Materials & Design 206 (2021) 109726

Contents lists available at ScienceDirect

Materials & Design

journal homepage: www.elsevier.com/locate/matdes

Geometrical defect detection for additive manufacturing with machine learning models

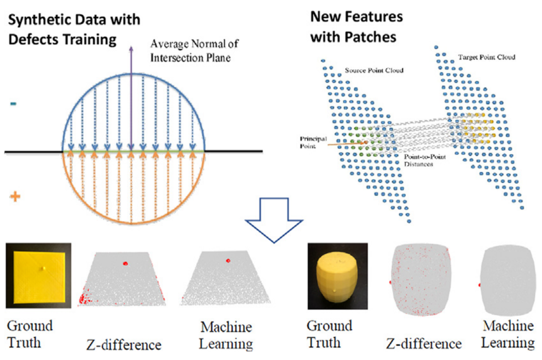
Rui Li [a], Mingzhou Jin [a,*], Vincent C. Paquit [b]

[a] Industrial and Systems Engineering, University of Tennessee, Knoxville, USA
[b] Oak Ridge National Laboratory, USA

HIGHLIGHTS

- A systematic machine-learning framework for additive manufacturing defect detection with synthetic data for training.
- Features associated with the new "patch concept" for more macro-level information.
- Five machine-learning methods tested with both synthetic and experimental data.
- Better defect detection performance on additively manufactured objects than the Z-difference method.

GRAPHICAL ABSTRACT



ARTICLE INFO

Article history:
Received 16 February 2021
Revised 30 March 2021
Accepted 9 April 2021
Available online 20 April 2021

Keywords:
Additive manufacturing
Defect detection
Machine learning

ABSTRACT

This study proposed a scheme based on Machine Learning (ML) models to detect geometric defects of additively manufactured objects. The ML models are trained with synthetic 3D point clouds with defects and then applied to detect defects in actual production. Using synthetic 3D point clouds rather than experimental data could save a huge amount of training time and costs associated with many prints for each design. Besides distance differences of individual points between source and target point clouds, this scheme uses a new concept called "patch" to capture macro-level information about nearby points for ML training and implementation. Numerical comparisons of prediction results on experimental data with different shapes showed that the proposed scheme outperformed the existing Z-difference method in the literature. Five ML methods (*Bagging of Trees, Gradient Boosting, Random Forest, K-nearest Neighbors* and *Linear Supported Vector Machine)* were compared under various conditions, such as different point cloud densities and defect sizes. *Bagging* and *Random Forest* were found the two best models regarding predictability; and the right patch size was found to be at 20. The proposed ML-based scheme is applicable to *in-situ* defect detection during additive manufacturing with the aid of a proper 3D data acquisition system.

© 2021 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

1. Introduction

Additive Manufacturing (AM) produces three-dimensional (3D) objects layer by layer according to computer-aided design (CAD) models. AM has advantages on complicated geometric designs over traditional manufacturing [1] but has limitations of slow production speed, high cost, and low accuracy [2]. It is estimated that superficial defects caused 10% of failures in AM-manufactured parts [3]. AM defects include discontinuities and flaws, such as porosity, layer shift, inclusions, delamination and lack of fusion. All these defects may lead to geometrical inaccuracy

* Corresponding author at: 851 Neyland Drive, TN 37996, USA.
E-mail address: jin@utk.edu (M. Jin).

https://doi.org/10.1016/j.matdes.2021.109726
0264-1275/© 2021 The Author(s). Published by Elsevier Ltd.
This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

and negatively influence mechanical properties [4]. Many AM parts need postprocesses to satisfy surface finish requirements. Meanwhile, AM processes normally take a long time, hours or even days, to finish one part. Although AM machine and material costs are decreasing in recent years, they are still higher than traditional manufacturing [5]. Therefore, *in-situ* defect detection during AM processes instead of post-production measurement will help to save production time and expense through early termination or real-time parameter adjustment. *In-situ* defect detection has been a technical challenge for the AM community for decades [6 7]. A major barrier is the complex and dynamic characteristics of AM [8] which make *in-situ* data acquisition and analysis difficult. The real-time data acquisition system for defect detection is an important part of an *in-situ* quality control system. There are many different acquisition techniques, such as vision-based [9] ultrasound [10] acoustic emission [11] laser scanning [12] electromagnetic [13] radiographic [1415] and thermographic techniques [8] Among them, vision-based detection recently became popular because of its automation and reliability [9]. The vision-based detection methods for AM may use two types of data, (i) 2D images and (ii) 3D meshes.

Methods dealing with 2D information have been developed for a long time but have some limitations. 2D images do not supply enough information for defect typologies so it is difficult to detect precise metrology of small defects, which are important to AM processes requiring high accuracy [16]. Most methods processing 2D information continuously take photos or record videos during AM build processes. The 2D photos or frames extracted from videos are analyzed by various imaging process techniques. Visible light cameras with infrared (IR) filters and high-speed shutters or IR light cameras are often used in this kind of AM monitoring [5]. Barua et al. [9] used digital cameras with macro lenses as an acquisition tool to monitor the melt pool of laser metal deposition (LMD). They took the RGB values of the images to calculate a temperature map with a linear regression model. The temperature map reflects the presence of flaws by the altered conduction of heat. Different light patterns can be used to detect defects as well. When the light from different orientations is projected on a product, a defect may disrupt the continuity of the contrast change [17]. Another way to identify defects with 2D information is to project a structured highlight on object surfaces [18 19]. Usually there is a dominant reflection from specular surfaces. For instance, the structured highlight from point or striped light sources is reflected by the specular surface to form a reflectance map. If a light source is moving or multiple light sources from different directions project on the same object, local orientations of the surface can be estimated from the viewing direction, source direction and specular constraint. Extended Gaussian images (EGI), combined with the structured highlight method, can represent the 3D shape of an object surface and estimate the orientation histogram. The calculated shape features using the orientation histogram can show distribution properties of local surface orientations. It is acknowledged that this approach for shape classification does not need registration and matching of local position features. Adopting lighted stripes may enhance accuracy but requires greater computational effort, which is not suitable for online AM monitoring [19]. Aluze et al. [20] improved the computational efficiency by using both lighted and dark stripes to increase the contrast between defective and flawless areas. However, the method is only suitable to certain surfaces because the light-ray path does not reflect on certain areas of a concave shape.

Methods dealing with 3D information are arising because of the recent progress in 3D data processing techniques. Some 3D acquisition systems require multiple digital cameras at different angles to collect images and then reconstruct 3D models, while others use 3D scanners to rebuild the 3D models of parts [21]. Madrigal et al. [16] used the deformation of the light patterns to reconstruct 3D models with a descriptor called Model Points Feature Histogram (MPFH). The MPFH data are used as the inputs for a *support vector machine (SVM)*, a supervised machine learning (ML) model, to do classification. Inline Coherent Imaging (ICI) is an integrated low coherence interferometric imaging technique to reconstruct 3D models, which is closely related to spectral-domain optical coherence tomography (SD-OCT) and provides new insight into melt pool geometry [16 22]. Holzmond and Li [23] proposed an *in-situ* approach in which two digital cameras are set up above the working plate from opposite directions to reconstruct 3D point clouds. A point cloud is a set of data points in space to represent the geometry of objects. The Z-differences of points between the reconstructed point cloud (source point cloud) and the point cloud generated from designed 3D mesh (target point cloud) are calculated after registration with the Iterative Closest Point (ICP) algorithm. Here, *Z-difference* means the distance along *z*-axis from the horizontal aligned source point cloud to the target point cloud. Registration, also known as point matching, is a process of finding a spatial transformation that aligns the source point cloud and target point cloud. A threshold of Z-differences between two clouds is used to classify defects. Although this approach is efficient and easy to apply in production, its accuracy was not reported. In this study, we tested their method and compared it with our method regarding accuracy in subsection 2.3.

