Solid Freeform Fabrication 2022: Proceedings of the 33rd Annual International Solid Freeform Fabrication Symposium – An Additive Manufacturing Conference
Reviewed Paper

Logistic regression classification to predict regional anomalies in nominally printed volume of separate test pieces

Andrew Lang[1,2], James Castle[1], Douglas A. Bristow[2], Robert G. Landers[3], and Venkata Sriram Siddhardh Nadendla[4]

[1]The Boeing Company
[2]Mechanical and Aerospace Engineering, Missouri University of Science and Technology
[3]Aerospace and Mechanical Engineering, University of Notre Dame
[4]Computer Science, Missouri University of Science and Technology

Abstract

Supervised machine learning techniques have struggled to accurately predict voxel-wise occurrence of anomalies in metal powder bed parts printed with optimal processing parameters. This work discusses a method to visualize machine learning model predictions in 3D to interrogate patterns in the predictions. A simple logistic regression classifier, with cross validation and an optimized classification threshold, is trained using synthetic in situ features, a machine parameter, and post-process output labels. The developed classifier is shown to outperform deep learning and boosted classifiers on the datasets used. Voxel-wise prediction performance is very low, but 3D representation of model predictions shows the developed model can predict anomalies in the correct region of the printed part. The practical use of the developed method is demonstrated by predicting the occurrence of anomalies in nominally printed volume using a model that had been trained on a dataset printed with induced defects.

1.0 Introduction

Laser Powder Bed Fusion (LPBF) is a form of Additive Manufacturing where a laser selectively fuses consecutive layers of metal powder to additively build components [1]. There has been an extensive body of work predicting and validating quality of parts printed with LPBF, particularly with machine learning [2, 3]. Research in this field includes techniques for online monitoring, terminating or adjusting build parameters in situ, and post-build analysis to validate performance and reliability of the build. With machine learning, performance measures of a resulting part can be predicted based on in situ signals, and a picture of resulting part quality can be formed independent of non-destructive investigation (NDI), such as X-ray computed tomography (XCT) or other quality testing.

Machine Learning is a computer program that adapts to new circumstances and detects and extrapolate patterns without being explicitly programed to do so [4]. There are three main types of learning: reinforcement learning, unsupervised learning, and supervised learning. Reinforcement learning is where an agent learns by a series of reinforcements: penalties for negative results and rewards for positive results. In unsupervised learning, the agent learns patterns in the input even though no explicit feedback is supplied; the most common task being clustering. Contrary to

unsupervised learning, in supervised learning the agent observes example input-output pairs and learns a function that maps from input to output. This paper focuses on supervised machine learning where a logistic regression model is formed to map input in situ signals and machine parameters to output labels defined by clustered XCT images. The results on a model trained with logistic regression classifier are compared to results of models trained with several other machine learning models. The machine learning models in this work are implemented in Python with Scikit-learn [5].

Supervised machine learning is a common method for quality prediction of material properties (e.g., density) in LPBF with many studies focusing on process-property correlation between process parameters and property outputs of the printed part [6]. For in situ defect detection, thermal, optical, acoustic, and other sensing can be used. Input parameters for a supervised machine learning model can be in situ signal data, process and physics models, and/or machine parameters. A classifier can then be trained to predict property outputs, using labels generated on outputs such as density, surface quality, and mechanical properties.

Many studies show model performance on parts printed with a large number of flaws, often induced with non-ideal processing conditions (higher/lower laser power, exposure time, contaminated powder, increased/decreased scan speed or hatch spacing, etc) [7, 8, 9, 10, 11]. Voxel-wise machine learning performance is typically very low in volume printed under nominal conditions where defects were minimal or not intentionally induced. High voxel-wise performance of predictive models, measured by both accuracy and recall, has been a particular challenge of in situ-based machine learning models for metal LBPF. Most of the research on in situ monitoring to predict part density investigates the correlation between in situ signals and overall part density with only a few that demonstrate location of flaws. The few studies that focus on specific flaw identification tend to have high false positive rates [12]. In this study, a machine learning model is trained on a section of a test part with induced defects and then the trained model is used to generate predictions on sections of test parts that were not printed with induced defects.

