Application of Deep Learning on Erosion Rates of TBCs

Research was done to create qualitative and quantitative characterisation of the microstructural features of TBCs, which can be applied to the study the erosion rates.

4.1. Literature Review on Characterisation for the Microstructural Features of TBCs Using Machine-Learning and Deep-Learning Approaches

The primary motivation for exploring the use of ML and DL models in this project is to expedite the prediction of microstructure mechanical properties of TBCs compared to more expensive simulations involving physics-based constitutive models.

The microstructural characterisation and identification of TBCs is an extremely challenging task. Each microstructure is an integral part of TBCs and is required to provide thermal conductivity and to accommodate operational thermal stresses and reduce erosion of the TBCs. Accurate characterisation of the TBCs microstructure is difficult due to the complex pore morphology and ultra-thin coating thickness. Usually, gathering and analysis of the coating microstructural features are performed experimentally and analytically. Of all this experimental and analytical work, image analysis is one of the most important steps. Image analysis can be defined as different computer-based levels for identification, description, and diagnosis of the elements from an image. During the past few years, machine-learning (ML) and deep-learning (DL) approaches have made incredible progress and have been widely utilised for prediction. In the material science, ML and DL have also been used by many researchers and have been proven to be able to provide satisfactory accurate results when adopted for the microstructure characterisation and quantification. In this section, a survey and literature review will be given on characterisation for the microstructural features of TBCs using machine-learning and deep-learning approaches or deep-learning based models.

4.1.1. Support Vector Machine Method Optimised by the Cuckoo Search Algorithm (CS-SVM)

A novel hybrid machine-learning method was proposed by Ye et al. [27] to predict the microstructural features of TBCs using thermal spray processing parameters based on a support vector machine method optimised by the cuckoo search algorithm (CS-SVM). In this work, atmospheric-plasma-sprayed (APS) TBCs samples with multifarious microstructural features were acquired by modifying the spray powder size, spray distance, and spray power during thermal spray processing. The processing parameters were used as the inputs for the CS-SVM model. Then, the porosity, the pore-to-crack ratio, the maximum Feret's diameter, the aspect ratio, and the circularity were counted and treated as the targets for the CS-SVM model.

The cuckoo search algorithm is a meta-heuristic algorithm developed by Xin-she Yang and Suash Deb in 2009 [28]. The algorithm was developed based on the parasitic reproduction strategy possessed by the cuckoo population itself [28]. The CS algorithm uses egg nests on behalf of the solutions. In the oversimplified case, there is one egg per nest, and the cuckoo's egg signifies a new solution. The aim is to adopt new and latently better solutions instead of suboptimal solutions.

The predicted results obtained after the optimisation and training procedure of the CS-SVM model were compared to the results of experimental data. As a result, the squared correlation coefficient (R^2) of CS-SVM model showed that the prediction accuracy was over 95%, and the root mean square error (RMSE), mean absolute error (MAE), and mean absolute percentage error (MAPE) were less than 0.1. All these metrics verified the reliability of the CS-SVM model.

4.1.2. K-Means Clustering Algorithm

The K-means clustering algorithm, which retains the spatial information of the constituent phases, was used by Vignesh et al. [29] to deconvolute the data to determine the phase-level properties. The details of the K-means clustering algorithm were described in the work of Hartigan and Wong [30]. The main principle of k -means is to partition ' n ' observations into ' k ' clusters. K-means is an iterative refinement technique. It moves the center of the cluster to a new point, k cluster centers are randomly initialised, and then, we minimise the sum of square distances within the cluster in each iteration. When the

distance within the cluster cannot be reduced, this iterative process converges, thus separating the data into a predetermined number of clusters. The number of clusters into which the data has to be binned can be determined either by prior knowledge of the number of phases present in the mapped region or by iteratively running the algorithm for a different number of clusters and picking the optimal number by error minimisation. As the algorithm partitions the data in such a way that the data points within the same cluster are more similar than those in other clusters, it is expected that each cluster represents a distinct phase/feature in the spatial map. Once clustered, the mean and standard deviation of the data points in each cluster can be used as a quantitative measure of the property of the corresponding phase. This method of deconvolution has been applied to all spatial property maps acquired from the bond coat, top coat, and bond coat–top coat interface regions to obtain the properties of individual phases/features (β -NiAl, γ/γ' -Ni, YSZ, TGO) present in them. Unlike the Gaussian deconvolution-based methods, this method preserves the spatial information of each constituent phase, which is useful for studying the evolution of a local mechanical property.