In summary, most existing defect detection methods for AM processes cannot realize *in-situ* detection. Some methods can be applied *in-situ*, but they either cannot handle small-sized defects or did not report accuracy. ML algorithms are promising to increase the detection accuracy of AM defects [16] because ML may use multiple features simultaneously rather than only the distances between points (e.g., [23]). ML has been applied in broad ranges of manufacturing [24 25] and demonstrated great potential for quality control, especially in complex environments [26]. Taking only the field of metalworking as an example, ML models have already been implemented in process control of sheet metal forming [27] rolling [28] machining [29] welding [30] casting [31] etc. ML algorithms are also used recently in different AM processes. Quite a number of researches focused on the melt pool of powder bed fusion (PBF) process. Binary classifiers were applied on PBF machines with 3D CT scan data and their accuracy are greater than 80% [32]. Support Vector Machines and Convolutional Neural Network (CNN) were proposed to identify quality level of PBF process with a high speed camera as the image data acquisition system [33]. L. Scime and J. Beuth conducted a series of research of defect detection on laser powder bed fusion process with various machine learning and deep learning methods [34–37]. Similarly, ML and deep learning methods were also utilized in quality monitoring of Fused Filament Fabrication (FFF) [38] and Selective Laser Melting (SLM) [39]. However, ML normally requires a large data volume to train models. If all the data are from experiments, the data collection stage will be time-consuming and costly because of energy and material waste. To address this issue, this study proposes to generate a large volume of synthetic data for model training and then use the trained model for detection during real production to save time and cost.

2. Methodology

AM defects can be detected in real time by periodically comparing the 3D reconstructed model of an in-building part with the design mesh. The whole AM production process can be divided into multiple phases. For example, the building process of a turbine blade can be separated into four phases, as shown in Fig. 1. Monitoring devices, such as digital cameras or 3D scanners, can collect

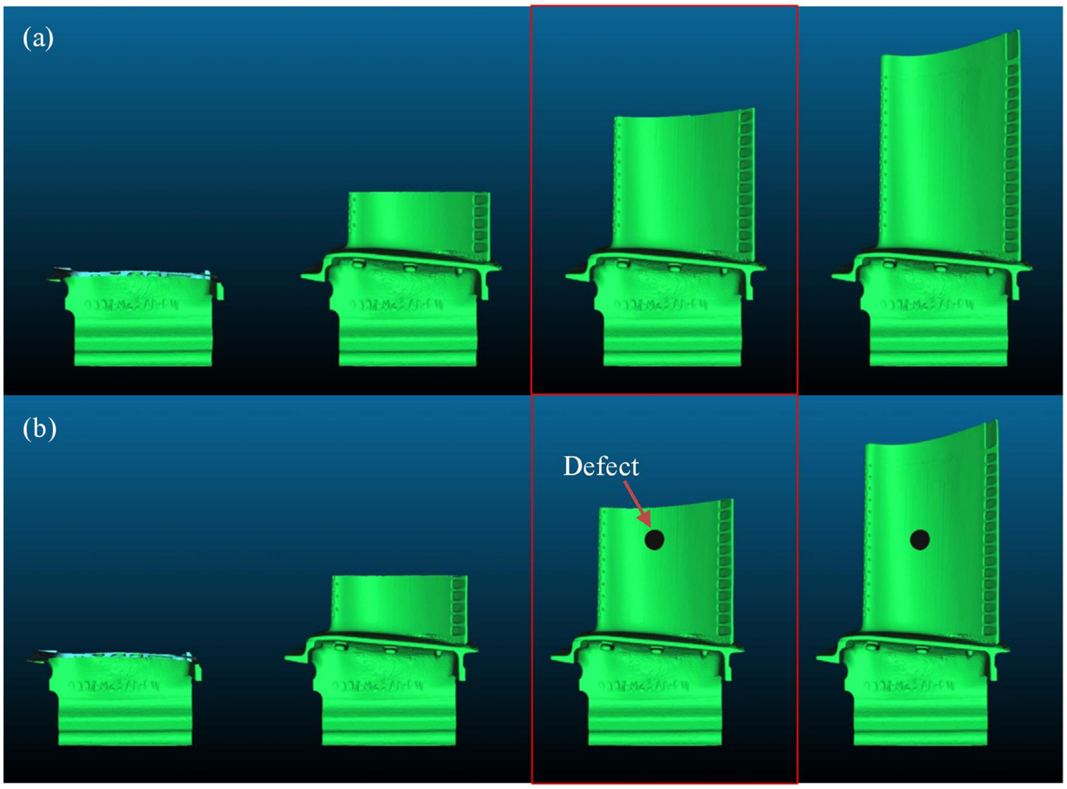


Fig. 1. Schematic of additively manufacturing a turbine blade with four phases: (a) perfect turbine blade; (b) turbine blade with a hemispherical defect on it.

visual data at the end of each phase. After the 3D mesh of each phase is reconstructed, defects can be detected by comparing these 3D meshes with the original design mesh. If defects are detected in the third phase, as shown in the red rectangle frame in Fig. 1, the AM process can be terminated to save time and materials without finishing the whole build.

2.1. Method framework

The main framework of the proposed method is illustrated in Fig. 2 with three steps: data generation, data preparation, and model building. For a given design, it is almost impossible to create a large number of AM builds for ML model training to detect defects because of the long build time and high costs associated with each build. Instead, we propose to use synthetic data for ML model training. A 3D mesh from the design file of an AM part is first used to generate synthetic point clouds, which include randomly created defects. The shapes of those artificial defects may be selected based on the prior experiences of defects for the same or similar AM machines and part designs. In this study, hemispherical bump defects are used to illustrate and validate the proposed method. Defective and perfect synthetic point clouds can be generated through a two-way approach that will be discussed in subsection 2.2 in detail. We use Poisson sampling and add white noise to simulate data points that are collected through 3D scanning during implementation. The AM part point cloud with hemispherical defects is referred to as a source point cloud (defective); and the cloud without defects is called a target point cloud (perfect). Data preparation includes registration of the source and target point clouds, calculation of the distance and other features of each data point, which are stored in comma-separated values (CSV) for ML training. We trained five common ML models with the preprocessed datasets and evaluated them based on two criteria, F-measure and G-mean. The models and the whole proposed method were tested and compared with actual AM parts in Section 3.

2.2. 3D synthetic data generation

The original 3D triangle mesh of an AM part is used as the base to add artificial defects. Defects of various shapes can be generated with python library *Open3D* [40] such as spheres, cuboids, tetrahedrons, icosahedrons, octahedrons, and torus. A hemisphere is selected in this study since it can represent common defects like bumps in AM production. After the spherical-defect triangle mesh is created, the location for the defect on the surface of the AM part is randomly chosen. Any point in the cloud converted from the triangle mesh of the AM part is a candidate of the central location of the spherical defect with the same probability.

The union of a defect and a base shape is an important part of data generation for ML training and may follow two different ways (called Way 1 and Way 2), depending on whether the base mesh is watertight. Way 1: if the base triangle mesh is watertight, like a barrel, the union can be easily realized by the *union* function in Library *Trimesh* [41]. Way 2: if the base triangle mesh is not watertight, like a flat surface, the *union* function of triangle meshes will not work and a manual union process is necessary. To make a hemisphere-shaped defect from a sphere and add it to the base mesh, we first calculate a vector from each point on the sphere surface to the intersection between the sphere and the mesh. Based on the sign of the dot product of this vector and the normal vector of the intersection plane, every point on the sphere can be classified into either the "internal" or "external" half. Only the external half of the sphere point cloud and the base mesh out of the intersection part, which is a plane for a flat shape, are kept and united (Fig. 3). The united AM part with a hemisphere-shaped defect mesh is then used to generate a 3D point cloud, and a Poisson disk sampling from *Open3d* is applied. Poisson disk sampling means each point has approximately

