sensors

MDPI

Article

Three-Dimensional Printing Quality Inspection Based on Transfer Learning with Convolutional Neural Networks

Cheng-Jung Yang [1], Wei-Kai Huang [2] and Keng-Pei Lin [2,*]

[1] Program in Interdisciplinary Studies, National Sun Yat-sen University, Kaohsiung 80424, Taiwan
[2] Department of Information Management, National Sun Yat-sen University, Kaohsiung 80424, Taiwan
* Correspondence: kplin@mis.nsysu.edu.tw

Abstract: Fused deposition modeling (FDM) is a form of additive manufacturing where three-dimensional (3D) models are created by depositing melted thermoplastic polymer filaments in layers. Although FDM is a mature process, defects can occur during printing. Therefore, an image-based quality inspection method for 3D-printed objects of varying geometries was developed in this study. Transfer learning with pretrained models, which were used as feature extractors, was combined with ensemble learning, and the resulting model combinations were used to inspect the quality of FDM-printed objects. Model combinations with VGG16 and VGG19 had the highest accuracy in most situations. Furthermore, the classification accuracies of these model combinations were not significantly affected by differences in color. In summary, the combination of transfer learning with ensemble learning is an effective method for inspecting the quality of 3D-printed objects. It reduces time and material wastage and improves 3D printing quality.

Keywords: fused deposition modeling; image analysis; quality inspection; transfer learning; ensemble learning

Citation: Yang, C.-J.; Huang, W.-K.; Lin, K.-P. Three-Dimensional Printing Quality Inspection Based on Transfer Learning with Convolutional Neural Networks. *Sensors* **2023**, *23*, 491. https://doi.org/10.3390/s23010491

Academic Editors: Kong Fah Tee and Bin Huang

Received: 4 December 2022
Revised: 26 December 2022
Accepted: 30 December 2022
Published: 2 January 2023

1. Introduction

The advent of Industry 4.0 has birthed a fresh pursuit for increasingly productive and cost-efficient manufacturing technologies, and three-dimensional (3D) printing has become a key technology for parts manufacturing. Fused deposition modeling (FDM) 3D printing is currently the most popular type of 3D printing in the consumer space, and it has found applications in many industries [1–4]. FDM can be used to quickly generate proofs of concept for geometrically complex products [5,6], and it is a valuable tool for versatile manufacturing because of its ability to use a wide variety of filament materials [7]. Furthermore, FDM requires little postprocessing and has short processing times [8–11]. However, it takes many hours to produce large parts via FDM printing, and it is possible for various defects to form during this process, which may degrade the final product or cause it to fail. This increases the time, material, and effort required. Therefore, it is necessary to monitor printing quality, and visual inspection is by far the best way to obtain timely feedback. Hence, many technological innovations for this purpose have been based on visual inspection.

There are numerous examples in the literature where machine learning and deep learning have been successfully used to detect warping and delamination, measure surface roughness, monitor the 3D printing process, and predict printing quality. For example, supervised learning algorithms such as the naive Bayes classifier [12–14], k-nearest neighbors [13], random forest [13], decision tree [13,14], and support vector machine [13,15,16] have been used to train models to predict and detect defects. Accuracy comparisons have been performed among various convolutional neural networks (CNNs) [13,17–19]. Furthermore, CNNs have been combined with various machine learning methods to evaluate the accuracy of defect-detection models [20–22]. The CNNs have also been utilized in image

quality assessments [23]. Zhou et al. proposed to utilize a dual-stream convolutional neural network that simulates dual views of the human visual system to predict the perceptual quality of stereoscopic images [23].

Over time, numerous CNNs have been developed to increase the classification accuracy. Simonyan and Zisserman were the first to propose the use of very deep CNNs to increase the fineness of the extracted features [24], which culminated in the emergence of the Visual Geometry Group (VGG) series of CNNs for image recognition and classification [25–28]. Szegedy et al. proposed the Inception architecture, which increases the computational efficiency and facilitates increases in the network depth and width [29]. Szegedy et al. subsequently added batch normalization layers to regularize intermediate features, which significantly accelerate learning and convergence, and they reduced the size of the feature map by using factorization with parallel pooling and convolution [30]. Since the gradients may vanish or explode in very deep networks, and degradation can occur if the depth of an optimal shallow network is increased, He et al. proposed the learning of residuals to address this problem. By adding residual connections (skip connections) to the outputs of a CNN, the accuracy could be improved in very deep networks. This led to the creation of the ResNet series [31,32]. ResNet has been used for fault diagnosis in industrial manufacturing [33,34], rolling bearings [35], and rotating machinery [36]. The increases in network depth and width facilitated by the ResNet architecture led to tremendous improvements in CNN performance. ResNet has also been utilized as a pretrained model for image quality assessment. Sun et al. utilized six ResNet34s to extract features from six views of 360-degree images and combined them with a regressor for image quality assessment of VR devices [37].

In 2019, Tan et al. used the neural architecture search (NAS) technique in Google AutoML to design a baseline network, and they scaled the width, depth, and resolution of this network using the compound coefficient to create a family of models called EfficientNet [38]. In 2021, Tan et al. published the v2 versions of the EfficientNet models, which are smaller and faster to train than their predecessors [39]. Zhou et al. used EfficientNetV2 for fine-tuning the EfficientNet-B7 pretrained model to create a machine that extracts features from whole-body portraits and generates Lego brick models [40]. This approach was also used for dog nose print matching [41] and prediction of the energy consumption of 3D printing processes [42].

Kim et al. adopted VGG19 pretrained for transfer learning to detect the filament tangling in the 3D printing process [28]. Baumgartl et al. used thermographic imaging data to train a CNN to detect defects in laser powder bed fusion in metal 3D printing [18]. Banadaki et al. used Inception-v3, which is also based on a convolutional neural network, for real-time surface defect detection and grading during FDM printing [43]. Jin et al. proposed the use of Inception-v3 as a transfer learning model to classify the extent of delamination and predict warping [33]. Razaviarab et al. proposed the use of transfer learning in combination with a closed-loop machine learning algorithm to automatically detect defects in 3D printing [44].

The work of Kadam et al. [13] is most relevant to our work, which studied the fault detection of FDM by combining the pretrained models (AlexNet, GoogLeNet, ResNet18, ResNet50, EfficientNet-B0) with popular classification methods including KNN, SVM, naïve Bayes classifier, etc. Their results showed that the SVM achieved the best accuracy in four of the five combined pretrained models, and the combination of SVM with AlexNet resulted in the best accuracy.

Combining CNN with different machine learning methods usually results in good performance for fault detection and classification. Although ensemble learning is an important part of machine learning, the combination of transfer learning and ensemble learning for surface defect detection in 3D printing, which would reduce time and material wastage, has yet to be investigated. Furthermore, although color selection is critical for FDM-printed objects, there are no reports about the effects of color on the accuracy of defect-detection algorithms. Therefore, CNN-based transfer learning was combined with

six ensemble learning models to detect and classify surface defects in FDM-printed 3D objects. The effects of color on the accuracies of these model combinations were analyzed. The findings of this study reveal the model combinations that are most accurate for each color and geometry.

There have been reports in bearing fault diagnosis that using CNN for feature extraction with the boosting method LightGBM for classificiation can result in better accuracy than using only CNN [45]. The transfer learning with CNN-based pretrained models on large-scale image datasets such as GoogleNet and AlexNet has been successfully utilized on 3D printing image classification [13]. In this work, we further combine the transfer learning with bagging and boosting approaches to make an ensemble approach.

The neural network models usually have high variances, and in 3D printing image classification, the dataset is usually rather small, which may increase the variance of prediction. Therefore, we capitalize on the ensemble approach, the bagging and boosting, to improve the performance on training classifiers from the features extracted by the neural networks-based pretrained models.

The bagging algorithm trains a group of classifiers on different subsamples of the dataset to make an ensemble classifier. Bagging can help improve the stability of prediction, especially for unstable approaches such as neural networks. It can also reduce the variances and prevent overfitting, benefitting the 3D printing image classification of small samples, since collecting the samples of 3D printing is very time-consuming, and small samples are prone to overfitting. The boosting algorithm trains a series of weak learners from the extracted features and combine to an ensemble classifier, which is expected to reduce the variances and biases in the classifier, and therefore improve the classification performance.

In addition to combining the existing pretrained models with the ensemble approach, we also study the application of the newly developed pretrained models for transfer learning in 3D printing classification. The newer pretrained models, either featured with deeper layers or different network structure, have been continuously developed to improve the classification performance. The experimental results show that the ensemble approaches, either bagging or boosting, can effectively improve the classification performance. Therefore, by utilizing the transfer learning with newly developed pretrained models and combining them with the ensemble approach, we can achieve a better classification performance in 3D-printed image classification.

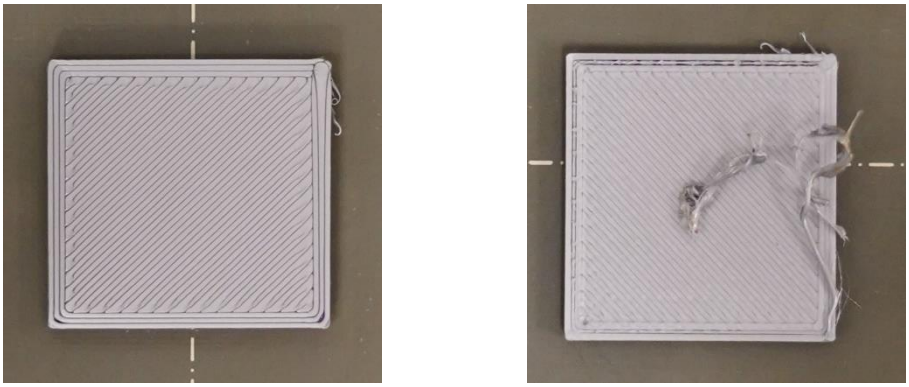
The remainder of this paper is organized as follows. Section 2 introduces the materials and algorithms used in this study. Section 3 presents the experimental data and the findings of the data analysis. Section 4 discusses how the printing geometry and filament color are related to the accuracy of each algorithm, as well as the contributions of these factors to algorithm accuracy. Finally, the conclusions and outlook are presented in Section 5.