The high speed nanoindentation results obtained by Vignesh et al. [29] show good agreement with existing literature and experimental results. The phase-level characteristics obtained by using this approach can be easily used for microstructure-based finite element analysis, and the large datasets obtained using extensive mapping can also be used to develop data-driven models and to predict the remaining life of TBCs.

4.1.3. Constructed BP-GPR Algorithm Was Presented Based on URCAS Technique

A novel method for characterising the TBCs porosity through a BP neural network optimising the Gaussian process regression (GPR) algorithm, which is termed the BP-GPR algorithm and based on an ultrasonic reflection coefficient amplitude spectrum (URCAS), was presented and reported by Ma et al. [31–33]. First, the characteristic parameters of the URCAS are optimised using BP neural network combined with high determination coefficient R^2 rules. Then, the optimisation parameters are used to train the GPR algorithm for predicting unknown TBCs porosity.

The BP-GPR method was demonstrated through a series of finite element method (FEM) simulations, which were implemented on random pore models (RPMs) of a plasma spraying ZrO_2 coating with a thickness of 300 μm and porosities of 1%, 3%, 5%, 7%, and 9%. Simulation results indicated that the relative errors of the predicted porosity of RPMs were 6.37%, 7.62%, 1.07%, and 1.07%, respectively, which have 32% and 48% higher accuracy than that predicted only by the BP neural network or GPR algorithm. It is verified that the proposed BP-GPR method can accurately characterise the porosity of TBCs with complex pore morphology.

4.1.4. Convolutional Neural Networks (CNNs)

Different types of deep neural networks were studied for surface defect analysis, in which the optimal structure was obtained for the best performance [34]. Among many deep neural network models, convolutional neural networks (CNNs) are commonly applied to analyse visual imagery and have achieved much success in image classification. Recently, CNNs have also been utilised to study the effective characteristics of complex materials and showed much potential for the efficient and accurate prediction of a material's effective properties from its structure (e.g., presented in the form of images). Convolutional neural networks (CNNs) are representative deep learning architectures that have been used for defect detection due to their strong feature extraction capabilities [35]. Based on convolutional neural networks, the method of defect segmentation in manufacturing is also proposed, especially to deal with this situation when the dataset is small scale [36].

Azimi et al. [37] propose a deep learning method for microstructural classification in the examples of certain microstructural constituents of low-carbon steel. This novel method employs pixel-wise segmentation via fully convolutional neural network (FCNN) accompanied by a max-voting scheme.

Wu et al. [38] report their application of deep learning methods for predicting the effective diffusivity (D_e) of two-dimensional porous media from images of their structures. Pore structures are built using reconstruction methods and represented as images, and their effective diffusivity is computed by lattice Boltzmann (LBM) simulations. The datasets thus generated are used to train convolutional neural networks (CNNs) models and evaluate their performance.

Some research also proved that CNNs are able to achieve satisfied accuracy results when deployed for microstructure quantification within thermal spray coatings. Chen et al. [39] collected the ground truth of the porosity mask on two full-size images and then randomly selected pixels in the top coat layer (TCL) to form sub-images centred at the pixel and with different sizes to train convolutional neural networks (CNNs) with various architecture. Their proposed approach was evaluated on a dataset of 150 images of size 100 by 100 randomly selected from a set of 30 high-resolution thermal barrier coating images. Their results indicate that the CNN-based models achieved higher average classification accuracy of 100% at the confidence level of 90% for a VGG16-based model and a lower average relative error (ARE) of 0.113.

The work of Chen et al. was enhanced by Lu et al. [40]. An improved CNNs methodology was proposed by implementing some additional operations, including data augmentation and transfer learning. In their work, the images collected correspond to coatings from three types of powders generated by (1) Type A: Metco 601NS, (2) Type B: Metco 995C, and (3) Type C: Metco 204BNS. The size of our dataset is 159 raw images, where the number of Type A, Type B, and Type C images are 50, 49, and 60 respectively.