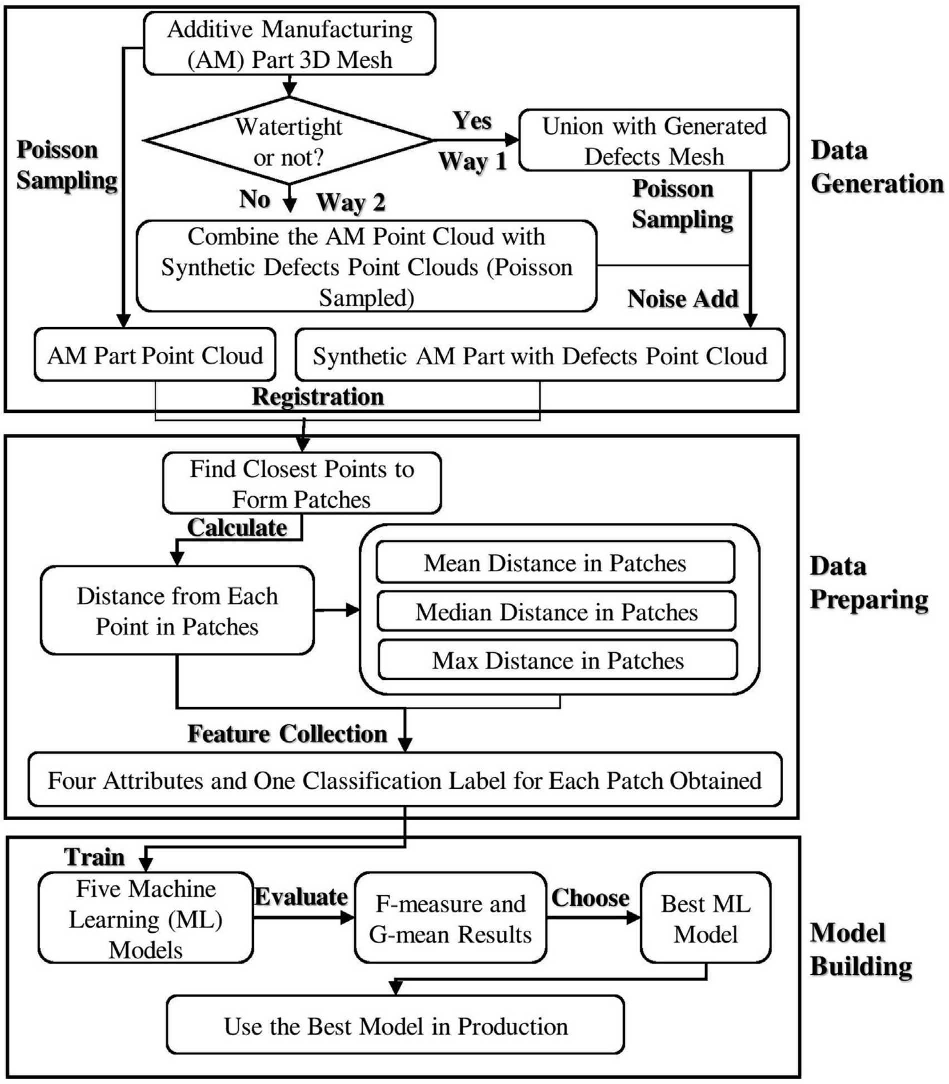

Additive Manufacturing (AM) Part 3D Mesh
Watertight or not?
Yes
Way 1
Union with Generated Defects Mesh
Data Generation
Poisson Sampling
No
Way 2
Poisson Sampling
Combine the AM Point Cloud with Synthetic Defects Point Clouds (Poisson Sampled)
Noise Add
AM Part Point Cloud
Synthetic AM Part with Defects Point Cloud
Registration
Find Closest Points to Form Patches
Calculate
Distance from Each Point in Patches
Mean Distance in Patches
Median Distance in Patches
Max Distance in Patches
Data Preparing
Feature Collection
Four Attributes and One Classification Label for Each Patch Obtained
Train
Five Machine Learning (ML) Models
Evaluate
F-measure and G-mean Results
Choose
Best ML Model
Model Building
Use the Best Model in Production

Fig. 2. Schematic framework of the proposed method based on synthetic dataset and ML.

the same distance to the neighboring points. It is known that in reality the reconstructed 3D models always have some noise, so random noises $\vec{v} \cdot l$ are added on each point along a certain direction $\vec{v}$, where l follows a normal distribution (i.e.,l $N(0, \sigma)$ mm) and σ is chosen according to printing condition from experience. The parameter $\vec{v}$ here is the unit normal vector at each point.

2.3. Data preparation for ML

After data generation, we have a 3D point cloud with a known defect (called source point cloud) and a base 3D point cloud without any defect (called target point cloud). We define I and J for the set of points in the source cloud and target cloud, respectively. During the training stage, we have a label $L_i \in \{0, 1\}$for each point i in the source cloud, where "0″ means "non-defective" while "1" means "defective". During the implementation stage, we are trying to classify each point i in the source cloud, which is collected through some sensors, into one of these two categories. One opportunity for AM defect detection is that we have the target cloud from a design file to facilitate this classification. The next step is data registration to match source point cloud and target point cloud to extract the features of each point in the source cloud, which will be used in ML training. During the ML implementation stage, experimental data need to be denoised before registration. The Point-to-Point Iterative Closest Point (ICP) algorithm from python library *Open3D* [24] and *CloudCompare* [42] are used in this study for the registration. ICP is a widely used algorithm to align two point clouds through minimizing the total difference between them. Afterwards, point-to-point distances, $d_{ij}, \forall i \in I, \forall j \in J$, between the source and target point clouds are calculated. For each point i in the source cloud, we find a point $j^*(i)$ in the target cloud with the minimum distance from i and define this minimum distance as $D_i = \min_{\forall j \in J} d_{ij}$, where $j^*(i) = \operatorname{argmin}_{\forall j \in J} d_{ij}$. In other words, $D_i = d_{ij^*(i)}$.In this study, besides D_i, we also use the information of neighboring points around point i in the source cloud for the classification because neighboring points may be correlated due to a defect while their errors in data collection during implementation (or noises in the authentic cloud) may be rather independent.

To reflect the correlation and connection among neighboring points in the source point cloud, a concept called "patch" is introduced to obtain a more holistic relationship among the minimum distances to the target cloud of closing points in the source point cloud (Fig. 4). For each point $i \in I$, its corresponding patch P_i is a set of $n_p$points in the source cloud that are the closest to point i, where $P_i \subset I$ and $i \in P_i$. Here, the patch size n_p is set at 20 point in this study and could be adjusted in practice.

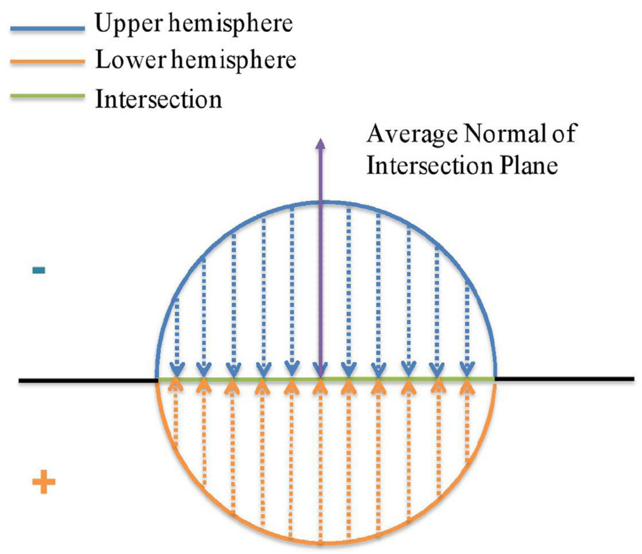


Fig. 3. Method to generate a hemisphere from a sphere as defect to be united with an AM part. When the dot products of the vectors (from points on sphere to intersection plane) have the same signs, either positive (orange part) or negative (blue part), these points are selected for the union with the AM part. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