2. Materials and Methods

All the 3D printing was performed using a Prusa i3 mk3s 3D printer (Prusa, Prague, Czech Republic) [46]. with a polylactic acid filament [47]. All the photographs were captured using a Sony a7 III camera (Sony, New York, NY, USA) [48]. Image preprocessing and model training and testing were performed on the Google Colab platform [49].

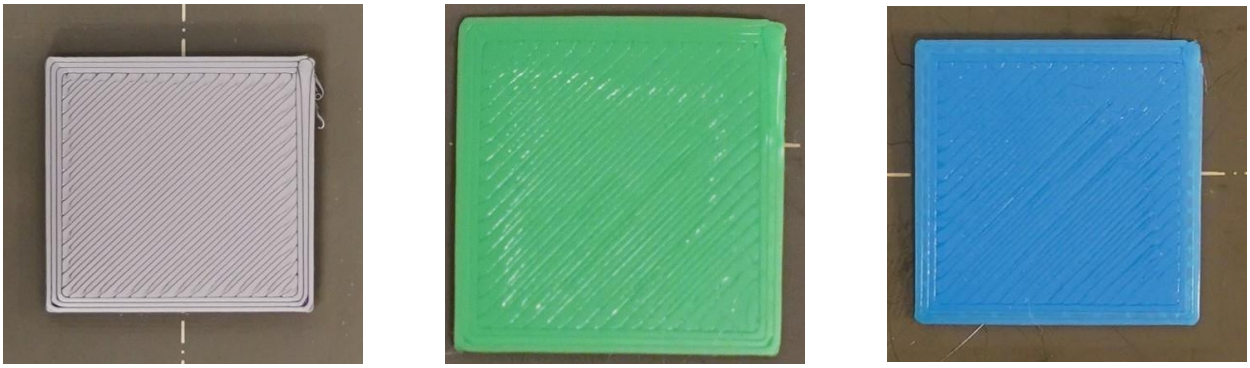
2.1. Classification Principles

To construct an image dataset of defective and nondefective samples based on the classification principles shown in Figure 1, the images were manually labeled one by one. Nondefective samples had smooth and fully filled surfaces; all other samples were classified as defective. The filament colors were gray, green, and blue. Figure 2 shows the pictures of the finished layer. The collected dataset contains the pictures of every printing layer.



(**a**) (**b**)

Figure 1. Photographs of (**a**) nondefective and (**b**) defective samples.



(**a**) (**b**) (**c**)

Figure 2. Photographs of (**a**) gray, (**b**) green, and (**c**) blue samples.

2.2. Experimental Procedures

The experimental procedures of this study are shown in Figure 3. First, the 3D printer was configured, and a camera was set up to capture photographs of the 3D-printed object. In Step 2, the collected photographs were cropped and classified to create the image dataset. In Step 3, the image dataset was divided into training and testing sets at a 7:3 ratio using split_train_test. In Step 4, the Python Open Source Computer Vision Library was used to extract the red, green, and blue color model (RGB); GRAY; and hue, saturation, and value color model (HSV) values of each image, and the angle of each image was varied to produce images with different angles. In Step 5, a variety of CNNs were used for feature extraction from the images. In Step 6, a variety of ensemble learning algorithms were used to train defect-detection models; the accuracy of each model was then evaluated. Finally, the model accuracies were analyzed. The flow graph of the proposed method is shown in Figure 4.

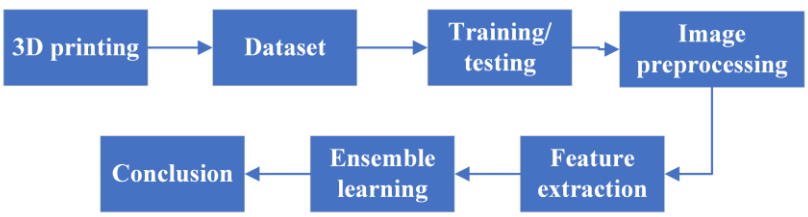


Figure 3. Experimental procedure.

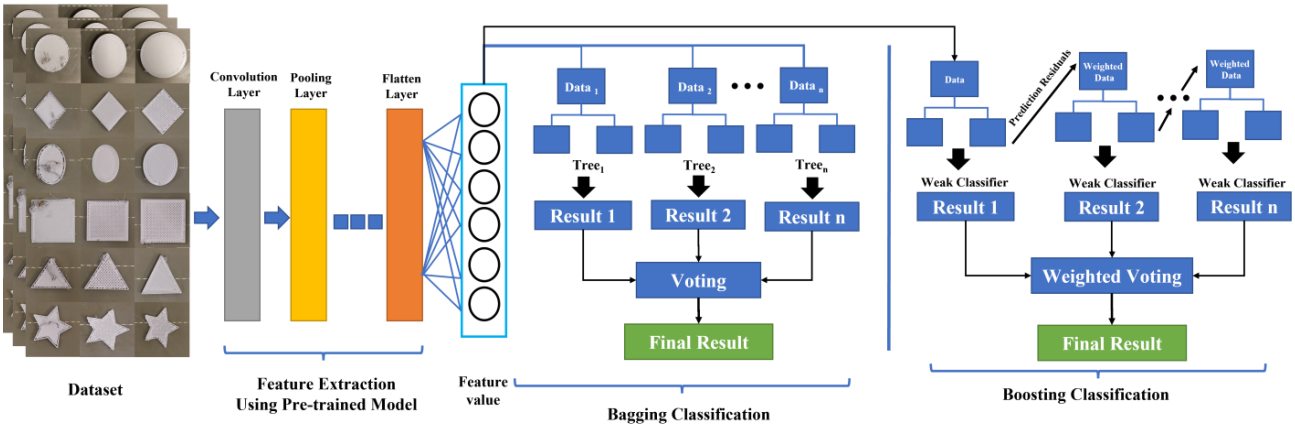


Figure 4. The flow graph of the proposed method for bagging or boosting classification.

2.3. CNN

The CNNs used in this study were only used as pretrained models for feature extraction from the image dataset. Therefore, a variety of ensemble learning algorithms were used for classification, training, and testing to increase the accuracy. The CNNs used in this study included VGG16, VGG19, InceptionV3, ResNet50, EfficientNetB0, and EfficientNetV2L. A brief overview of these models is presented below.

2.3.1. VGG16

VGG16 is a 16-layer CNN with 13 convolutional layers and 3 fully connected layers [29]. As this network architecture has a large number of weights and fully connected nodes, its parameter space is large, which results in long training times. The size of the VGG16 model is 528 MB, and its input images are 224 × 224 RGB images. The architecture of VGG16 is shown in Figure 5.

Figure 5. VGG16 architecture.

2.3.2. VGG19

VGG19 is a 19-layer CNN with 16 convolutional layers and 3 fully connected layers (three more convolutional layers than VGG16) [29]. Similar to VGG16, VGG19 has a large number of weights and fully connected nodes. Its size is 549 MB, and its inputs are also 224 × 224 RGB images. The VGG19 architecture is shown in Figure 6.

Figure 6. VGG19 architecture.

2.3.3. InceptionV3

InceptionV3 is the third generation of the Inception architecture. It consists of several block modules, which have a global average pooling layer instead of a fully connected

layer at the end. These block modules give InceptionV3 a total of 47 layers [30]. In contrast to ResNet, InceptionV3 avoids representational bottlenecks early in the network to prevent losses of feature information. As the InceptionV3 model can increase the width and depth while maintaining computational efficiency, its size is only 92 MB. Its inputs are 299 × 299 RGB images. The InceptionV3 architecture is shown in Figure 7.

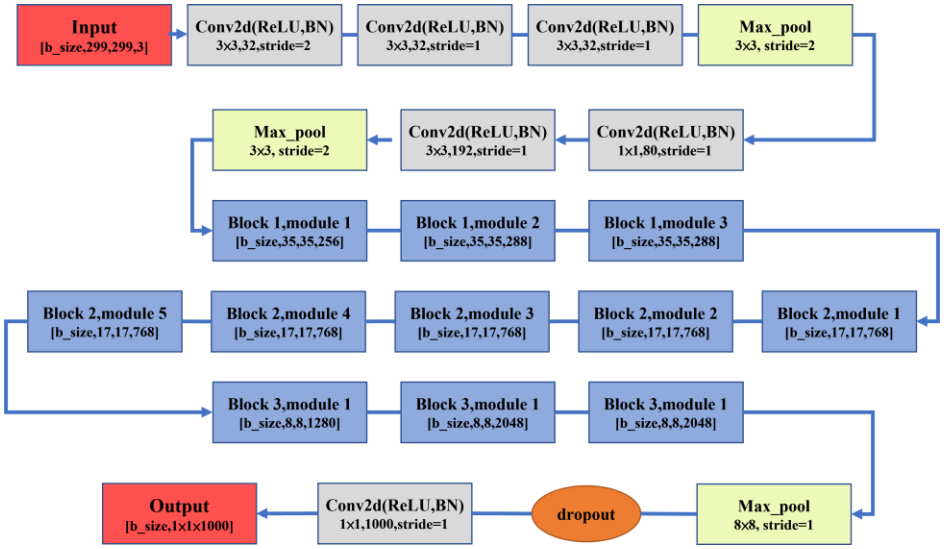


Figure 7. InceptionV3 architecture.