The overall training process proposed by Lu et al. [40] can be summarised in the following seven steps:

1. Procure ground truth mask of a raw image by manually labelling each pixel as one of the four classes, including (1) mount material, (2) top coat layer—microstructure, (3) top coat layer—coating material, and (4) substrate.
2. Extract the topcoat layer (TCL) from the raw image according to the ground truth mask.
3. Choose a sub-image size, and then, randomly select pixels in the TCL and use the individual pixel selected as the center to crop sub-images in the chosen size.
4. Augment the cropped sub-images by mirroring and rotating sub-images to form a training dataset.
5. Resize the sub-images and feed them into CNNs with pre-trained parameters to train CNN models;
6. Select the best CNN model;
7. Recognise microstructure/coating material in a pixel-wise manner in the TCL area, which was obtained according to the ground truth mask, by applying the CNN model on sub-images extracted from the TCL area to evaluate the performance of the CNN model for microstructure and coating material classification.

The CNN model that gives the highest classification accuracy among the five models corresponding to that combination of CNN model and sub-image size was assumed as the best model.

4.1.5. Image Processing Tools

ImageJ (ImageJ software, NIH, Bethesda, MA, USA) is an open-source code software that was used by a lot of researchers in the field of material science [41]. The source codes and more information are available at <https://imagej.nih.gov> since 2011.

Ghai et al. developed five-phase model of thermal conductivity of porous thermal barrier coatings. The flowchart describing the process to obtain the thermal conductivity of a coating can be seen in Figure 15. As the first step to propose the model, the length, height, and orientation angle of the defects from the SEM images of TBCs were obtained using Image J software.

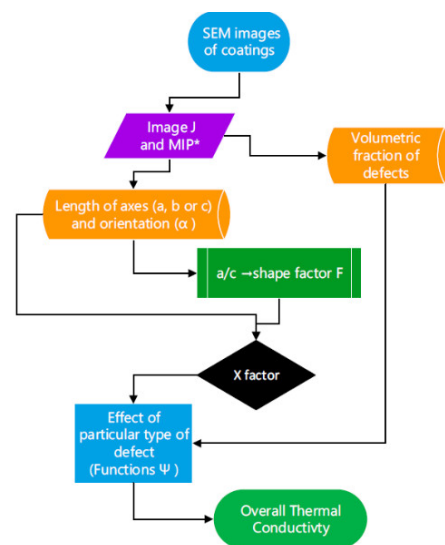


Figure 15. The process to obtain the thermal conductivity of a coating [41].

Bolelli et al. [42] studied how the nanomechanical properties of thermal barrier coatings (TBCs) vary during thermal cycling, as a way to shed new light on their failure mechanisms. In Section 2.2 of the paper of Bolelli et al., the image processing of SEM micrographs was provided. The thickness of the TGO scale developed on top of the bond coat in the thermally cycled samples was measured by image analysis (ImageJ software, NIH, Bethesda, MA, USA) using ten $4000\times$ FEG-SEM micrographs acquired in backscattered electrons (BSE) contrast mode. The porosity of thermally cycled top coats was measured on polished cross-sections by image analysis (Leica Application Suite software), using three $100\times$ optical micrographs each covering a field of view of 1.36 mm.

4.2. Using Deep Learning Methods for Prediction

4.2.1. Convolutional Neural Networks (CNNs) in Python with Karas

The long-term goal of application deep learning is collecting SEM images of TBCs and processing the images using the deep learning approach to extract information from the microstructure of the TBCs, including the porosity, pore-to-crack ratio, and pore size (equivalent diameter), and applying those microstructure features to the prediction or evaluation of TBC thermal-mechanical properties and life prediction.

We mainly focus on the prediction of porosity and data augment. Possible ML and DL modelling approaches (multiple linear regression (MLR), backpropagation (BP) neural network, support vector machine (SVM), and convolutional neural networks (CNNs)) might be the candidates to characterise the microstructural features.

In this work, convolutional neural networks (CNNs) are chosen as a deep learning tool because it is the most influential and successful innovation in the field of computer vision. CNNs have shown their success in many practical case studies and applications.

In our present work, a preliminary test of CNNs is inbuilt by using Python (Version 3.7, in Spyder/Anocoda Navigator), and the steps are briefly described below:

1. The first step is collecting data. Since this is the preliminary test case, the dataset adopted here is the Fashion-MNIST dataset, which is a dataset of Zalando's article images, with 28×28 grayscale images of 70,000 fashion products from 10 categories and 7000 images per category.
2. The second step is importing libraries including Tensorflow (an open-source software library for dataflow programming across a range of tasks), Keras (an open-source neural network library written in Python), and CNN (convolution neural network, a class of deep, feed-forward artificial neural networks). The codes used to import libraries can be seen below:

```

import keras
from keras.models import Sequential, Input, Model
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D, MaxPooling2D
from keras.layers.normalization import BatchNormalization
from keras.layers.advanced_activations import LeakyReLU
import numpy as np
from keras.utils import to_categorical
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from keras.datasets import fashion_mnist.