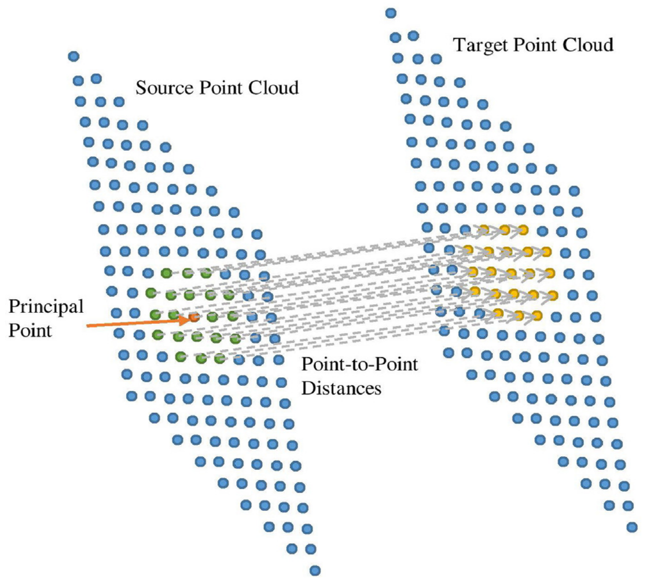


Fig. 4. Concept of a patch: Each dot represents one specific point in point clouds. Green dots represent points in the patch in the source point cloud while yellow dots are for the corresponding patch in the target point cloud. Blue points are other points outside the patch in both clouds. The orange dot is the principal point of the patch in the source point cloud. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Since the minimum distance of every point in the source cloud to the target cloud is known, the average, median, and maximum distances to the target cloud among all points in a patch P_i can be calculated as follows and used to feature the corresponding source point i.

$$E_i = \frac{\sum_{k \in P_i} D_k}{n_p}, \tag{1}$$

$$m_i = \underset{k \in P_i}{\operatorname{median}} D_k, \text{and} \tag{2}$$

$$M_i = \max_{k \in P_i} D_k. \tag{3}$$

The four features of $D_i, E_i, m_i$and$M_i$and the label of L_i were used for each source point during ML training.

2.4. Model training and evaluation

Five ML algorithms, *Bagging of Trees (Bagging), Gradient Boosting, Random Forest, Linear Support Vector Machines (Linear SVM)* and *K-nearest Neighbors (KNN)*, were applied to build models and compared in this study. *Bagging (Bootstrap Aggregation)* methods randomly draw samples (bootstrap) from dataset with replacement, train multiple base classifiers, and aggregate the prediction results by voting. The base classifiers are mostly decision trees [43]. Its mathematical formula is given as

$$\hat{f}_{avg}(x) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^b(x), \tag{4}$$

where B is the number of bootstrapping datasets and $\hat{f}^b(x)$ is the classification tree built from the b^{th} dataset.

One of the firstly generated boosting algorithm in history is *Adaptive Boosting* (*AdaBoosting*) [44] which summarizes and weights the predictions of a lot of weak learners (one-split decision trees). The prediction from all weak classifiers $G_m(x)$ are combined through a weighted majority vote to produce the final prediction

$$G(x) = sign\left(\sum_{m=1}^{M} \alpha_m G_m(x)\right), \tag{5}$$

where $\alpha_1, \alpha_2, \cdots, \alpha_M$are the final weights affiliating with each respective $G_m(x)$. Higher weights are given to more accurate classifiers in the sequence. *Gradient boosting* improves *AdaBoosting* by optimizing an arbitrary differentiable loss function. It can adjust the decision tree's depth so the weak learners are not decision stumps anymore [45]. *Random Forest* is like an extension of *Bagging*. The difference of it from Bagging is that its classifiers can choose features instead of using all features to make a split at each node of the decision trees [46]. *Random Forest* improves the variance reduction of *Bagging* by reducing the correlation between the trees. *Support Vector Machines* (*SVM*) can find the samples close to the boundary of different classes (support vectors). *SVM* finds a hyperplane that maximize the distance between support vectors and the hyperplane [47]. The maximal margin hyperplane is the solution to the following optimization problem:

$$maximize_{\beta_0,\beta_1,\cdots,\beta_p,\xi_1,\cdots,\xi_n} M \tag{6}$$

$$subjectto \sum_{j=1}^{p} \beta_j^2 = 1, \tag{7}$$

$$y_i(\beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi}) \geq M(1 - \xi_i), \forall i \in \{1, \cdots, n\} \tag{8}$$

$$\xi_i \geq 0, \sum_{i=1}^{n} \xi_i \leq C. \tag{9}$$

Here, M represents the margin of the hyperplane and the optimization problem chooses $\beta_0, \beta_1, \cdots, \beta_p$ that maximizes M. (x_i, y_i) is the i^{th} observation. C is a nonnegative tuning parameter. $\xi_1, \cdots, \xi_n$ are slack variables that allow individual observations to be on the wrong side of the hyperplane. Because the two classes (defective and non-defective) in the source cloud data of this paper do not overlap much in the feature space, *Linear SVM* [48] is employed because *Linear SVM* is much faster than *SVMs* with other kernels. KNN classifies an object through voting by its neighbors [49] and use the following function.

$$\hat{Y}(x) = \frac{1}{k}\sum_{x_i \in N_k(x) \subseteq T} y_i, \tag{10}$$

where $N_k(x)$ is the neighborhood of x, T is the training set, $\widehat{Y}(x)$ is the estimated observation value, and k is the number of closest points in the training set. All functions of these five ML algorithms are from python library scikit-learn [50].

When the portion of the defective points in the whole cloud is too small, especially for the objects with small radius defects, the ML prediction of rare events (i.e., defective points) can be unsatisfactory. Classifiers may predict everything as the major class. Solutions for this kind of imbalanced datasets include sampling methods and algorithmic methods [51]. Sampling methods, like over-sampling and under-sampling, are popular solutions [52]. Adding rare class samples into original datasets and removing major class samples from original datasets are known as over-sampling and under-sampling, respectively. Under-sampling may cause loss of useful information, so Synthetic Minority Over-sampling Technique (SMOTE) [53] is first used on data in this study. SMOTE generates synthetic samples instead of replication of the original samples. SMOTE demands that there should be at least six samples for the minority class. For cases with fewer than six rare samples, random over-sampling is used to compensate. Both SMOTE and random over-sampling methods are called in this study from a python library *Imbalanced-Learn* [54].

For some cases where the above sampling methods do not work, a naive way of sampling is applied. Since the 3D data are synthetically created, one way to increase the defect point sets is to simply increase the number of hemispherical defects. This renders more balanced datasets. In order to solve the data imbalance problem, algorithmic methods are another alterative. *SVM* and *KNN* with over-sampling are supposed to be effective to predict minority class [55]. Ensemble-based methods, like *Bagging, Boosting* and *Random Forest*, are also beneficial to imbalanced dataset problems [56]. Ensemble means a combination of multiple classifiers to enhance the prediction accuracy. Because of data imbalance (i.e., the proportion of defective points in the whole point cloud is too small), using the simple accuracy (number of correctly classified points over number of all points) cannot effectively tell the model performance on prediction of defective points. *F-measure* and *G-mean* (Geometric mean) [57] are applied to evaluate ML models in this paper; they are commonly used in imbalanced data classification and focus more on the minority class [55]. From the prediction results, the numbers (points) of true positive (*TP*), true negative (*TN*), false positive (*FP*) and false negative (*FN*) out of the classification are known, which define *Recall*, *Specificity*, and *Precision*, as shown in Table 1. Here, "positive" means defective while "negative" means non-defective.

With the contents in the confusion matrix, *Recall*, *Specificity* and *Precision* are defined as below:

$$Precision = \frac{TP}{TP + FP}, \tag{11}$$

$$Recall = \frac{TP}{TP + FN}, \tag{12}$$

$$Specificity = \frac{TN}{TN + FP} \tag{13}$$

Table 1
Definition of Confusion Matrix.

		True Condition	
		Positive	Negative
Predicted Condition	Positive	# of True Positive (*TP*)	# of False Positive (*FP*)
	Negative	# of False Negative (*FN*)	# of True Negative (*TN*)

Furthermore, two metrics of F-measure and G-mean to compare ML models are determined as follows.

$$F - measure = \frac{2 \times \mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}} \tag{14}$$

And

$$G - mean = \sqrt{\mathrm{Recall} \times Specificity} \tag{15}$$

3. Results

To demonstrate the method proposed in Section 2 and evaluate its effectiveness, this study tested the method on synthetic data and conducted experiments for validation with 3D scanning data collected from actual AM-produced parts.