2.3.4. ResNet50

ResNet50 is a part of the ResNet family of models, and as its name suggests, it has 50 layers, which consist of 49 convolutional layers and 1 fully connected layer [31]. As the model uses residual connections instead of fully connected layers and consists of residual blocks, its size is only 98 MB. The inputs of ResNet50 are 224 × 224 RGB images. The ResNet50 architecture is shown in Figure 8.

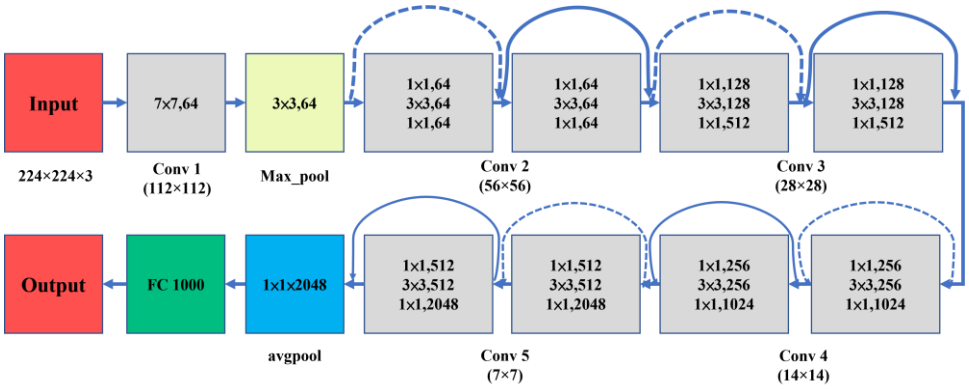


Figure 8. ResNet50 architecture.

2.3.5. EfficientNetB0

EfficientNetB0 is a model from the EfficientNet series. It consists of 5 large modules with different numbers of submodules, giving it a total of 237 layers [38]. The MBConv6 modules that constitute EfficientNetB0 are depthwise separable convolutions from the MobileNet architecture with ResNet-like residual connections, and there is an expansion layer that increases the number of channels by a factor of 6 [50]. Furthermore, NAS was used to determine the depth, width, and number of channels of EfficientNetB0. B0 is the smallest model in the EfficientNet series, and its size is only 29 MB. The inputs of this model are 224 × 224 RGB images. The EfficientNetB0 architecture is shown in Figure 9.

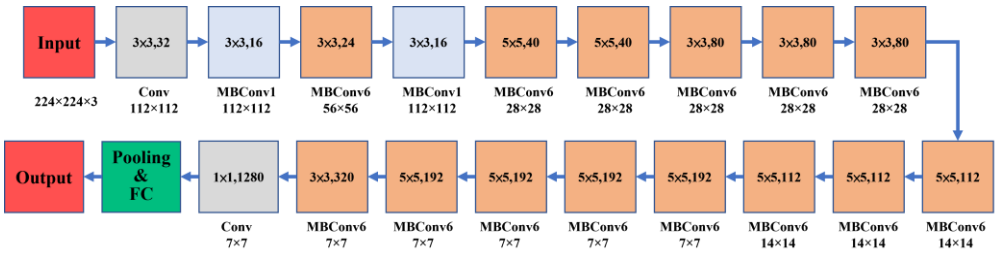


Figure 9. EfficientNetB0 architecture.

2.3.6. EfficientNetV2L

EfficientNetV2L comes from the EfficientNetV2 family of models. To increase the training speed and parameter efficiency, the V2 edition of EfficientNet uses training-aware NAS and scaling and Fused-MBConv modules in shallow networks. EfficientNetV2 also uses an improved progressive learning method, where the regularization strength (dropout rate, RandAugment magnitude, and mixup ratio) is adjusted according to the image size, which increases the accuracy and training speed [39]. As the V2L model contains a large number of layers, its size is 479 MB. Its inputs are 480 × 480 RGB images. Details regarding the architecture of EfficientNetV2L are presented in Table 1.

Table 1. EfficientNetV2-L architecture.

Stage	Operator	Stride	Channels	Layers
0	Conv3 × 3	2	32	1
1	Fused-MBConv1, k3 × 3	1	32	4
2	Fused-MBConv4, k3 × 3	2	64	7
3	Fused-MBConv4, k3 × 3	2	96	7
4	MBConv4, k3 × 3, SE0.25	2	192	10
5	MBConv6, k3 × 3, SE0.25	1	224	19
6	MBConv6, k3 × 3, SE0.25	2	384	25
7	MBConv6, k3 × 3, SE0.25	1	640	7
8	Conv1 × 1 & Pooling & FC	-	1280	1

2.4. Ensemble Learning Algorithms

2.4.1. Bagging

The idea of bagging is to randomly sample the training set to train multiple independent classifiers with normalized weights, which then vote on the final result [51–53]. Random forest (RF) bagging was used in this study, which is a supervised algorithm. It is an advanced version of the decision tree architecture. An RF consists of multiple decision trees, and besides selecting random samples from the training set (bagging), random subsets of features are drawn to train each tree. Although overfitting tends to occur when a decision tree becomes too deep, the RF architecture resists overfitting by having multiple decision trees, which allows RF to work accurately and efficiently on large high-dimensional datasets [51,52].

2.4.2. Boosting

The idea of boosting is to combine multiple weak classifiers into a single strong classifier. Boosting is an iterative approach, where input data that were misclassified by the older classifier are given a higher weight when training a new classifier. This allows the new classifier to learn features, and thus increases the accuracy. Finally, the iteratively trained weak classifiers vote (with weights) to produce the final result [53–55]. The boosting models used in this method were AdaBoost, GBDT, XGBoost, LightGBM, and CatBoost.

1. AdaBoost The idea of AdaBoost is to create a strong classifier by summing weighted predictions from a set of weak classifiers. AdaBoost, which is short for adaptive boosting, uses the misclassified samples of the preceding classifiers to train the next

generation of classifiers. This is an iterative approach where weighted training data are used instead of random training samples, so that the classifier can focus on hard-to-classify training data. A new classifier is added at each iteration, until the error falls below a threshold. As the model is effectively a strong classifier, it is robust against overfitting. However, noisy data and outliers should be avoided to the greatest extent possible [56,57].

2. Gradient-Boosting Decision Tree (GBDT) The GDBT is a multiple-additive regression tree and is a technique where a strong classifier is formed by combining many weak classifiers. The GDBT model is applicable to both classification and regression problems. Every prediction differs from the actual value by a residual; in GDBT, the log-likelihood loss function is used to maximize the probability that the predicted value is the real value. To prevent overfitting, the residual and predicted residual are calculated, and the predicted residual is multiplied by the learning rate. New trees are generated one after another to correct the residual until it approaches 0, that is, until the prediction approaches the true value [58–61].
3. Extreme Gradient Boosting (XGBoost) XGBoost is a method where additive training is combined with gradient boosting. In each iteration, the original model is left unchanged, and a new function is added to correct the error of the previous tree. The risk of overfitting is minimized through regularization and the addition of a penalty term Ω to the loss function. XGBoost combines the advantages of bagging and boosting, as it allows the trees to remain correlated with each other while utilizing random feature sampling. In contrast to other machine learning methods that cannot handle sparse data, XGBoost can efficiently handle sparse data through sparsity-aware split finding. In this method, the gains obtained from adding sparse data to the left and right sides of a tree are calculated, and the side that gives the highest gain is selected [61–64].
4. LightGBM LightGBM is a type of GDBT that uses histogram-based decision trees, which traverse the dataset and select optimal splitting points based on discrete values in a histogram. This reduces the complexity of tree node splitting and makes LightGBM very memory- and time-efficient. LightGBM uses gradient-based one-side sampling to retain training instances with large gradients, as well as exclusive feature bundling to reduce the dimensionality [61,65–67].
5. CatBoost CatBoost is another GDBT-based model. To create unbiased predictions, CatBoost uses ordered boosting to reduce the degree of overfitting and uses oblivious trees as base predictors. In many competitions hosted by Kaggle, CatBoost achieved the highest accuracies and smallest log-loss values [61,66–70].

3. Experiments

The gray dataset contains 1275 samples, where 575 are nondefective and 700 are defective. The green dataset contains 1464 samples, where 735 are nondefective and 729 are defective. The blue dataset contains 1274 samples, where 644 are nondefective and 630 are defective. Detailed statistics for each shape are shown in Table 2. The printing temperature of extruder is 210 °C and bed is 60 °C, and the fill density is 20%. The retraction speed is 35 mm/s, and printing speed is 20 mm/s. The layer thickness is 0.1 mm, and there are two top solid layers.

3.1. Effects of Geometric Differences

To present all the pretrained model + ensemble learning combinations and their accuracies, we provide a visual representation of their accuracies on gray-colored geometries in Figure 10. We also show the accuracy of using AlexNet with SVM (raw data are presented in Appendix D), which was the best combination reported in [13], as the baseline for comparison. The performance of the baseline is shown as a dotted regular hexagon in each color–shape combination.

Table 2. Statistics of the dataset.