```

3. The third step is splitting the dataset: the training set has 60,000 images, and the test set has 10,000 images. The codes related to data splitting are listed below:

```

train_X, valid_X, train_label, valid_label = train_test_split(train_X, train_Y_one_hot,
    test_size = 0.2, random_state = 13)
print("Training data shape: ", train_X.shape, train_Y.shape)
print("Testing data shape: ", test_X.shape, test_Y.shape).

```

4. Then, the CNN model is built by three sub-steps: convolution, pooling, and flattening. Codes related to the building of CNN models are

```

batch_size = 64
epochs = 20
num_classes = 10
fashion_model = Sequential()
fashion_model.add(Conv2D(32, kernel_size = (3, 3), activation = 'linear', padding = 'same',
    input_shape = (28,28,1)))
fashion_model.add(LeakyReLU(alpha = 0.1))
fashion_model.add(MaxPooling2D((2, 2), padding = 'same'))
fashion_model.add(Dropout(0.25))
fashion_model.add(Conv2D(64, (3, 3), activation = 'linear', padding = 'same'))
fashion_model.add(LeakyReLU(alpha = 0.1))
fashion_model.add(MaxPooling2D(pool_size=(2, 2), padding = 'same'))
fashion_model.add(Dropout(0.25))
fashion_model.add(Conv2D(128, (3, 3), activation='linear', padding = 'same'))
fashion_model.add(LeakyReLU(alpha=0.1))
fashion_model.add(MaxPooling2D(pool_size = (2, 2), padding = 'same'))
fashion_model.add(Dropout(0.4))
fashion_model.add(Flatten())
fashion_model.add(Dense(128, activation = 'linear'))
fashion_model.add(LeakyReLU(alpha = 0.1))
fashion_model.add(Dropout(0.3))
fashion_model.add(Dense(num_classes, activation = 'softmax'))
fashion_model.compile(loss = keras.losses.categorical_crossentropy, optimizer = keras.optimizers.
    Adam(), metrics = ['accuracy']).

```

5. Then, the CNN model can be trained and tested using image data, and predicted results can be printed, exported, and analysed. In this preliminary test case, it can be seen from Figure 16 that the test loss and accuracy is improved by using a larger epochs number.

- epochs = 4
 Test loss: 0.2750724524140358
 Test accuracy: 0.8963000178337097
 Found 8897 correct labels
 Found 1103 incorrect labels

- epochs = 10
Test loss: 0.23426873979568483
Test accuracy: 0.9126999974250793
Found 9081 correct labels
Found 919 incorrect labels

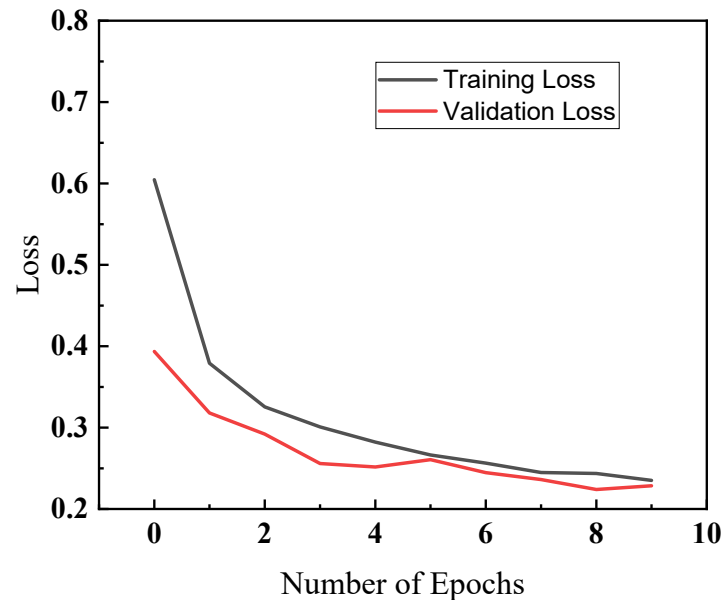


Figure 16. Training and validation loss (Y axis) for different epochs (X axis) using the CNNs model for a fashion preliminary case.