Compiling all the elements of the produced part results in a unique “fingerprint”. The fingerprint of an additive build is the fusion of four categories of parameters: machine parameters (laser power, scan strategy, etc), measured in situ signals (optical, thermal, acoustic, etc), process modeling and simulation (physics models, computational fluid dynamics, finite element analysis, etc), and post process evaluation (XCT, SEM, strength/ fatigue testing, etc). Like a human fingerprint, each build has a unique fingerprint based on the unique process characteristics and resulting part. These dimensions of build parameters comprise a full-view picture of the build, from the inputs in the machine to the part printed. This paper demonstrates how some elements of one part’s fingerprint, specifically thermal in situ signals, laser power, and XCT, can be used to predict density voids in a test piece printed separately from the training dataset.

2.0 Experimental Setup

For this study, two ASTM E8 tensile specimens [13] have been printed in 304L stainless steel using a Renishaw AM250 LPBF machine. The two tensile specimens are referred to in this paper as Part-1 and Part-2. Each test specimen is 50 mm tall and has a 4 mm diameter neck. The

nominal laser power for the builds was 200W and a section of lettering ("Missouri S&T") was printed at 100W inside the neck of the test piece to induce defects. The point distance, d_p, and the hatch spacing, d_h, were held at constant 60 µm and 85 µm, respectively. In situ data was captured using a Short Wave Infrared (SWIR) camera, XCT imaging was used to generate a 3D array of the part density, and 2D SEM images were taken of extracted sections of the test specimen.

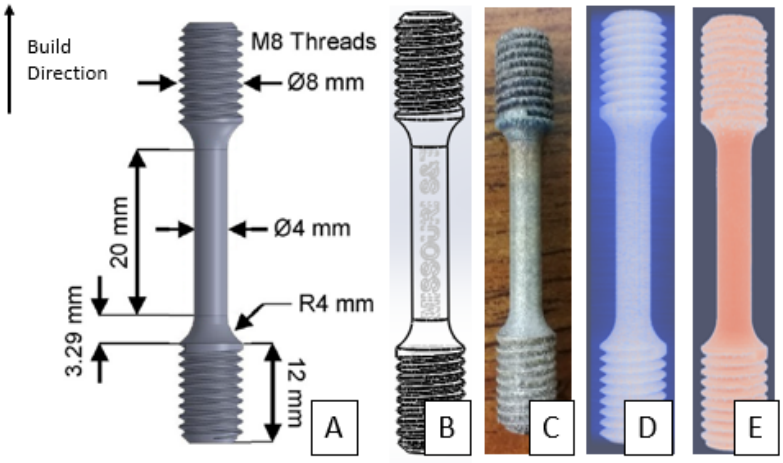

Figure 1: ASTM E8 tensile specimen (a), low power lettering printed in the neck to induce defects (b), the final tensile bar printed by LPBF (c), 3D reconstruction of in situ SWIR imaging (d), and 3D reconstruction of XCT imaging (e).

Three of the four dimensions of a build fingerprint are represented in this study: measured in situ signals are captured by the SWIR imaging, the machine parameter, laser power, is utilized, and post-process product evaluation is included with the XCT imaging. This study does not include process modeling and simulation elements, the fourth dimension of the build fingerprint.

2.1 Measured in Situ Signals: SWIR

The SWIR camera set up and thermal feature extraction has been performed in other work [7, 14]; the process is summarized in this section. An FLIR SC6201 SWIR camera measured visible light emitted from the melt pool. The camera was installed at an observation angle of 15º above the build chamber to observe the build. The camera pixel array was reduced to an 80 × 80 pixel window enabling high frame rate recording (~2500 Hz). A non-uniformity correction (NUC) was performed to account for differences in the SWIR measurements across the imaging area due to the observation angle ($\theta = 15°$) of the SWIR camera shown in Figure 3. The final corrected pixel dimensions were 125 µm x 125 µm for each layer. The layer height of this build was 50 µm.