3.1. Results with synthetic data

First, we tested our method on a simple plane. Point clouds were generated from a 50 mm $\times$ 50 mm square plane triangle mesh. Hemispheres with 1.5–2 mm radii were added as defects in the point clouds, as shown in Fig. 5(a). To investigate the method performance on more complex shapes, a 3D triangle mesh of a barrel from [58] was rescaled within a cubic box with 50 mm as its edge length. Then it was combined with hemispherical defects that have 1.5–2 mm radii to create a synthetic defective point cloud, as shown in Fig. 5(b). When generating the point clouds, we added a random noise $\vec{v} \cdot l$ into each point, where l was drawn from a normal distribution $N(0, 0.025)$ in mm for the plane and $N(0, 0.05)$ in mm for the barrel, as introduced in subsection 2.2. These parameters are selected according to our evaluations in experiments.

We created 50 defective point clouds with randomly created defects for each shape. Among them, 40 point clouds were used to train each of the five ML models, and the remaining 10 clouds were used for validation. To test the patch concept in subsection 2.3, we considered four sizes of 10, 20, 30, and 40 points per patch. Fig. 6 shows the *F-measure* and *G-mean* results of each ML model for the barrel with defects of a 1.5–2 mm radius under different patch sizes. The figure shows clearly that the patch size of 20 points performed best. The optimal patch size for the best ML performance should be relevant to the sizes of defects and the density of the point cloud. As a result, the patch size of 20 was used in the remaining experiments.

Five ML models (i.e., *Bagging of Trees, Gradient Boosting, Random Forest, Linear SVM* and *K- nearest Neighbors*) were trained and validated with the barrel synthetic data. As mentioned in subsection 2.4, *F-measure* and *G-mean*, rather than accuracy, were used to compare the models due to the great imbalance between the two classes. Defects with four different radii of [1, 1.5], [1.5, 2], [2, 2.5], and [2.5, 3]mm were separately added into the point clouds

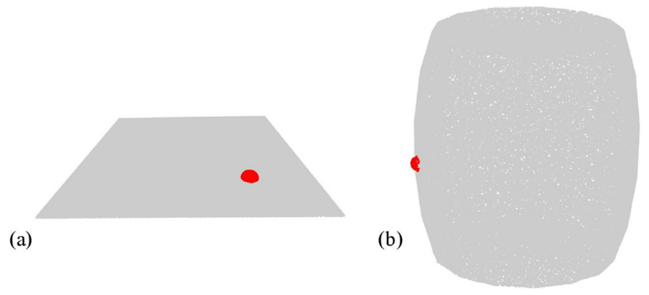


Fig. 5. Synthetic Point Cloud of (a) square plane; (b) barrel. Red points represent defective points. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

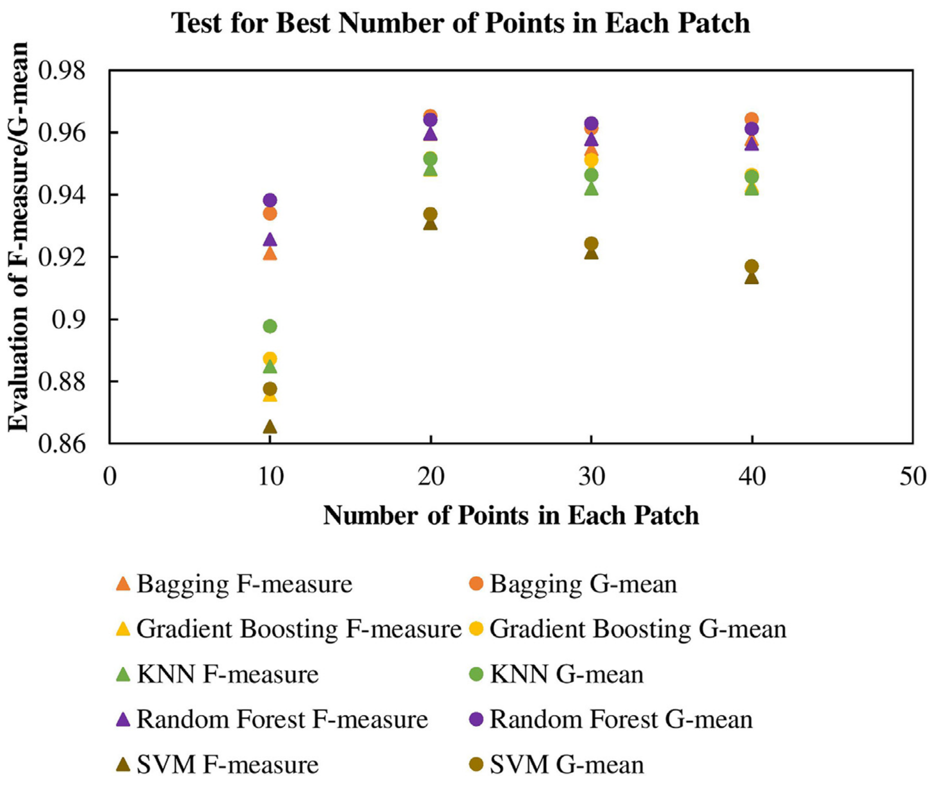


Fig. 6. Evaluation results with synthetic data with different numbers of points in each patch.

to see the performance of the ML models on various defect sizes. Each point cloud point had about 100,000 points and included 10 hemispherical defects. *F-measure* of each group can be determined. The *F-measure* results of the five ML models are plotted in Fig. 7. All five models perform well with *F-measure* above 90%. Among them, *Bagging* and *Random Forest* are the best two models in predictability. Regarding computational time, *Linear SVM* is fastest, usually completing the calculation within only one or two minutes, including both tuning and training, while the other four models took hours just for tuning.

3.2. Validation with 3D scanned data

In addition to using synthetic point clouds to evaluate the effectiveness of the proposed ML method, we tested the method with actual AM parts of plane and barrel shapes printed by MakerBot Replicator 2X. This 3D printer is a full-featured desktop Fused Filament Fabrication (FFF) printer using acrylonitrile butadiene styrene plastic filament with a build volume of 155 × 246 × 163 mm. Surface roughness of FFF process usually depends on the build orientation, but the feed rate has no significant effect [59]. In this study, we intend to detect the defects after they form but do not investigate the influence of process parameters on the shape and size of the geometric defects of AM-produced parts. Fig. 8 shows the manufactured samples of the plane and barrel with the same designed files as synthetic data in subsection 3.1. There are designed hemispherical defects on the surfaces of these square plane and barrel samples, shown by texts and orange arrows in Fig. 8. The samples were then scanned by an EinScan SP Desktop 3D scanner. The scanned data were down-sampled into point-clouds with the same point density as in synthetic data and denoised via the software *CloudCompare*.

We made great efforts to apply the traditional statistical method called *Z*-difference on an FFF 3D printer in [23]. Two cameras were set above the plate of a 3D printer to reconstruct the 3D

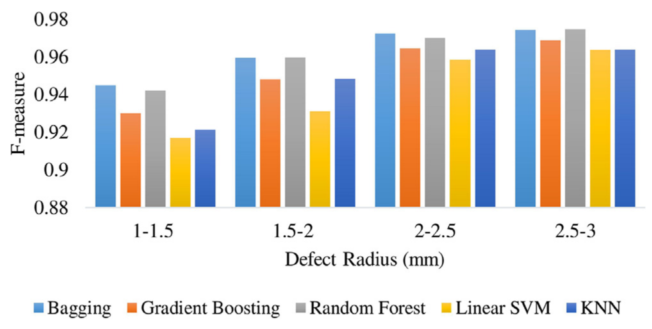


Fig. 7. *F-measure* of five ML models on synthetic barrel data with different hemispherical defect sizes.