	Gray Nondefective	Gray Defective	Green Nondefective	Green Defective	Blue Nondefective	Blue Defective
Square	90	104	144	114	117	83
Star	124	134	97	119	76	90
Circle	112	98	156	126	112	127
Oval	71	122	107	131	124	107
Diamond	62	106	134	133	108	105
Triangle	116	136	97	106	107	118

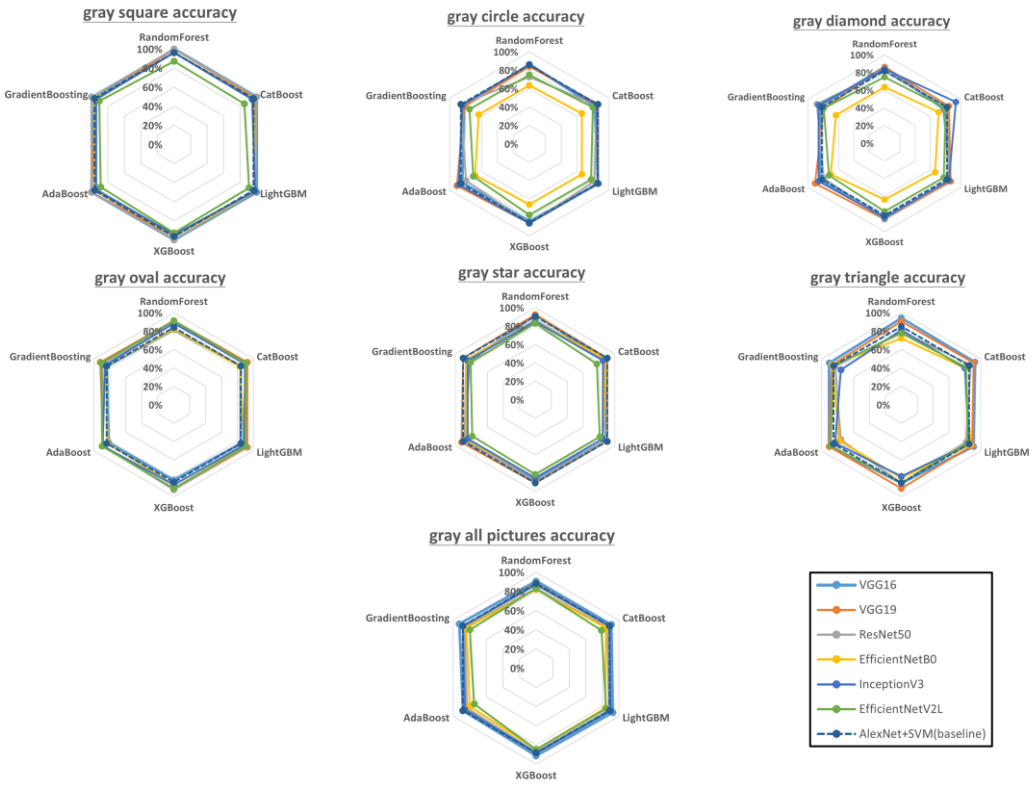


Figure 10. Accuracy of each model combination for gray-colored geometries (raw data are presented in Appendix A). The dotted regular hexagon is the baseline using AlexNet with SVM.

Here, we summarize the results for each gray-colored geometry. Gray squares: For this dataset, VGG16, VGG19, and ResNet50 consistently obtained high accuracies with all the ensemble learning models; EfficientNetV2L, by contrast, performed poorly with all the ensemble learning models. Gray circles: The highest accuracy was achieved by VGG19 + AdaBoost (90.91%), followed by InceptionV3 + AdaBoost (88.64%), whereas EfficientNetB0 had the lowest accuracy with all the ensemble learning models. Gray diamonds: InceptionV3 + CatBoost had the highest accuracy (93.18%), followed by VGG19 + AdaBoost (90.91%). Again, EfficientNetB0 had the lowest accuracy, regardless of which ensemble learning model it was paired with. Gray ovals: The VGG16 + Catboost, VGG16 + Xgboost, VGG16 + GradientBoosting, and ResNet50 + XGboost pairings were tied for the highest accuracy (92.37%), followed by VGG19 + RandomForest and EfficientNetV2L + RandomForest (91.60%). Gray stars: VGG16 + RandomForest, VGG19 + RandomForest, and VGG19 + AdaBoost were tied for the highest accuracy (92.45%); the lowest accuracies were obtained with EfficientNetV2L + any ensemble learning model. Gray triangles: VGG16 + RandomForest achieved the highest accuracy (94.44%), followed by VGG16 + Catboost (92.59%). All

gray pictures: VGG16 + LightGBM and VGG16 + GradientBoosting achieved the highest overall accuracy (92.44%), followed by VGG16 + XGboost (91.41%).

3.2. Effects of Color

Next, defect classification was performed on the green and blue geometries. Figure 11 shows the accuracy of each model combination for green geometries. Green squares: VGG16 + ResNet50 and VGG19 + ResNet50 were both highly accurate, but in contrast to the case of gray squares, InceptionV3 + RandomForest and EfficientNetV2L + AdaBoost also achieved the highest level of accuracy. Green circles: VGG19 + InceptionV3 and VGG19 + RandomForest were the most accurate combinations (93.94%), followed by VGG19 + LightGBM (92.42%). EfficientNetV2L consistently exhibited the lowest accuracy, regardless of the ensemble learning model. Green diamonds: EfficientNetB0 + LightGBM was the most accurate combination (92.59%), followed by EfficientNetB0 + GradientBoosting (90.74%). Green ovals: VGG19 + CatBoost, VGG19 + InceptionV3, and VGG19 + XGboost were tied for the highest accuracy (98.31%). EfficientNetB0 and EfficientNetV2L were consistently the least accurate models, with all the ensemble learning models. Green stars: ResNet50 + AdaBoost, ResNet50 + GradientBoosting, and EfficientNetB0 + GradientBoosting were tied for the highest accuracy (95.83%), whereas EfficientNetV2L had the lowest accuracy with all the ensemble learning models. Green triangles: VGG16 + LightGBM, VGG16 + AdaBoost, and InceptionV3 + AdaBoost were tied for the highest accuracy (94%), followed by VGG16 + XGboost and VGG16 + GradientBoosting (92%). All green geometries: VGG19 + XGboost and InceptionV3 + Catboost were tied for the highest accuracy (92.33%), followed by VGG16 + RandomForest and VGG16 + LightGBM (92.02%).

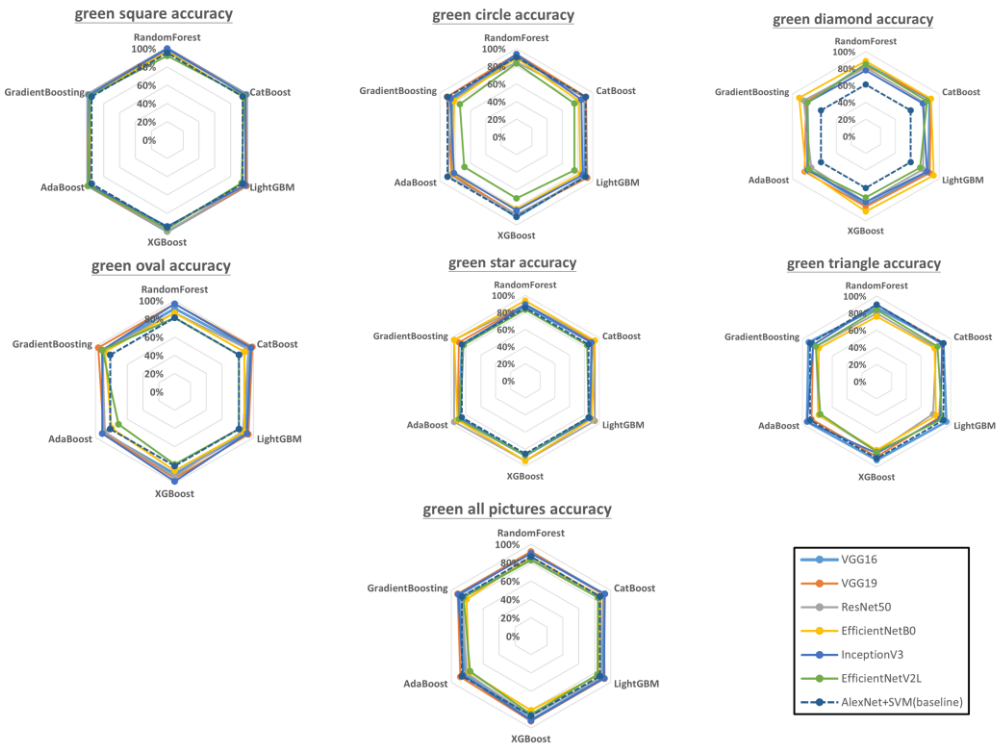


Figure 11. Accuracy of each model combination for green-colored geometries (raw data are presented in Appendix B). The dotted regular hexagon is the baseline using AlexNet with SVM.

Figure 12 shows the accuracy of each model combination for blue geometries. Blue squares: VGG19 + Catboost and VGG19 + AdaBoost had the highest accuracy (97.96%), whereas EfficientNetV2L had the lowest accuracy when combined with ensemble learning. Blue circles: VGG16 + GradientBoosting had the highest accuracy (98.31%), followed by VGG16 + LightGBM, VGG16 + XGboost, and InceptionV3 + Catboost, which were

tied for second place (96.61%). Blue diamonds: VGG16 + RandomForest had the highest accuracy (96.15%), followed by VGG16 + LightGBM and VGG16 + XGboost (both 94.23%). EfficientNetV2L had the lowest accuracies with ensemble learning. Blue ovals: VGG19 + Catboost had the highest accuracy (98.18%), whereas EfficientNetB0 had the lowest accuracies with ensemble learning. Blue stars: InceptionV3 + Catboost had the highest accuracy (96.36%). Blue triangles: EfficientNetB0 + Catboost had the highest accuracy (96.23%), followed by EfficientNetB0 + GradientBoosting and ResNet50 + RandomForest, which both had an accuracy of 90.57%. All blue geometries: VGG16 + RandomForest had the highest overall accuracy (93.73%), followed by VGG16 + LightGBM (92.74%). EfficientNetV2L had the lowest accuracies with ensemble learning.