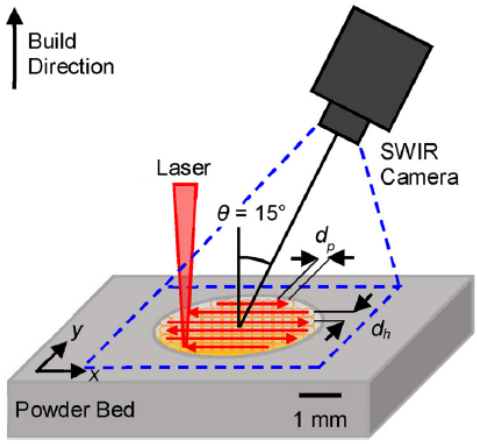

Figure 2: Schematic of SWIR camera observations [7]

Thermal features from the time-series recording were extracted to a single image representation of each layer with three separate metric arrays: time above threshold, τ, maximum radiance, L_{max}, and maximum radiance decrease rate, ΔL_{max}. Each 2D layer was concatenated, layer-by-layer, to create a 3D voxel-based reconstruction of the tensile specimen in HDF5 format.

The time above threshold is the total time that a given pixel is above a set threshold. The threshold selected for this study is based on the apparent phase transition region of a melt pool and is set to 12,000 counts of radiance. The time above threshold is:

$$\tau(x,y) = |\{t: L(x,y,t) \geq threshold\}| \quad (1)$$

Where $L(x, y, t)$ denotes the radiance measurement of the pixel, (x, y) at time, t.

The maximum radiance is simply the maximum value recorded at each pixel across all time values of the layer:

$$L_{max}(x,y) = |\{t: max[L(x,y,t)]\}| \quad (2)$$

Finally, the maximum radiance decrease rate is:

$$\Delta L_{max}(x,y) = \left|\{t: -min\left[\frac{L(x,y,t+1)-L(x,y,t)}{\Delta t}\right]\}\right| \quad (3)$$

2.2 Machine Parameter: Laser Power

There are many machine parameters in LPBF. These parameters can be programmed inputs into the build like scan spacing and velocity, laser beam power and radius, or dwell time between layers. These parameters are fed into the part design or reported from the machine itself (rather than in situ signals or images from the melt pool or powder bed). The only machine parameter used in this study is laser power. Figure 3 shows an example of a slice of the Laser Power array of

Part-1 highlighting where “Missouri S&T” was printed in the center of the part at 100W to induce defects for model training. The nominal build volume of the parts was printed at 200W.

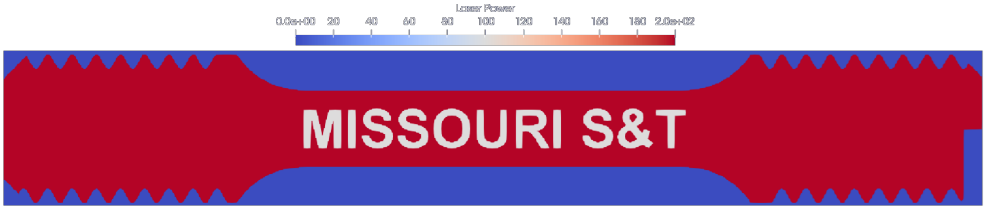

Figure 3: The Laser Power array of Part-1, sliced through the center of the x-axis

2.3 Post Process Product: X-Ray Computed Tomography Imaging

Post-process XCT inspection of both test pieces was performed by NSI using a custom radiograph tool. NSI scanned the test piece with a MeVX6™ High Energy system using a 6 MeV linear accelerator (LINAC) at 450 kV and performed 360º step scan imaging with a focal spot size of 24.15 µm. The full geometry of the tensile piece was captured in 291 x 281 x 1706 voxels. Each voxel is a cube with dimensions 29.96 µm in X, Y, and Z.

The three-dimensional XCT arrays were clustered into two classes using Otsu’s threshold selection method [15]. Otsu’s method is an unsupervised machine learning clustering technique that searches for a threshold that minimizes the intra-class variance across the dataset [16]. These two classes form the basis of the supervised machine learning training output used in section 3.

Figure 4 shows an example of an XCT slice from Part-1, z-slice 929, at approximately the beginning of the letter “U” in the center section of induced defects. The raw XCT image at this layer is shown (a) along with the image of the same layer in the analysis tool (b). The results of the clustering are shown with class 0 represented in blue and class 1 represented in red (c).