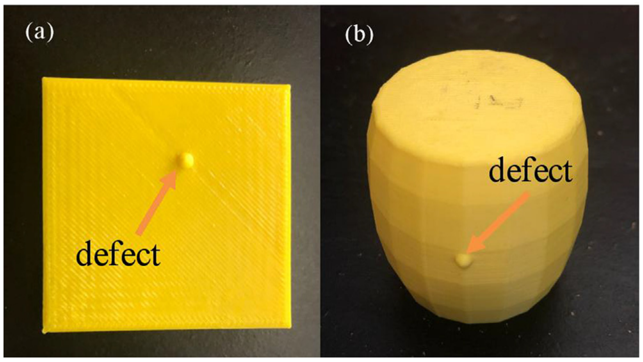


Fig. 8. Experimental printed samples: (a) square plane; (b) barrel. The designed hemispherical defects on the samples are pointed out with orange arrows. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

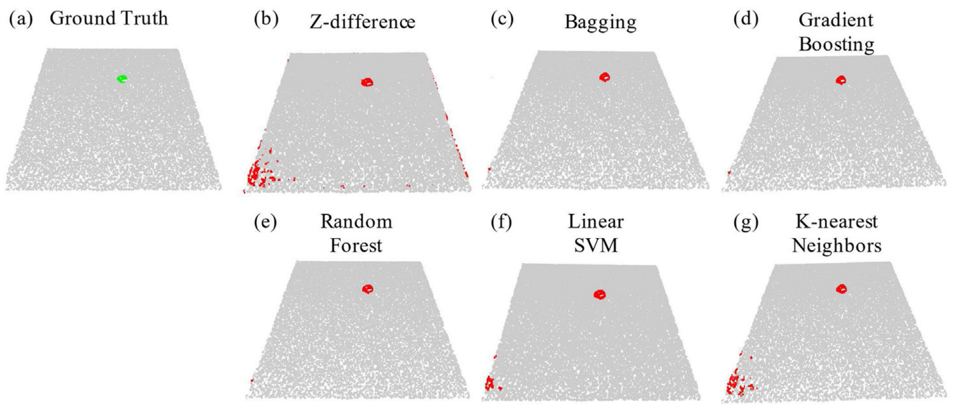


Fig. 9. (a) Ground Truth of hemispherical defect (green points) and Prediction results on plane experimental data (red points) with (b) Traditional Statistical Method; (c) Bagging; (d) Gradient Boosting; (e) Random Forest; (f) Linear SVM; (g) K-nearest Neighbors. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

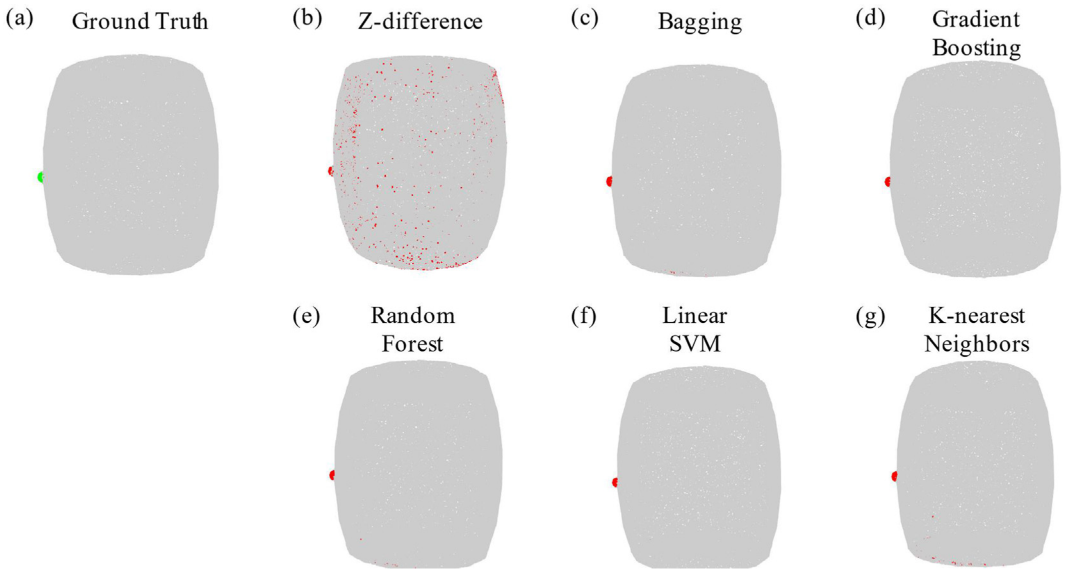


Fig. 10. (a) Ground Truth of hemispherical defect (green points) and Prediction results on barrel experimental data with (b) Z-difference Method in [43]; (c-g) five ML models in this study. Red points present predicted defective points. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

point cloud of a surface. In their method, a threshold $t = \theta\widehat{\sigma}$ was used with the *Z*-difference from reconstructed point cloud to horizontal plane to detect defects, where $\widehat{\sigma}$ is the standard deviation of *Z*-difference of all points. To compare prediction results of their method and our method, $\theta = 2$ was chosen because it performed best on our data when we compared multiple θ values. If $|Dist(x) - D| > t$, where $Dist(x)$ is the *Z*-difference at point x and D is the distance mean among all points, x is listed in the outlier set; otherwise, it belongs to the inlier set. Holzmond and Li showed that the method works well in reality, but its accuracy was not presented in their paper [23].

For a fair comparison, the *Z*-difference method and our method with five ML models were applied to the same experimental data (one square plane and one barrel) and their performance measures of *accuracy*, *F-measure* and *G-mean* were calculated. Figs. 9 and 10 illustrate the classification results for the plane and barrel, respectively, in which red points are the predicted defective points. From the figures it is seen that many noise were predicted as defective points with *Z-difference* method, especially for the barrel example which has a higher complexity than flat planes, while five ML methods all outperform *Z-difference*.

Tables 2 and 3 list the effectiveness of the six methods regarding the three measures on the plane and barrel, respectively. It is clear that our ML method trained with synthetic data outperforms the *Z*-difference method in detection by all three criteria, especially for complex shapes like a barrel. The $F - measure$ of the *Z-difference* method is only 5.51%, while the $F - measure$of *Gradient Boosting* is 89.08%. Different from Fig. 5, we notice that some predicted defective points of ML models are not in the predefined hemispherical defect set. Those points were caused by actual small defects formed during printing. The plane is not exactly flat, and the bottom boundary of the barrel is not very clear. Similar to the testing results on the synthetic data, *Bagging, Gradient Boosting* and *Random Forest* work well on both parts. According to the $F - measure$ values of both synthetic and experimental data, *Gradient Boosting* is the best ML algorithm for detecting defects with a radius above 1.5–2 mm on AM parts.

Reasons why the precision of the *Z*-difference method is lower than ML methods are not difficult to conceive. Fig. 10 plots a histogram of distance with regard to outliers, inliers, and true defect points in the *Z*-difference method on a barrel experimental scanned data. The whole histogram in Fig. 11(a) cannot clearly

Table 2
Comparison of the *Z*-difference and Five ML Models for Detection Effectiveness on the Plane.

Measure	*Z*-difference Method	Bagging	Gradient Boosting	Random Forest	Linear SVM	KNN
Accuracy	0.9886	0.9978	0.9978	0.9978	0.9956	0.9926
F-measure	0.3534	0.7391	0.7391	0.7391	0.5842	0.4578
G-mean	0.4633	0.7656	0.7656	0.7656	0.6424	0.5448

Table 3
Comparison of the *Z*-difference and Five ML Models for Detection Effectiveness on the Barrel.