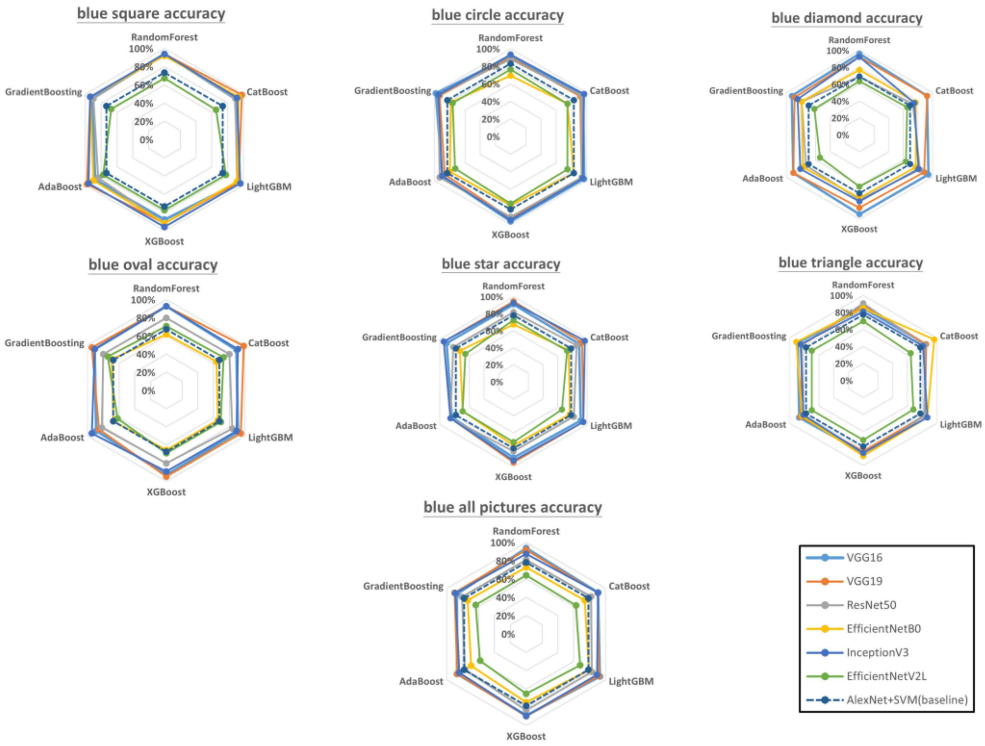


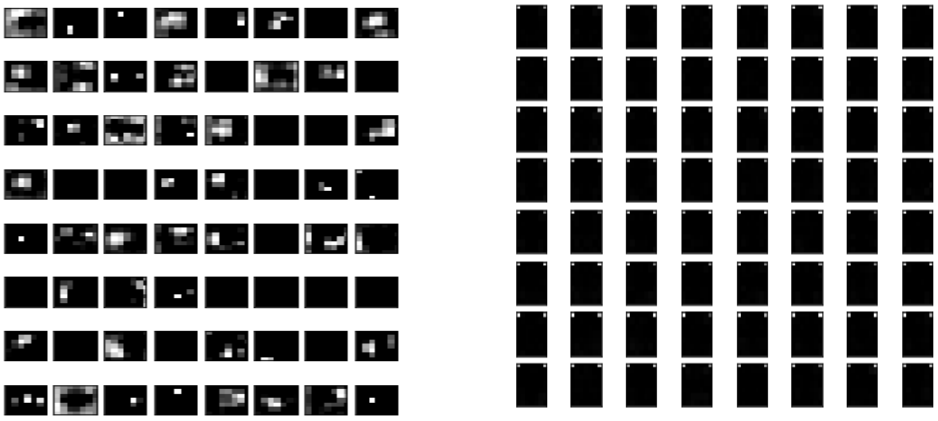
Figure 12. Accuracy of each model combination for blue-colored geometries (raw data are presented in Appendix C). The dotted regular hexagon is the baseline using AlexNet with SVM.

4. Discussion

According to the literature, bagging ensembles are outperformed by boosting ensembles in most scenarios [71]. However, our experimental results indicated that this does not hold true for surface defect detection in 3D-printed geometries. As shown in Figures 10–12, bagging ensembles yielded the highest accuracies. This may be because bagging reduces variance instead of bias, which can prevent overfitting. Furthermore, in contrast to bagging ensembles, the trees in boosting ensembles are correlated with each other. This makes it possible to form incorrect correlations, leading to worse performance compared with bagging.

The performance of a CNN can be significantly improved by increasing its depth or using novel structures, e.g., by using residual learning, factorization, and modules instead of layers, which increases the accuracy [72,73]. However, the results in Figures 10–12 indicate that this is not always true. Although deeper networks and novel CNN structures allow for the extraction of finer details, if the details are too fine, the network can be misled. Correct predictions may then be misjudged as being incorrect and vice versa. This phenomenon was apparent in the visualized feature maps of the last layer in the VGG16 and EfficientNetV2L networks (Figure 13), where VGG 16 features 16 layers in the network and EfficientNetV2L features 1028 layers. The last layer of the VGG16 network always

contained a well-defined feature, in contrast to EfficientNetV2L, which did not contain a clear feature in its last layer because of its excessive depth, causing the degradation in classification accuracy.



(**a**) (**b**)

Figure 13. Visualized feature maps of the last layer in the (**a**) VGG16 and (**b**) EfficientNetV2L networks.

In theory, different filament colors result in different levels of hue saturation and brightness, which can yield significant differences in accuracy. Studies on the correlation between color data and image classification have revealed that the classification accuracy tends to be higher with bright colors [74]. However, the results of this study indicated that the classification accuracy does not differ significantly among the gray-, green-, and blue-colored geometries. This is because the surface defect (roughness) detection is only affected by the surface smoothness and voids (whether the surface is fully filled); as long as the pictures are sufficiently clear, the classification accuracy does not vary significantly with respect to the color.

In industrial manufacturing, ensemble learning can be applied to various sensor data to accurately diagnose and predict faults. Studies [75–80] have indicated that ensemble learning can be used to predict and evaluate the performance of industrial machinery and detect faults. In the present study, the mean accuracy of surface quality classification was >90%, and an accuracy of 100% was achieved in some cases. Therefore, it is feasible to evaluate the quality of FDM-printed products by using image recognition technology in conjunction with a CNN.

5. Conclusions

Transfer learning with pretrained models was combined with ensemble learning to classify the quality of 3D-printed objects. The objective was to identify the combination of algorithms that yields the highest classification accuracy with variations in object geometry and color. The following conclusions are drawn.

1. The surface quality of FDM 3D-printed objects can be accurately classified by combining transfer learning with ensemble learning.
2. The combination of VGG16 or VGG19 with ensemble learning gave the highest accuracy for gray-colored geometries. Although model combinations with EfficientNetB0 and EfficientNetV2L exhibited the highest accuracy in a few instances, these models were relatively inaccurate in most situations.
3. Although boosting ensembles usually outperform bagging ensembles, in this case (quality inspection of 3D-printed objects), the combination of a transfer learning model with a bagging ensemble often resulted in better accuracy. Therefore, it was unable to prove that boosting is superior to bagging (or vice versa) in this study.

4. Although deeper networks with novel structures often achieve better CNN performance (and a higher classification accuracy), this rule does not apply to quality inspections for FDM-printed objects.
5. In this study, the highest classification accuracy of the model combinations did not vary significantly with respect to the color and geometry. Therefore, the filament color does not significantly affect the classification accuracy.

In our future work, we will develop real-time solutions to monitor 3D printing quality and detect printing failures by combining machine learning with a camera module. In addition, we will continue to incorporate the latest machine learning techniques for increasing the overall classification accuracy, to automate visual anomaly detection and eliminate the time and financial costs associated with manual inspections.

Author Contributions: Conceptualization, C.-J.Y. and K.-P.L.; methodology, W.-K.H.; software, W.-K.H.; validation, C.-J.Y., W.-K.H. and K.-P.L.; formal analysis, W.-K.H.; investigation, C.-J.Y. and W.-K.H.; resources, C.-J.Y.; data curation, W.-K.H.; writing—original draft preparation, C.-J.Y. and W.-K.H.; writing—review and editing, C.-J.Y. and K.-P.L.; visualization, C.-J.Y.; supervision, C.-J.Y. and K.-P.L.; project administration, C.-J.Y.; funding acquisition, C.-J.Y. and K.-P.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the National Science and Technology Council, Taiwan, under Grant NSTC 111-2621-M-110-001 and 110-2410-H-110-030-MY3.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Table A1. Accuracy of each model combination for gray squares.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	100%	100%	100%	96.30%	96.30%	87.04%
Catboost	100%	98.15%	100%	96.30%	94.44%	85.19%
LightGBM	100%	98.15%	98.15%	98.15%	98.10%	90.74%
XGboost	100%	96.30%	100%	96.30%	94.44%	92.59%
AdaBoost	100%	100%	100%	96.30%	94.44%	88.89%
GradientBoosting	96.3%	94.44%	100%	96.30%	94.44%	90.74%

Table A2. Accuracy of each model combination for gray circles.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	86.36%	84.09%	72.73%	63.64%	86.36%	75.00%
Catboost	81.82%	86.36%	84.09%	65.91%	84.09%	79.55%
LightGBM	86.36%	86.36%	81.82%	65.91%	86.40%	77.27%
XGboost	84.09%	86.36%	86.36%	65.91%	86.36%	77.27%
AdaBoost	84.09%	90.91%	79.55%	68.18%	88.64%	70.45%
GradientBoosting	81.82%	81.82%	86.36%	63.64%	84.09%	75.00%