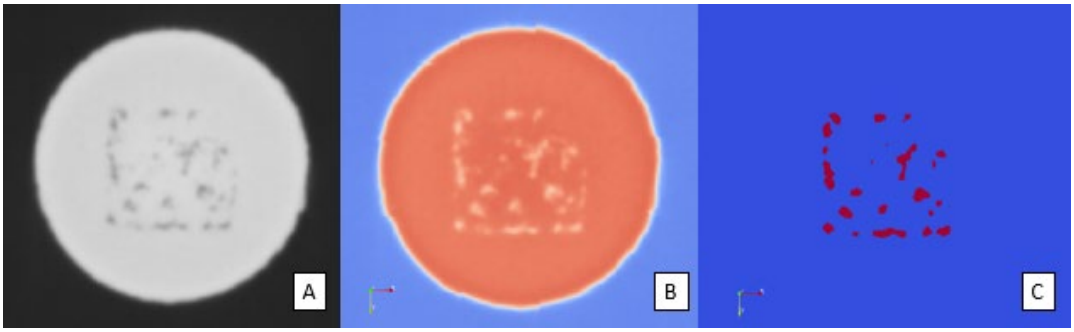

Figure 4: Part 1, z-slice 929, shown for the raw XCT image (A), the analysis tool (B), and the clustered array (C)

2.4 Fusion of the Build Fingerprint

DREAM.3D, an open-source tool kit that allows for construction of customized workflows made up of individual filters to process and analyze data [17], is used in this study to pre-process the datasets prior to machine learning and fuse the part fingerprints into a single hierarchical matrix comprised of each array. Workflows were created to perform the XCT clustering outlined in section 2.3, define the part boundary of each dataset, and register and fuse all datasets together into one fingerprint with a common grid. The supervised machine learning model outlined in section 3 was trained and tested on this final fused hierarchical matrix for both test pieces.

The Robust Automatic Threshold Selection (RATS) algorithm, an unsupervised automated edge detector [18], was used to define the part boundary of each dataset and zero out the values outside the part. This serves two purposes: to clean the datasets prior to machine learning and to allow for creation of a boundary point cloud for registration. RATS was implemented in DREAM.3D with the *Robust Automatic Threshold* filter. Figure 5 shows the time above threshold array for Part-1 before (a) and after (b) RATS is applied to remove values outside the part boundary.

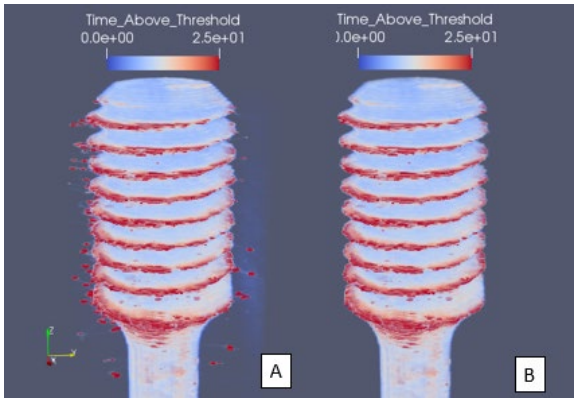

Figure 5: The time above threshold array for part 1 before (A) and after (B) values outside the part boundary are removed

Registration was performed by creating a boundary point cloud of each array and applying an affine transformation matrix with the Iterative Closest Point (ICP) technique [19]. ICP was implemented in DREAM.3D with the *Iterative Closest Point* filter and arrays were transformed using the *Apply Transformation to Geometry* filter. Finally, the datasets were fused to a common grid with the *Fuse Regular Grids* which sampled the arrays using nearest neighbor interpolation [20]. Figure 6 demonstrates the registration process on two arrays for Part-1 where the arrays are initially misaligned (a), boundary point clouds are created (b), and the arrays are transformed to the same affine space (c).