4.2.2. Work Plan for Next Steps

- Image Collecting

Possible image sources include TBC SEM micrographs collected from the literature and experimental images from NRC. Based on the information gathered from the literature review, the amount of image data to be collected should not be less than 200.

- Image data augmentation

Overfitting refers to the phenomenon in the process of ML or DL with a very large amount of data and very high variance such as to perfectly model the training data. On the contrary, many application domains cannot access big data, such as topics related to TBC microstructures. Image data augmentation is a technique that can be used to artificially expand the size of a training dataset by creating modified versions of images in the dataset.

- Perform data training using available data and a CNNs model to obtain porosity data or porosity classification information.
- Check ImageJ software source codes to optimise the image acquisition and procession by utilising ML and DP approaches.

5. Conclusions

Artificial intelligence technology comprising of machine learning (ML), deep learning (DL), and big data (BD) treatment approaches have been evolving at a rapid pace to facilitate material designing, developments, and property prediction. This project has undertaken the prediction of erosion rate (ER) for two types of thermal barrier coatings (EB-PVD and APS) using machine learning models and methodology. In addition, some efforts have been made to explore the possibility of the application of DL approaches to the microstructure characterisation of TBCs.

1. Experimental erosion rate (ER) data have been compiled from literature for YSZ TBCs and EB-PVD TBCs used for the training, testing, and validation of ML models.

A dataset was collected containing five inputs (wt.% of Y_2O_3 , impact angle, impact velocity, particle size, and measurement temperature) and the total number of samples is 245. Following data collection, the dataset is subjected to sorting, filtering, extracting, and exploratory analysis. PCA analysis in this study shows that APS TBC data, impact velocity, particle size, and impact angle show enough correlation with the output variable ER. Regarding the EB-PVD data, test temperature shows the largest correlation with the ER.

2. A quick analysis was performed to check ten ML models using the Tecsis Model Selection Tool (MST). It can be seen that the gradient boosting regression (GB), random forest regression (RF), decision tree regression (DT), and Adaboost classifier (AC) approaches provide a relatively good prediction.
3. Based on the results of the quick analysis, the prediction of erosion rates of two types of TBCs was performed using four approaches, namely neural networks (NNs), gradient boosting regression (GBR), decision tree regression (DTR), and random forest regression (RFR). The training and testing and prediction results are analysed, presented, and discussed at length in the report. The results indicate that for APS TBC, all four approaches show overall good prediction performance on the erosion rates. In addition, the GBR approach needs a relatively short computing time and better results. For EB-PVD TBC, the results of the GBR approach also showed better agreement with real measurement data. When APS and EB-PVD data are considered together, the results of prediction are unsatisfactory, indicating the importance of the TBC types on the erosion rate prediction. In brief, machine learning is a promising tool for material science predictions, especially for the highly complex TBC systems.
4. A preliminary exploration of applications of deep learning approaches on the characterisation of microstructures of TBCs was also performed. A review and survey work have been performed on the available research on the evaluation and prediction of materials' microstructure properties, especially on the porosity using deep learning approaches in the past several years. The review work helps us understand what types of deep learning approaches can be adopted to improve the accuracy of measurements and prediction of microstructure properties. In addition, a preliminary deep learning case using CNNs models was studied in the Python and Keras environment.

6. Future Work

This research work is part of collaborations between Tecsis and the Aerospace Division, National Research Council of Canada (NRC, Canada) to study the application of machine learning in predicting thermomechanical properties (including thermal conductivity, erosion rates) and microstructures for TBC applications. The possible extensions of this preliminary work reported here are as follows:

- In this work, four different machine learning methods, including feedforward neural networks (NNs), gradient boosting regression (GBR), decision tree regression (DTR), and random forest regression (RFR), were investigated for the prediction of erosion rates of different types of TBCs. It will be desirable to investigate various other advanced regression models and algorithms such as support vector regression (SVR), Gaussian process regression (GPR), and radial basis function (RBF) neural networks for the prediction of TBC conductivity.
- Hyper-parameters (for example, the number of hidden layers and nodes of neural networks, maximum depth in the case of the decision tree) setting is a key step in the process of machine learning and deep learning and has a big influence on the performance of machine learning algorithms. In the materials science application of machine learning algorithms, these hyper-parameters are generally chosen by trial-and-error methods or based on experience. In this work, we have adopted trial and error experiments or simple grid searches to determine these hyper-parameters. For future work, it will be beneficial to investigate ways to perform hyper-parameters optimisation (best hyper-parameters searching) for the above-mentioned advanced

regression models and algorithms using some global optimisation techniques such as the genetic algorithm, genetic programming (GP), differential evolution (DE), the evolution strategy (ES), and evolutionary programming (EP) [43].