Measure	*Z*-difference Method	Bagging	Gradient Boosting	Random Forest	Linear SVM	KNN
Accuracy	0.9737	0.9992	0.9997	0.9992	0.9959	0.9979
F-measure	0.0551	0.7397	0.8908	0.7347	0.3986	0.5104
G-mean	0.1695	0.7993	0.9600	0.7946	0.5045	0.5976

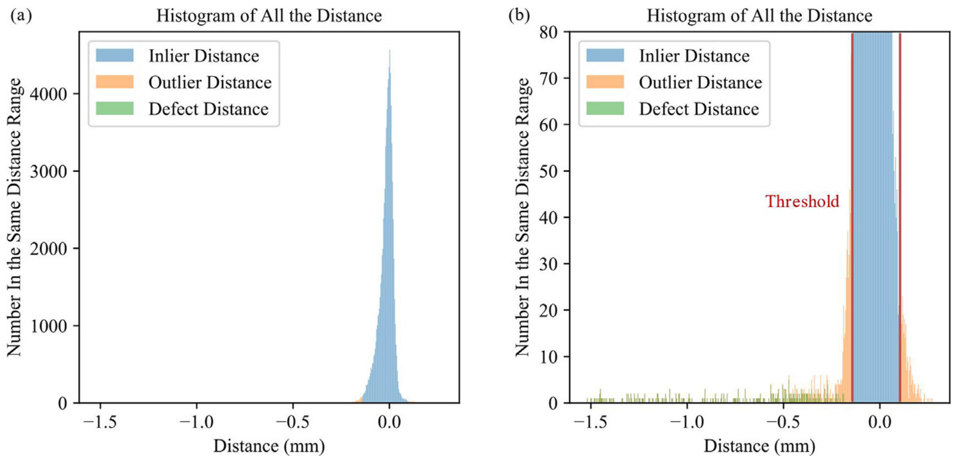


Fig. 11. Histogram of distance with regard to outliers(orange), inliers(blue) and defective points(green) in *Z*-difference method on barrel experimental data: (a) whole histogram of all distance with 500 bins; (b) partial histogram with 500 bins when zooming in. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

show the distances of outlier and defect points, so Fig. 11(b) zooms in and shows the locations of those points at the bottom of *y* axis. Some outliers, which are non-defective points, are classified as defective points. There is an overlap between outliers and defective points regarding distance. Therefore, using only the distance of each individual point from source point cloud to target point cloud is not enough to effectively classify points into the two classes of defective and non-defective, no matter how we change the threshold value. However, if more features related to nearby points are included as attributes, such as maximum distance, mean distance, and median distance of a patch around an individual point, we increase the dimensions of the space for classification.

4. Discussion and conclusions

This paper proposes an ML-based approach that uses synthetic data in training to detect geometric defects in AM parts. Using the synthetic data rather than actually printing and scanning many physical parts for each design for training can save huge amounts of time and money. After incorporating information from nearby points through the new "patch" concept, the proposed method outperforms the traditional statistical method in detection accuracy. Since the ML models can be trained by synthetic data and saved in a .sav format, the trained models can be called and run quickly in real-time AM production to realize in-situ detection. This study investigated five ML models regarding $F-measure$ and $G-mean$, and demonstrated high precision on both synthetic data and 3D scanned data from actual AM parts, outperforming the statistic method used in the literature. Our method can be applied to any AM process as long as the process is open air or allowed to set up visual data (images or 3D scanning data) acquisition system in it. Another requirement to implement our method is that the process has to be paused when the data acquisition system is working if it has strong light sparkling or reflection (which often occurs in laser-based processes), because strong light may impact the vision data collection.

Although experiments show promising results of our proposed method, its applications in real AM processes require further efforts. Right now, the features used for ML models are still all related to the distances between the target and source point clouds, though the "patch" concept includes some neighborhood information. More features, such as curvature, may be used to capture more macro-level features beyond individual points to improve classification precision. However, we need to address the high sensitivity of curvature calculated from point clouds. The balance between the drawbacks and benefits of having more features may be another interesting research topic. Furthermore, to realize in-situ application of our methods, an accurate real-time 3D scanner needs to be set up on the AM machines. The AM parts were scanned in reality after they were fully produced in this study. Another issue is the speed of the registration step, which took a significant amount of time in this study and will potentially hurt the training efficiency when the method is applied to many AM designs. The proposed method trains ML models for each AM design and then uses it separately in production, even though the trained models can be used repeatedly for the same design during production. We will investigate whether ML models can be trained and tested with objects of different shapes, which can significantly improve implementation efficiency. In addition, an investigation on the influence of process parameters on the shape and size of the geometric defects of AM-produced parts will be an interesting future research direction and may improve the detection accuracy. All the above research questions are worth study, and we are confident that they can all be thoroughly addressed.

This study is expected to lead to progress in AM quality control and management. For instance, criteria or policies can be developed based on the real-time classification results to decide whether an AM production should be stopped or continue at any moment. An early termination for a defective part means huge cost and time savings for AM because of its long build time compared to conventional manufacturing. We plan to explore dynamic decision-making models, such as Markov Decisions Processes, based on detection results in the future. Furthermore, we plan to develop a complete tool, from synthetic data generation, experimental data collection, model training, and implementation in real production. The tool should be able to generate synthetic defective point clouds for various shapes of AM parts, consider defect shapes, and incorporate various geometrical accuracy tolerance requirements.