Table A3. Accuracy of each model combination for gray diamonds.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	86.36%	86.36%	81.82%	63.64%	84.09%	75.00%
Catboost	77.27%	84.09%	81.82%	70.45%	93.18%	79.55%
LightGBM	86.36%	86.36%	84.09%	65.91%	84.10%	77.27%
XGboost	84.09%	86.36%	86.36%	63.64%	84.09%	77.27%
AdaBoost	84.09%	90.91%	79.55%	70.45%	86.36%	72.73%
GradientBoosting	84.09%	79.55%	88.64%	63.64%	86.36%	79.55%

Table A4. Accuracy of each model combination for gray ovals.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	81.82%	91.60%	89.31%	82.44%	88.55%	91.60%
Catboost	84.09%	92.37%	88.55%	83.21%	90.08%	90.84%
LightGBM	84.09%	92.37%	90.84%	88.55%	87.80%	90.08%
XGboost	81.82%	92.37%	92.37%	87.02%	87.02%	90.84%
AdaBoost	81.82%	88.55%	89.31%	83.21%	89.31%	90.08%
GradientBoosting	84.09%	92.37%	89.31%	84.73%	86.26%	90.84%

Table A5. Accuracy of each model combination for gray stars.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	92.45%	92.45%	86.79%	84.91%	84.91%	83.02%
Catboost	86.79%	90.57%	86.79%	88.68%	84.91%	77.36%
LightGBM	88.68%	90.57%	88.68%	84.91%	84.90%	81.13%
XGboost	86.79%	90.57%	88.68%	84.91%	84.91%	81.13%
AdaBoost	88.68%	92.45%	88.68%	86.79%	84.91%	79.25%
GradientBoosting	83.02%	90.57%	86.79%	86.79%	84.91%	81.13%

Table A6. Accuracy of each model combination for gray triangles.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	94.44%	90.74%	79.63%	72.22%	81.48%	77.78%
Catboost	92.59%	90.74%	87.04%	79.63%	79.63%	81.48%
LightGBM	90.74%	88.89%	81.48%	87.04%	85.20%	83.33%
XGboost	85.19%	90.74%	85.19%	79.63%	77.78%	85.19%
AdaBoost	90.74%	90.74%	75.93%	77.78%	83.33%	88.89%
GradientBoosting	90.74%	87.04%	83.33%	87.04%	75.93%	85.19%

Table A7. Accuracy of each model combination for all gray-colored geometries.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	90.72%	88.66%	83.16%	82.16%	86.94%	83.16%
Catboost	90.72%	88.32%	85.91%	84.19%	88.32%	78.69%
LightGBM	92.44%	89.00%	87.63%	83.51%	89.30%	84.44%
XGboost	91.41%	88.66%	87.29%	84.88%	88.32%	84.79%
AdaBoost	87.97%	84.88%	83.16%	79.73%	85.91%	74.57%
GradientBoosting	92.44%	87.63%	86.25%	84.19%	88.32%	79.73%

Appendix B

Table A8. Accuracy of each model combination for green squares.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	100%	100%	98.44%	96.88%	100%	92.19%
Catboost	100%	100%	100%	95.31%	98.44%	96.88%
LightGBM	100%	100%	98.15%	95.31%	98.40%	93.75%
XGboost	100%	100%	100%	95.31%	94.44%	96.88%
AdaBoost	100%	100%	98.44%	96.88%	98.44%	100%
GradientBoosting	100%	100%	100%	96.88%	98.44%	96.88%

Table A9. Accuracy of each model combination for green circles.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	90.91%	93.94%	84.85%	89.39%	93.94%	83.33%
Catboost	89.39%	90.91%	89.39%	81.82%	84.85%	75.76%
LightGBM	89.39%	92.42%	90.91%	83.33%	86.40%	75.76%
XGboost	90.91%	87.88%	87.88%	81.82%	83.33%	69.70%
AdaBoost	84.85%	84.85%	89.39%	83.33%	81.82%	68.18%
GradientBoosting	84.85%	90.91%	87.88%	81.82%	86.36%	74.24%

Table A10. Accuracy of each model combination for green diamonds.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	85.19%	83.33%	81.48%	88.89%	77.78%	85.19%
Catboost	87.04%	88.89%	79.63%	88.89%	77.78%	83.33%
LightGBM	85.19%	87.04%	77.78%	92.59%	83.30%	74.07%
XGboost	81.48%	83.33%	77.78%	88.89%	77.78%	72.22%
AdaBoost	77.78%	83.33%	74.07%	81.49%	79.63%	77.78%
GradientBoosting	79.63%	81.48%	84.48%	90.74%	79.63%	79.63%

Table A11. Accuracy of each model combination for green ovals.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	91.53%	96.61%	86.44%	86.44%	96.61%	81.36%
Catboost	94.92%	98.31%	93.22%	88.14%	96.61%	81.36%
LightGBM	89.83%	93.22%	86.44%	88.14%	91.50%	81.36%
XGboost	89.83%	94.92%	93.22%	86.44%	98.31%	79.66%
AdaBoost	89.83%	89.83%	88.14%	79.66%	91.53%	71.19%
GradientBoosting	91.53%	96.61%	91.53%	88.14%	91.53%	91.36%

Table A12. Accuracy of each model combination for green stars.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	87.50%	89.58%	89.58%	93.75%	89.58%	83.33%
Catboost	89.58%	91.67%	93.75%	93.75%	89.58%	83.33%
LightGBM	87.50%	93.75%	93.75%	89.58%	87.50%	85.42%
XGboost	87.50%	93.75%	93.75%	93.75%	87.50%	87.50%
AdaBoost	89.58%	91.67%	95.83%	91.67%	87.50%	89.58%
GradientBoosting	85.42%	89.58%	95.83%	95.83%	83.33%	83.33%

Table A13. Accuracy of each model combination for green triangles.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	86%	90%	80%	76%	90%	84%
Catboost	90%	90%	80%	78%	88%	82%
LightGBM	94%	84%	76%	80%	86%	88%
XGboost	92%	88%	82%	80%	84%	82%
AdaBoost	94%	88%	78%	78%	94%	76%
GradientBoosting	92%	86%	82%	78%	86%	82%

Table A14. Accuracy of each model combination for all green-colored geometries.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	91.72%	92.02%	83.74%	86.50%	90.18%	82.82%
Catboost	91.41%	91.10%	87.42%	86.81%	92.33%	84.05%
LightGBM	90.80%	92.02%	88.34%	84.36%	91.40%	83.44%
XGboost	91.10%	92.33%	87.12%	80.67%	91.72%	84.97%
AdaBoost	82.21%	88.65%	80.67%	77.61%	85.28%	76.38%
GradientBoosting	90.18%	92.02%	86.50%	80.67%	91.10%	84.66%

Appendix C

Table A15. Accuracy of each model combination for blue squares.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	93.88%	93.88%	93.88%	91.84%	93.88%	67.35%
Catboost	89.80%	97.96%	89.80%	93.88%	91.84%	65.31%
LightGBM	93.88%	93.88%	91.84%	91.84%	95.90%	77.55%
XGboost	87.76%	91.84%	91.84%	91.84%	95.92%	77.55%
AdaBoost	87.76%	97.96%	83.67%	89.80%	95.92%	77.55%
GradientBoosting	91.84%	93.88%	89.80%	93.88%	93.88%	67.35%

Table A16. Accuracy of each model combination for blue circles.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	91.53%	89.83%	88.14%	69.49%	93.22%	76.27%
Catboost	94.92%	91.53%	93.22%	74.58%	96.61%	74.58%
LightGBM	96.61%	93.22%	93.22%	83.05%	94.90%	74.58%
XGboost	96.61%	93.22%	91.53%	77.97%	94.92%	76.27%
AdaBoost	89.83%	86.44%	93.22%	79.66%	89.83%	72.88%
GradientBoosting	98.31%	93.22%	94.92%	77.97%	94.92%	76.27%

Table A17. Accuracy of each model combination for blue diamonds.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	96.15%	92.31%	69.23%	76.92%	92.31%	63.46%
Catboost	92.31%	92.31%	73.08%	76.92%	75.00%	65.38%
LightGBM	94.23%	88.46%	82.69%	76.92%	80.80%	63.46%
XGboost	94.23%	86.54%	76.92%	75.00%	78.85%	61.54%
AdaBoost	90.38%	90.38%	73.08%	76.92%	80.77%	53.85%
GradientBoosting	92.31%	88.46%	78.85%	78.85%	84.62%	61.54%

Table A18. Accuracy of each model combination for blue ovals.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	92.73%	92.73%	80.00%	61.82%	92.73%	70.91%
Catboost	89.09%	98.18%	80.00%	63.64%	90.91%	72.73%
LightGBM	90.91%	94.55%	83.64%	65.45%	89.10%	69.09%
XGboost	92.73%	94.55%	80.00%	65.45%	89.09%	69.09%
AdaBoost	89.09%	85.45%	81.82%	67.27%	94.55%	61.82%
GradientBoosting	90.91%	94.55%	80.00%	69.09%	90.91%	74.55%

Table A19. Accuracy of each model combination for blue stars.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	90.91%	94.55%	81.82%	67.27%	92.73%	72.73%
Catboost	89.09%	92.73%	85.45%	74.55%	96.36%	72.73%
LightGBM	90.91%	94.55%	81.82%	76.36%	94.50%	65.45%
XGboost	89.09%	94.55%	81.82%	74.55%	92.73%	70.91%
AdaBoost	83.64%	85.45%	85.45%	69.09%	85.45%	69.09%
GradientBoosting	90.61%	94.55%	81.82%	72.73%	94.55%	65.45%

Table A20. Accuracy of each model combination for blue triangles.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	84.91%	88.68%	90.57%	84.91%	81.13%	69.81%
Catboost	84.91%	84.91%	83.02%	96.23%	81.13%	64.15%
LightGBM	83.02%	83.02%	83.02%	86.79%	86.80%	67.92%
XGboost	84.91%	83.02%	88.68%	88.68%	84.91%	69.81%
AdaBoost	86.79%	84.91%	81.13%	83.02%	81.13%	69.81%
GradientBoosting	88.68%	83.02%	83.02%	90.57%	84.91%	69.81%

Table A21. Accuracy of each model combination for all blue-colored geometries.