- Collecting SEM images of TBC and gathering enough ground truth (information provided by direct observation) on the raw images is the first and important step of creating qualitative and quantitative characterisation for the microstructural features of TBCs. More efforts will be made in this step to generate a TBC SEM images database.
- Study the data augmentations based on deep learning is also a research direction that has received much attention. To build useful deep learning models, the validation error must continue to decrease with the training error. For TBC microstructure applications, data augmentation is a very powerful method in the condition of limited numbers of images. In the future, a study will be performed to study different image data augmentations techniques including flipping, color space, cropping, translation, noise injection, color space transformation, etc.
- In addition to the CNNs model, some new background subtraction DL algorithms [44,45], such as Generative Adversarial Networks (GANs), Foreground Generative Adversarial Network (FgGAN), and Deep Auto Encoder Networks (DAE) can be applied to the prediction/measurement of porosity (or other microstructure of TBC from SEM images).

Author Contributions: Y.L. and R.R.: Conceptualization, Methodology, Validation, Investigation, Writing—original draft. K.C.: Conceptualization, Writing—review and editing, Project administration, Resources, Funding acquisition. P.P.: Writing—review and editing, Supervision, Resources, Funding acquisition. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Research Council Canada, DTS Program, A1-016131.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data will be made available on reasonable request.

Conflicts of Interest: The authors report no declarations of interest.

References

1. Swar, R.; Hamed, A.; Shin, D.; Woggon, N.; Miller, R. Deterioration of Thermal Barrier Coated Turbine Blades by Erosion. *Int. J. Rotating Mach.* **2012**, *2012*, 601837. [[CrossRef](#)]
2. Nicholls, J.; Lawson, K.; Johnstone, A.; Rickerby, D. Methods to reduce the thermal conductivity of EB-PVD TBCs. *Surf. Coat. Technol.* **2002**, *151–152*, 383–391. [[CrossRef](#)]
3. Lee, J.-H.; Yi, H.; Jeon, Y.-S.; Won, S.; Chun, J. TBC: A clustering algorithm based on prokaryotic taxonomy. *J. Microbiol.* **2012**, *50*, 181–185. [[CrossRef](#)]
4. Algenaid, W.; Ganvir, A.; Calinas, R.F.; Varghese, J.; Rajulapati, K.V.; Joshi, S. Influence of microstructure on the erosion behaviour of suspension plasma sprayed thermal barrier coatings. *Surf. Coat. Technol.* **2019**, *375*, 86–99. [[CrossRef](#)]
5. Janos, B.; Lugscheider, E.; Remer, P. Effect of thermal aging on the erosion resistance of air plasma sprayed zirconia thermal barrier coating. *Surf. Coat. Technol.* **1999**, *113*, 278–285. [[CrossRef](#)]
6. Fleck, N.; Zisis, T. The erosion of EB-PVD thermal barrier coatings: The competition between mechanisms. *Wear* **2010**, *268*, 1214–1224. [[CrossRef](#)]
7. Xiao, Y.; Yang, L.; Zhou, Y.; Wei, Y.; Wang, N. Dominant parameters affecting the reliability of TBCs on a gas turbine blade during erosion by a particle-laden hot gas stream. *Wear* **2017**, *390–391*, 166–175. [[CrossRef](#)]
8. Wellman, R.; Nicholls, J. A review of the erosion of thermal barrier coatings. *J. Phys. D Appl. Phys.* **2007**, *40*, R293–R305. [[CrossRef](#)]
9. Davis, A.; Boone, D.; Levy, A. Erosion of ceramic thermal barrier coatings. *Wear* **1986**, *110*, 101–116. [[CrossRef](#)]
10. Bruce, R.W. Development of 1232 °C (2250 °F) erosion and impact tests for thermal barrier coatings. *Tribol. Trans.* **1998**, *41*, 399–410. [[CrossRef](#)]
11. Nicholls, J.R.; Jaslier, Y.; Rickerby, D.S. Erosion of EB-PVD thermal barrier coatings. *Mater. High Temp.* **1998**, *15*, 15–22. [[CrossRef](#)]
12. Eaton, H.E.; Zajchowski, P. High temperature particulate erosion of plasma sprayed YSZ versus selected powder characteristics and plasma torch designs. *Surf. Coat. Technol.* **1999**, *120–121*, 28–33. [[CrossRef](#)]
13. Nicholls, J.; Deakin, M.; Rickerby, D. A comparison between the erosion behaviour of thermal spray and electron beam physical vapour deposition thermal barrier coatings. *Wear* **1999**, *233–235*, 352–361. [[CrossRef](#)]
14. Nicholls, J.R.; Wellman, R.; Deakin, M.J. Erosion of thermal barrier coatings. *Mater. High Temp.* **2003**, *20*, 207–218. [[CrossRef](#)]