CRediT authorship contribution statement

Rui Li: Conceptualization, Methodology, Software, Validation, Formal analysis, Writing - original draft, Visualization. **Mingzhou Jin:** Supervision, Conceptualization, Methodology, Writing - review & editing, Resources. **Vincent C. Paquit:** Conceptualization, Methodology.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] N. Guo, M.C. Leu, Additive manufacturing: Technology, applications and research needs, Front. Mech. Eng., vol. 8, no. 3. Springer, pp. 215–243, 08-Sep-2013.
[2] I. Campbell, D. Bourell, I. Gibson, Additive manufacturing : rapid prototyping comes of age, Rapid Prototyp. J. 18 (2012) 255–258.
[3] R. Leach, Optical Measurement of Surface Topography, Springer, Berlin Heidelberg, 2011.
[4] C. Mandache, Overview of non-destructive evaluation techniques for metal-based additive manufacturing, Mater. Sci. Technol. (United Kingdom) 35 (9) (2019) 1007–1015.
[5] D.S. Thomas, S.W. Gilbert, Costs and cost effectiveness of additive manufacturing: A literature review and discussion, NIST Spec. Publ. 1176 (2015) 1–96.
[6] S.K. Everton, M. Hirsch, P.I. Stavroulakis, R.K. Leach, A.T. Clare, Review of in-situ process monitoring and in-situ metrology for metal additive manufacturing, Mater. Des. 95 (Apr. 2016) 431–445.
[7] H. Kim, Y. Lin, T.L.B. Tseng, A review on quality control in additive manufacturing, Rapid Prototyp. J. 24 (3) (2018) 645–669.
[8] T. Wuest, D. Weimer, C. Irgens, K.D. Thoben, Machine learning in manufacturing: Advantages, challenges, and applications, Prod. Manuf. Res. 4 (1) (2016) 23–45.
[9] S. Barua, F. Liou, J. Newkirk, T. Sparks, Vision-based defect detection in laser metal deposition process, Rapid Prototyp. J. 20 (1) (2014) 77–86.
[10] M. Seifi, A. Salem, J. Beuth, O. Harrysson, J.J. Lewandowski, Overview of Materials Qualification Needs for Metal Additive Manufacturing, JOM 68 (3) (Mar. 2016) 747–764.
[11] Q.Y. Lu, C.H. Wong, Additive manufacturing process monitoring and control by non-destructive testing techniques: challenges and in-process monitoring, Virtual Phys. Prototyp. 13 (2) (Apr. 2018) 39–48.
[12] D.G. Lowe, Distinctive image features from scale-invariant keypoints, Int. J. Comput. Vis. 60 (2) (2004) 91–110.
[13] W. Lou et al., Internal defect detection in ferromagnetic material equipment based on low-frequency electromagnetic technique in 20# Steel Plate, IEEE Sens. J. 18 (16) (Aug. 2018) 6540–6546.
[14] C. Hu, Y. Wang, An Efficient Convolutional Neural Network Model Based on Object-Level Attention Mechanism for Casting Defect Detection on Radiography Images, IEEE Trans. Ind. Electron. 67 (12) (Dec. 2020) 10922–10930.
[15] I. Valavanis, D. Kosmopoulos, Multiclass defect detection and classification in weld radiographic images using geometric and texture features, Expert Syst. Appl. 37 (12) (Dec. 2010) 7606–7614.
[16] C.A. Madrigal, J.W. Branch, A. Restrepo, D. Mery, A method for automatic surface inspection using a model-based 3D descriptor, Sensors (Switzerland) 17 (10) (2017) 1–20.
[17] H. Shen, S. Li, D. Gu, H. Chang, Bearing defect inspection based on machine vision, Measurement 45 (2012) 719–733.
[18] S.K. Nayar, A.C. Sanderson, E. Weiss, D.A. Simon, Specular Surface Inspection Using Structured Highlight and Gaussian Images, IEEE Trans. Robot. Autom. 6 (2) (1990) 208–218.
[19] D. Perard, J. Beyerer, Three-dimensional measurement of specular free-form surfaces with a structured-lighting reflection technique, Three-Dimensional Imaging and Laser-based Systems for Metrology and Inspection III, 3204, 1997, pp. 74–80.
[20] D. Aluze, F. Merienne, C. Dumont, P. Gorria, Vision system for defect imaging, detection, and characterization on a specular surface of a 3D object, Image Vis. Comput. 20 (8) (Jun. 2002) 569–580.
[21] W. Lin, H. Shen, J. Fu, S. Wu, Online Quality Monitoring in Material Extrusion Additive Manufacturing Processes based on Laser Scanning Technology, Precis. Eng., Jun. (2019).
[22] J.A. Kanko, A.P. Sibley, J.M. Fraser, In situ morphology-based defect detection of selective laser melting through inline coherent imaging, J. Mater. Process. Technol. 231 (May 2016) 488–500.
[23] O. Holzmond, X. Li, In situ real time defect detection of 3D printed parts, Addit. Manuf. 17 (Oct. 2017) 135–142.
[24] D.-S. Kwak, K.-J. Kim, A data mining approach considering missing values for the optimization of semiconductor-manufacturing processes, Expert Syst. Appl. 39 (3) (Feb. 2012) 2590–2596.
[25] T. Wuest, D. Weimer, C. Irgens, K.-D. Thoben, Machine learning in manufacturing: advantages, challenges, and applications, Prod. Manuf. Res. 4 (1) (Jan. 2016) 23–45.
[26] J.A. Harding, M. Shahbaz, Srinivas, A. Kusiak, Data Mining in Manufacturing: A Review, J. Manuf. Sci. Eng., vol. 128, no. 4, pp. 969–976, Nov. 2006.
[27] E. Hamouche, E.G. Loukaides, Classification and selection of sheet forming processes with machine learning, Int. J. Comput. Integr. Manuf. 31 (9) (Sep. 2018) 921–932.
[28] F. Piltan, A.E. Prosvirin, I. Jeong, K. Im, J.-M. Kim, Rolling-Element Bearing Fault Diagnosis Using Advanced Machine Learning-Based Observer, Appl. Sci. 9 (24) (Dec. 2019) 5404.
[29] D.H. Kim et al., Smart Machining Process Using Machine Learning: A Review and Perspective on Machining Industry, in: International Journal of Precision Engineering and Manufacturing - Green Technology, vol. 5, no. 4. Korean Society for Precision Engineering, pp. 555–568, 01-Aug-2018.
[30] Y. Du, T. Mukherjee, T. DebRoy, Conditions for void formation in friction stir welding from machine learning, npj Comput. Mater. 5 (1) (Dec. 2019) 1–8.
[31] S. Shahane, N. Aluru, P. Ferreira, S.G. Kapoor, S.P. Vanka, Optimization of solidification in die casting using numerical simulations and machine learning, J. Manuf. Process. 51 (Mar. 2020) 130–141.
[32] C. Gobert, E.W. Reutzel, J. Petrich, A.R. Nassar, S. Phoha, Application of supervised machine learning for defect detection during metallic powder bed fusion additive manufacturing using high resolution imaging, Addit. Manuf. 21 (May 2018) 517–528.
[33] Y. Zhang, G.S. Hong, D. Ye, K. Zhu, J.Y.H. Fuh, Extraction and evaluation of melt pool, plume and spatter information for powder-bed fusion AM process monitoring, Mater. Des. 156 (Oct. 2018) 458–469.
[34] L. Scime, J. Beuth, A multi-scale convolutional neural network for autonomous anomaly detection and classification in a laser powder bed fusion additive manufacturing process, Addit. Manuf. 24 (Dec. 2018) 273–286.
[35] L. Scime, J. Beuth, Melt pool geometry and morphology variability for the Inconel 718 alloy in a laser powder bed fusion additive manufacturing process, Addit. Manuf., vol. 29, p. 100830, Oct. 2019.
[36] L. Scime, J. Beuth, Anomaly detection and classification in a laser powder bed additive manufacturing process using a trained computer vision algorithm, Addit. Manuf. 19 (Jan. 2018) 114–126.
[37] L. Scime, J. Beuth, Using machine learning to identify in-situ melt pool signatures indicative of flaw formation in a laser powder bed fusion additive manufacturing process, Addit. Manuf. 25 (Jan. 2019) 151–165.
[38] B.N. Narayanan, K. Beigh, G. Loughnane, N.U. Powar, Support vector machine and convolutional neural network based approaches for defect detection in fused filament fabrication, Appl. Mach. Learn. 11139 (6) (2019) 36.
[39] A. Caggiano, J. Zhang, V. Alfieri, F. Caiazzo, R. Gao, R. Teti, Machine learning-based image processing for on-line defect recognition in additive manufacturing, CIRP Ann. 68 (1) (Jan. 2019) 451–454.
[40] Q.-Y. Zhou, J. Park, V. Koltun, Open3D: A Modern Library for 3D Data Processing [Computer Software]. 29-Jan-2018.
[41] M. Dawson-Haggerty, Trimesh [Computer software]. 2019.

[42] D. Girardeau-Montaut, A. Maloney, and R. Janvier, CloudCompare [GPL software]. 2019.
[43] L. Breiman, Bagging Predictors, Mach. Learn. 24 (2) (1996) 123–140.
[44] Y. Freund, R.E. Schapire, A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting, J. Comput. Syst. Sci. 55 (1) (Aug. 1997) 119–139.
[45] J.H. Friedman, Greedy Function Approximation : A Gradient Boosting Machine 1 Function estimation 2 Numerical optimization in function space, Annu. Stat. 29 (5) (2001) 1189–1232.
[46] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[47] J.A.K. Suykens, J. Vandewalle, Least Squares Support Vector Machine Classifiers, Neural Process. Lett. 9 (3) (1999) 293–300.
[48] R.E. Fan, K.W. Chang, C.J. Hsieh, X.R. Wang, C.J. Lin, LIBLINEAR: A library for large linear classification, J. Mach. Learn. Res. 9 (2008) 1871–1874.
[49] N.S. Altman, An introduction to kernel and nearest-neighbor nonparametric regression, Am. Stat. 46 (3) (1991) 175–185.
[50] F. Pedregosa et al., Scikit-learn: Machine Learning in Python, J. Mach. Learn. Res. 12 (1) (2011) 29–33.
[51] B. Machado, T. Rodrigues, Z. Lopes, R. Lopes, M. Mesquita, Paraparesis: A rare presentation of thrombosis of the abdominal aorta, Eur. J. Intern. Med. 24 (1) (2013) e256.
[52] V. Ganganwar, An overview of classification algorithms for imbalanced datasets, Int. J. Emerg. Technol. Adv. Eng. 2 (4) (2012) 42–47.
[53] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: Synthetic Minority Over-sampling Technique, J. Artif. Intell. Res. 16 (Jun. 2002) 321–357.
[54] G. Lemaître, F. Nogueira, C.K. Aridas, Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning, J. Mach. Learn. Res. 18 (17) (2017) 1–5.
[55] Y. Yong, The Research of Imbalanced Data Set of Sample Sampling Method Based on K-Means Cluster and Genetic Algorithm, Energy Procedia 17 (Jan. 2012) 164–170.
[56] A. Abraham, S.M.A. Elrahman, A Review of Class Imbalance Problem, J. Netw. Innov. Comput. 1 (2013) 332–340.
[57] A. Tharwat, Classification assessment methods , Appl. Comput, Informatics, Aug. 2018.
[58] 3D Barrel. [Online]. Available: https://www.turbosquid.com/3d-models/old-barrel-3d-model-1217355.
[59] E. García Plaza, P. J. Núñez López, M. Á. Caminero Torija, and J. M. Chacón Muñoz, "Analysis of PLA geometric properties processed by FFF additive manufacturing: Effects of process parameters and plate-extruder precision motion," Polymers (Basel)., vol. 11, no. 10, Jan. 2019.