	VGG16	VGG19	ResNet50	EfficientNetB0	InceptionV3	EfficientNetV2L
RandomForest	93.73%	91.75%	81.19%	72.94%	87.46%	64.03%
Catboost	90.43%	90.10%	81.85%	73.60%	90.43%	62.71%
LightGBM	92.74%	91.42%	82.84%	78.88%	89.10%	67.99%
XGboost	89.11%	89.77%	83.17%	74.59%	89.44%	65.35%
AdaBoost	87.13%	86.80%	77.23%	69.31%	84.16%	58.09%
GradientBoosting	87.46%	90.10%	80.86%	73.60%	89.11%	63.70%

Appendix D

Table A22. Accuracy of the baseline (AlexNet with SVM) for all geometries in three colors.

	Gray	Green	Blue
Square	96.30%	95.31%	73.47%
Triangle	85.19%	90.00%	77.36%
Circle	85.71%	90.30%	83.05%
Oval	84.09%	81.36%	67.27%
Diamond	81.82%	61.11%	69.23%
Star	90.57%	85.42%	78.05%
All pictures	88.32%	86.50%	78.22%

References

1. Singamneni, S.; Lv, Y.; Hewitt, A.; Chalk, R.; Thomas, W.; Jordison, D. Additive Manufacturing for the Aircraft Industry: A Review. *J. Aeronaut. Aerospace Eng.* **2019**, *8*, 351–371. [CrossRef]
2. Lee, J.Y.; An, J.; Chua, C.K. Fundamentals and applications of 3D printing for novel materials. *Appl. Mater. Today* **2017**, *7*, 120–133. [CrossRef]
3. Liu, J.; Sheng, L.; He, Z.Z. Liquid metal wheeled 3D-printed vehicle. In *Liquid Metal Soft Machines*; Springer: Singapore, 2019; pp. 359–372.
4. Ricles, L.M.; Coburn, J.C.; Di Prima, M.; Oh, S.S. Regulating 3D-printed medical products. *Sci. Transl. Med.* **2018**, *10*, eaan6521. [CrossRef] [PubMed]
5. Calignano, F.; Manfredi, D.; Ambrosio, E.P.; Biamino, S.; Lombardi, M.; Atzeni, E.; Salmi, A.; Minetola, P.; Iuliano, L.; Fino, P. Overview on additive manufacturing technologies. *Proc. IEEE* **2017**, *105*, 593–612. [CrossRef]
6. Boschetto, A.; Bottini, L. Design for manufacturing of surfaces to improve accuracy in fused deposition modeling. *Robot. Comput. Integr. Manuf.* **2016**, *37*, 103–114. [CrossRef]
7. Valerga, A.P.; Batista, M.; Salguero, J.; Girot, F. Influence of PLA filament conditions on characteristics of FDM parts. *Materials* **2018**, *11*, 1322. [CrossRef]
8. Ottman, N.; Ruokolainen, L.; Suomalainen, A.; Sinkko, H.; Karisola, P.; Lehtimäki, J.; Lehto, M.; Hanski, I.; Alenius, H.; Fyhrquist, N. Soil exposure modifies the gut microbiota and supports immune tolerance in a mouse model. *J. Allergy Clin. Immunol.* **2019**, *143*, 1198–1206.e12. [CrossRef]
9. Conway, K.M.; Pataky, G.J. Crazing in additively manufactured acrylonitrile butadiene styrene. *Eng. Fract. Mech.* **2019**, *211*, 114–124. [CrossRef]
10. Heidari-Rarani, M.; Rafiee-Afarani, M.; Zahedi, A.M. Mechanical characterization of FDM 3D printing of continuous carbon fiber reinforced PLA composites. *Compos. B Eng.* **2019**, *175*, 107147. [CrossRef]
11. Ahmed, S.W.; Hussain, G.; Al-Ghamdi, K.A.; Altaf, K. Mechanical properties of an additive manufactured CF-PLA/ABS hybrid composite sheet. *J. Thermoplast. Compos. Mater.* **2021**, *34*, 1577–1596. [CrossRef]
12. Bacha, A.; Sabry, A.H.; Benhra, J. Fault diagnosis in the field of additive manufacturing (3D printing) using Bayesian networks. *Int. J. Online Biomed. Eng.* **2019**, *15*, 110–123. [CrossRef]
13. Kadam, V.; Kumar, S.; Bongale, A.; Wazarkar, S.; Kamat, P.; Patil, S. Enhancing surface fault detection using machine learning for 3D printed products. *Appl. Syst. Innov.* **2021**, *4*, 34. [CrossRef]
14. Zhang, Y.; Hong, G.S.; Ye, D.; Zhu, K.; Fuh, J.Y. Extraction and evaluation of melt pool, plume and spatter information for powder-bed fusion AM process monitoring. *Mater. Des.* **2018**, *156*, 458–469. [CrossRef]

15. Priya, K.; Maheswari, P.U. Deep Learnt Features and Machine Learning Classifier for Texture classification. In *Journal of Physics: Conference Series*; IOP Publishing: Bristol, UK, 2021; Volume 2070, p. 012108.
16. Delli, U.; Chang, S. Automated process monitoring in 3D printing using supervised machine learning. *Procedia Manuf.* **2018**, *26*, 865–870. [CrossRef]
17. Khadilkar, A.; Wang, J.; Rai, R. Deep learning–based stress prediction for bottom-up SLA 3D printing process. *Int. J. Adv. Manuf. Technol.* **2019**, *102*, 2555–2569. [CrossRef]
18. Baumgartl, H.; Tomas, J.; Buettner, R.; Merkel, M. A deep learning-based model for defect detection in laser-powder bed fusion using in-situ thermographic monitoring. *Prog. Addit. Manuf.* **2020**, *5*, 277–285. [CrossRef]
19. Pham, G.N.; Lee, S.H.; Kwon, O.H.; Kwon, K.R. Anti-3D weapon model detection for safe 3D printing based on convolutional neural networks and D2 shape distribution. *Symmetry* **2018**, *10*, 90. [CrossRef]
20. Katiyar, A.; Behal, S.; Singh, J. Automated Defect Detection in Physical Components Using Machine Learning. In Proceedings of the 8th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 17–19 March 2021; IEEE Publications: Piscataway, NJ, USA, 2021; pp. 527–532.
21. Garfo, S.; Muktadir, M.A.; Yi, S. Defect detection on 3D print products and in concrete structures using image processing and convolution neural network. *J. Mechatron. Robot.* **2020**, *4*, 74–84. [CrossRef]
22. Chen, W.; Zou, B.; Huang, C.; Yang, J.; Li, L.; Liu, J.; Wang, X. The defect detection of 3D-printed ceramic curved surface parts with low contrast based on deep learning. *Ceram. Int.* **2023**, *49*, 2881–2893. [CrossRef]
23. Zhou, W.; Chen, Z.; Li, W. Dual-Stream Interactive Networks for No-Reference Stereoscopic Image Quality Assessment. *IEEE Trans. Image Process.* **2019**, *28*, 3946–3958. [CrossRef]
24. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
25. Paraskevoudis, K.; Karayannis, P.; Koumoulos, E.P. Real-time 3D printing remote defect detection (stringing) with computer vision and artificial intelligence. *Processes* **2020**, *8*, 1464. [CrossRef]
26. Putra, M.A.P.; Chijioke, A.L.A.; Verana, M.; Kim, D.S.; Lee, J.M. Efficient 3D printer fault classification using a multi-block 2D-convolutional neural network. *J. Korean Inst. Commun. Inf. Sci.* **2022**, *47*, 236–245. [CrossRef]
27. Zhou, J.; Yang, X.; Zhang, L.; Shao, S.; Bian, G. Multisignal VGG19 network with transposed convolution for rotating machinery fault diagnosis based on deep transfer learning. *Shock Vib.* **2020**, *2020*, 1–12. [CrossRef]
28. Kim, H.; Lee, H.; Kim, J.S.; Ahn, S.H. Image-based failure detection for material extrusion process using a convolutional neural network. *Int. J. Adv. Manuf. Technol.* **2020**, *111*, 1291–1302. [CrossRef]
29. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
30. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826.
31. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
32. He, K.; Zhang, X.; Ren, S.; Sun, J. Identity Mappings in Deep Residual Networks. In *Eur Conference on Computer Vision*; Springer: Cham, Switzerland, 2016; pp. 630–645.
33. Jin, Z.; Zhang, Z.; Gu, G.X. Automated real-time detection and prediction of interlayer imperfections in additive manufacturing processes using artificial intelligence. *Adv. Intell. Syst.* **2020**, *2*, 1900130. [CrossRef]
34. Ruhi, Z.M.; Jahan, S.; Uddin, J. A Novel Hybrid Signal Decomposition Technique for Transfer Learning Based Industrial Fault Diagnosis. *Ann. Emerg. Technol. Comput. (AETiC)* **2021**, *5*, 37–53. [CrossRef]
35. He, D.; Liu, C.; Chen, Y.; Jin, Z.; Li, X.; Shan, S. A rolling bearing fault diagnosis method using novel lightweight neural network. *Meas. Sci. Technol.* **2021**, *32*, 125102. [CrossRef]
36. Zhang, W.; Li, X.; Ding, Q. Deep residual learning-based fault diagnosis method for rotating machinery. *ISA Trans.* **2019**, *95*, 295–305. [CrossRef]
37. Sun, W.; Luo, W.; Min, X.; Zhai, G.; Yang, X.; Gu, K.; Ma, S. MC360IQA: The Multi-Channel CNN for Blind 360-Degree Image Quality Assessment. In Proceedings of the 2019 IEEE International Symposium on Circuits and Systems (ISCAS), Sapporo, Japan, 26–29 May 2019; IEEE: Piscataway, NJ, USA, 2019.
38. Tan, M.; Le, Q. EfficientNet: Rethinking model scaling for convolutional neural networks. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
39. Tan, M.; Le, Q. EfficientNetV2: Smaller Models and Faster Training. In Proceedings of the 38th International Conference on Machine Learning, Online, 18–24 July 2021; pp. 10096–10106.
40. Zhou, G.; Luo, L.; Xu, H.; Zhang, X.; Guo, H.; Zhao, H. Brick Yourself Within 3 Minutes. In Proceedings of the International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; IEEE Publications: Piscataway, NJ, USA, 2022; pp. 6261–6267.
41. Li, B.; Wang, Z.; Wu, N.; Shi, S.; Ma, Q. Dog Nose Print Matching with Dual Global Descriptor Based on Contrastive Learning. *arXiv* **2022**, arXiv:2206.00580.