15. Wellman, R.; Nicholls, J.; Murphy, K. Effect of microstructure and temperature on the erosion rates and mechanisms of modified EB PVD TBCs. *Wear* **2009**, *267*, 1927–1934. [[CrossRef](#)]
16. Cernuschi, F.; Lorenzoni, L.; Capelli, S.; Guardamagna, C.; Karger, M.; Vaßen, R.; von Niessen, K.; Markocsan, N.; Menuey, J.; Giolli, C. Solid particle erosion of thermal spray and physical vapour deposition thermal barrier coatings. *Wear* **2011**, *271*, 2909–2918. [[CrossRef](#)]
17. Shin, D.; Hamed, A. Influence of micro-structure on erosion resistance of plasma sprayed 7YSZ thermal barrier coating under gas turbine operating conditions. *Wear* **2018**, *396–397*, 34–47. [[CrossRef](#)]
18. Behrens, J.T. Principles and procedures of exploratory data analysis. *Psychol. Methods* **1997**, *2*, 131–160. [[CrossRef](#)]
19. Howard, D.; Mark, B. Neural Network Toolbox Documentation. *Neural Netw. Tool* **2004**, 846.
20. Hagan, M.; Menhaj, M. Training feedforward networks with the Marquardt algorithm. *IEEE Trans. Neural Netw.* **1994**, *5*, 989–993. [[CrossRef](#)]
21. Winiczenko, R.; Górnicki, K.; Kaleta, A.; Janaszek-Mańkowska, M. Optimisation of ANN topology for predicting the rehydrated apple cubes colour change using RSM and GA. *Neural Comput. Appl.* **2018**, *30*, 1795–1809. [[CrossRef](#)]
22. Mackay, D.J.C. Bayesian Interpolation. *Neural Comput.* **1992**, *4*, 415–447. [[CrossRef](#)]
23. Qu, G.; Li, N. Accelerated Distributed Nesterov Gradient Descent. *IEEE Trans. Autom. Control* **2020**, *65*, 2566–2581. [[CrossRef](#)]
24. Friedman, J.H. Greedy function approximation: A gradient boosting machine. *Ann. Stat.* **2001**, *29*, 1189–1232. [[CrossRef](#)]
25. Molnar, C. *Interpretable Machine Learning. A Guide for Making Black Box Models Explainable*; Lulu: Morrisville, NC, USA, 2019.
26. Liaw, A.; Wiener, M. Classification and Regression by randomForest. *R News* **2002**, *2*, 18–22.
27. Ye, D.; Wang, W.; Zhou, H.; Fang, H.; Huang, J.; Li, Y.; Gong, H.; Li, Z. Characterization of thermal barrier coatings microstructural features using terahertz spectroscopy. *Surf. Coat. Technol.* **2020**, *394*, 125836. [[CrossRef](#)]
28. Yang, X.-S.; Deb, S. Cuckoo search via Lévy flights. In Proceedings of the 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), Coimbatore, India, 9–11 December 2009; pp. 210–214.
29. Vignesh, B.; Oliver, W.; Kumar, G.S.; Phani, P.S. Critical assessment of high speed nanoindentation mapping technique and data deconvolution on thermal barrier coatings. *Mater. Des.* **2019**, *181*, 108084. [[CrossRef](#)]
30. [Hartigan, J.A.; Wong, M.A. Algorithm AS 136: A K-Means Clustering Algorithm. *J. R. Stat. Soc. Ser. C Appl. Stat.* **1979**, *28*, 100–108. [[CrossRef](#)]
31. Ma, Z.; Zhang, W.; Luo, Z.; Sun, X.; Li, Z.; Lin, L. Ultrasonic characterization of thermal barrier coatings porosity through BP neural network optimizing Gaussian process regression algorithm. *Ultrasonics* **2020**, *100*, 105981. [[CrossRef](#)]