42. Wang, K.; Yu, L.; Xu, J.; Zhang, S.; Qin, J. Energy consumption intelligent modeling and prediction for additive manufacturing via multisource fusion and inter-layer consistency. *Comput. Ind. Eng.* **2022**, *173*, 108720. [CrossRef]
43. Banadaki, Y.; Razaviarab, N.; Fekrmandi, H.; Li, G.; Mensah, P.; Bai, S.; Sharifi, S. Automated quality and process control for additive manufacturing using deep convolutional neural networks. *Recent Prog. Mater.* **2022**, *4*, 1. [CrossRef]
44. Razaviarab, N.; Sharifi, S.; Banadaki, Y.M. Smart additive manufacturing empowered by a closed-loop machine learning algorithm. In *Nano-, Bio-, Info-Tech Sensors and 3D Systems III*; SPIE: Bellingham, WA, USA, 2019.
45. Jia, X.; Xiao, B.; Zhao, Z.; Ma, L.; Wang, N. Bearing fault diagnosis method based on CNN-LightGBM. In Proceedings of the IOP Conference Series: Materials Science and Engineering, Sanya, China, 12–14 November 2021; IOP Publishing: Bristol, UK, 2021; Volume 1043, p. 022066.
46. Available online: https://www.prusa3d.com/product/original-prusa-i3-mk3s-kit-3/ (accessed on 29 December 2022).
47. Available online: https://tw-3dp.com/ (accessed on 29 December 2022).
48. Available online: https://www.sony.com.tw/en/electronics/interchangeable-lens-cameras/ilce-7m3-body-kit (accessed on 29 December 2022).
49. Available online: https://colab.research.google.com/ (accessed on 29 December 2022).
50. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
51. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
52. Zhao, X.; Wu, Y.; Lee, D.L.; Cui, W. iforest: Interpreting random forests via visual analytics. *IEEE Trans. Vis. Comput. Graph.* **2018**, *25*, 407–416. [CrossRef]
53. Bühlmann, P. Bagging, boosting and ensemble methods. In *Handbook of Computational Statistics*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 985–1012.
54. Ghojogh, B.; Crowley, M. The theory behind overfitting, cross validation, regularization, bagging, and boosting: Tutorial. *arXiv* **2019**, arXiv:1905.12787.
55. Ganaie, M.A.; Hu, M. Ensemble Deep Learning: A Review. *arXiv* **2021**, arXiv:2104.02395. [CrossRef]
56. Freund, Y.; Schapire, R.E. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.* **1997**, *55*, 119–139. [CrossRef]
57. Wang, F.; Li, Z.; He, F.; Wang, R.; Yu, W.; Nie, F. Feature learning viewpoint of AdaBoost and a new algorithm. *IEEE Access* **2019**, *7*, 149890–149899. [CrossRef]
58. Friedman, J.H. Greedy function approximation: A gradient boosting machine. *Ann. Statist.* **2001**, *29*, 1189–1232. [CrossRef]
59. Anghel, A.; Papandreou, N.; Parnell, T.; De Palma, A.; Pozidis, H. Benchmarking and optimization of gradient boosting decision tree algorithms. *arXiv* **2018**, arXiv:1809.04559.
60. Shi, Y.; Li, J.; Li, Z. Gradient boosting with piece-wise linear regression trees. *arXiv* **2018**, arXiv:1802.05640.
61. Bentéjac, C.; Csörgő, A.; Martínez-Muñoz, G. A comparative analysis of gradient boosting algorithms. *Artif. Intell. Rev.* **2021**, *54*, 1937–1967. [CrossRef]
62. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
63. Sagi, O.; Rokach, L. Ensemble learning: A survey. *WIREs Data Mining Knowl. Discov.* **2018**, *8*, e1249. [CrossRef]
64. Zounemat-Kermani, M.; Batelaan, O.; Fadaee, M.; Hinkelmann, R. Ensemble machine learning paradigms in hydrology: A review. *J. Hydrol.* **2021**, *598*, 126266. [CrossRef]
65. Hancock, J.; Khoshgoftaar, T.M. Leveraging LightGBM for Categorical Big Data. In Proceedings of the IEEE Seventh International Conference on Big Data Computing Service and Applications (BigDataService), Athens, Greece, 17–20 July 2021; IEEE Publications: Piscataway, NJ, USA, 2021; pp. 149–154.
66. Tanha, J.; Abdi, Y.; Samadi, N.; Razzaghi, N.; Asadpour, M. Boosting methods for multi-class imbalanced data classification: An experimental review. *J. Big Data.* **2020**, *7*, 1–47. [CrossRef]
67. Zhang, Y.; Liu, J.; Shen, W. A review of ensemble learning algorithms used in remote sensing applications. *Appl. Sci.* **2022**, *12*, 8654. [CrossRef]
68. Dorogush, A.V.; Ershov, V.; Gulin, A. CatBoost: Gradient boosting with categorical features support. *arXiv* **2018**, arXiv:1810.11363.
69. Prokhorenkova, L.; Gusev, G.; Vorobev, A.; Dorogush, A.V.; Gulin, A. CatBoost: Unbiased boosting with categorical features. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 3–8 December 2018; pp. 6639–6649.
70. Hancock, J.T.; Khoshgoftaar, T.M. CatBoost for big data: An interdisciplinary review. *J. Big Data* **2020**, *7*, 94. [CrossRef]
71. González, S.; García, S.; Del Ser, J.; Rokach, L.; Herrera, F. A practical tutorial on bagging and boosting based ensembles for machine learning: Algorithms, software tools, performance study, practical perspectives and opportunities. *Inf. Fusion.* **2020**, *64*, 205–237. [CrossRef]
72. Shrestha, A.; Mahmood, A. Review of deep learning algorithms and architectures. *IEEE Access* **2019**, *7*, 53040–53065. [CrossRef]
73. Alzubaidi, L.; Zhang, J.; Humaidi, A.J.; Al-Dujaili, A.; Duan, Y.; Al-Shamma, O.; Santamaría, J.; Fadhel, M.A.; Al-Amidie, M.; Farhan, L.; et al. Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *J. Big Data* **2021**, *8*, 53. [CrossRef] [PubMed]

74. Wang, X.; Chen, C.; Yuan, J.; Chen, G. Color reproduction accuracy promotion of 3D-printed surfaces based on microscopic image analysis. *Int. J. Pattern Recognit. Artif. Intell.* **2020**, *34*, 2054004. [CrossRef]
75. Li, Z.; Zhang, Z.; Shi, J.; Wu, D. Prediction of surface roughness in extrusion-based additive manufacturing with machine learning. *Robot. Comput. Integr. Manuf.* **2019**, *57*, 488–495. [CrossRef]
76. Amihai, I.; Gitzel, R.; Kotriwala, A.M.; Pareschi, D.; Subbiah, S.; Sosale, G. An Industrial Case Study Using Vibration Data and Machine Learning to Predict Asset Health. In Proceedings of the 20th Conference on Business Informatics (CBI), Vienna, Austria, 11–13 July 2018; IEEE Publications: Piscataway, NJ, USA, 2018.
77. Paolanti, M.; Romeo, L.; Felicetti, A.; Mancini, A.; Frontoni, E.; Loncarski, J. Machine Learning Approach for Predictive Maintenance in Industry 4.0. In Proceedings of the 14th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA), Oulu, Finland, 2–4 July 2018; IEEE Publications: Piscataway, NJ, USA, 2018; pp. 1–6.
78. Jayasudha, M.; Elangovan, M.; Mahdal, M.; Priyadarshini, J. Accurate estimation of tensile strength of 3D printed parts using machine learning algorithms. *Processes* **2022**, *10*, 1158. [CrossRef]
79. Gardner, J.M.; Hunt, K.A.; Ebel, A.B.; Rose, E.S.; Zylich, S.C.; Jensen, B.D.; Wise, K.E.; Siochi, E.J.; Sauti, G.; Siochi, E.J.; et al. Machines as craftsmen: Localized parameter setting optimization for fused filament fabrication 3D printing. *Adv. Mater. Technol.* **2019**, *4*, 1800653. [CrossRef]
80. Rahmani Dabbagh, S.R.; Ozcan, O.; Tasoglu, S. Machine learning-enabled optimization of extrusion-based 3D printing. *Methods* **2022**, *206*, 27–40. [CrossRef]