32. Ma, Z.; Zhao, Y.; Luo, Z.; Lin, L. Ultrasonic characterization of thermally grown oxide in thermal barrier coating by reflection coefficient amplitude spectrum. *Ultrasonics* **2014**, *54*, 1005–1009. [[CrossRef](#)]
33. Ma, Z.; Luo, Z.; Lin, L.; Krishnaswamy, S.; Lei, M. Quantitative characterization of the interfacial roughness and thickness of inhomogeneous coatings based on ultrasonic reflection coefficient phase spectrum. *NDT E Int.* **2019**, *102*, 16–25. [[CrossRef](#)]
34. Park, J.; Sung, W. FPGA based implementation of deep neural networks using on-chip memory only. In Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Shanghai, China, 20–25 March 2016; pp. 1011–1015. [[CrossRef](#)]
35. Weimer, D.; Scholz-Reiter, B.; Shpitalni, M. Design of deep convolutional neural network architectures for automated feature extraction in industrial inspection. *CIRP Ann. Manuf. Technol.* **2016**, *65*, 417–420. [[CrossRef](#)]
36. Ren, R.; Hung, T.; Tan, K.C. A Generic Deep-Learning-Based Approach for Automated Surface Inspection. *IEEE Trans. Cybern.* **2017**, *48*, 929–940. [[CrossRef](#)] [[PubMed](#)]
37. Azimi, S.M.; Britz, D.; Engstler, M.; Fritz, M.; Mücklich, F. Advanced steel microstructural classification by deep learning methods. *Sci. Rep.* **2018**, *8*, 1–14. [[CrossRef](#)]
38. Wu, H.; Fang, W.-Z.; Kang, Q.; Tao, W.-Q.; Qiao, R. Predicting Effective Diffusivity of Porous Media from Images by Deep Learning. *Sci. Rep.* **2019**, *9*, 1–12. [[CrossRef](#)] [[PubMed](#)]
39. Chen, W.-B.; Standfield, B.N.; Gao, S.; Lu, Y.; Wang, X.; Zimmerman, B. A Fully Automated Porosity Measure for Thermal Barrier Coating Images. *Int. J. Multimed. Data Eng. Manag.* **2018**, *9*, 40–58. [[CrossRef](#)]
40. Lu, Y.; Chen, W.-B.; Wang, X.; Ailsworth, Z.; Tsui, M.; Al-Ghaib, H.; Zimmerman, B. Deep Learning-Based Models for Porosity Measurement in Thermal Barrier Coating Images. *Int. J. Multimed. Data Eng. Manag.* **2020**, *11*, 20–35. [[CrossRef](#)]
41. Ghai, R.S.; Chen, K.; Baddour, N. Modelling thermal conductivity of porous thermal barrier coatings. *Coatings* **2019**, *9*, 101. [[CrossRef](#)]
42. Bolelli, G.; Righi, M.G.; Mughal, M.Z.; Moscatelli, R.; Ligabue, O.; Antolotti, N.; Sebastiani, M.; Lusvarghi, L.; Bemporad, E. Damage progression in thermal barrier coating systems during thermal cycling: A nano-mechanical assessment. *Mater. Des.* **2019**, *166*, 107615. [[CrossRef](#)]
43. Slowik, A.; Kwasnicka, H. Evolutionary algorithms and their applications to engineering problems. *Neural Comput. Appl.* **2020**, *32*, 12363–12379. [[CrossRef](#)]
44. Bouwmans, T.; Javed, S.; Sultana, M.; Jung, S.K. Deep neural network concepts for background subtraction: A systematic review and comparative evaluation. *Neural Netw.* **2019**, *117*, 8–66. [[CrossRef](#)] [[PubMed](#)]
45. Liu, Y.; Ravichandran, R. Application of Machine Learning in Research of Solid Particle Erosion of APS-TBC and EB-PVD TBC at Elevated Temperatures. NRC Contract Research Number: 955871. 2021.