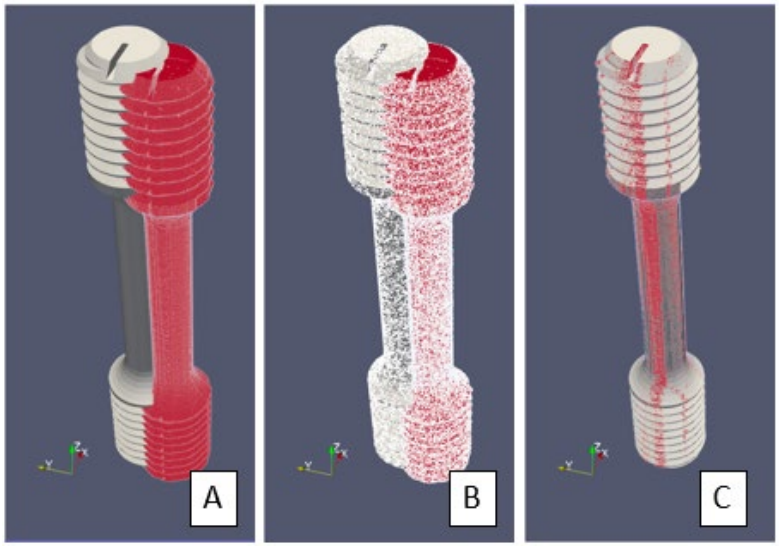

Figure 6: Registration of two arrays using Iterative Closest Point

The machine parameter, laser power, and in situ signal, SWIR, arrays were registered and transformed to the post process, XCT, arrays. All arrays were sampled to the resolution of the SWIR arrays (125 µm x 125 µm x 50 µm) resulting in one common resolution for each part fingerprint. Registration accuracy was found for both test pieces by point-to-plane Mean Average Error (MAE) [20] by measuring individual errors of points on the SWIR and laser power part boundaries to a triangular surface plane of the XCT array. Table 1, below, shows the MAE registration accuracy and standard deviation of both transformed arrays on each test piece.

Table 1: Registration accuracy of IR and Laser Power arrays to XCT array

		MAE (µm)	St. Dev
Part-1	IR	64.94	54.4
	Power	73.2	79.4
Part-2	IR	72.4	68.4
	Power	72.2	70.5

The hypotenuse of the voxels is $(125x125x50)^{1/3} = 92.1$ µm. In all cases, the registration is accurate to within a voxel size as the MAE is less than the voxel hypotenuse. This registration accuracy, within one voxel, is effective for model development.

3.0 Model Development

This study focuses on internal features of the part fingerprint so data arrays of cylinders inside the test pieces have been extracted. Additionally, 500 µm sections at the top and bottom of the test pieces have been removed from the data. Focusing on the cylinder shape removes surface voxels and gives a constant cross section for the data analysis. Data from the part surface may be analyzed in future studies once a robust model has been developed. Datasets from both test pieces

are then sliced in to three sections, Nominal-A, Center, and Nominal-B, giving six independent datasets for model development, training, and testing. The Center sections have all the volume printed at low power. The Nominal-A and -B sections were both printed at nominal machine parameters and do not include induced defects. Figure 7 shows the original data array for time above threshold (A) and the final cropped cylinder data array with each of the three sections (B) for Part-1.

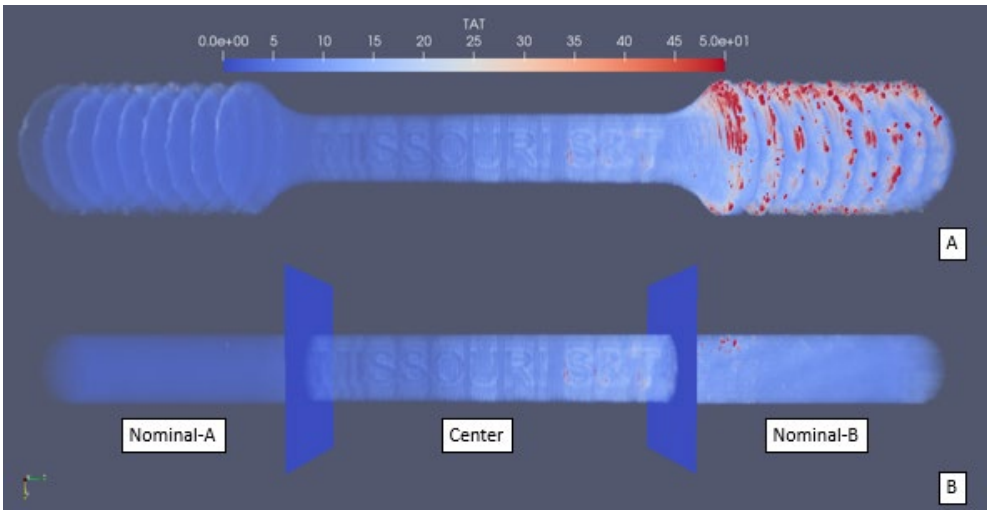

Figure 7: The time above threshold data array for Part-1 (A) and the cropped array with three sections used for model development, training, and testing (B)

Model development and training was performed on the Center section of Part-2. In section 4, the final model was trained on the Center section of Part-2 and tested against the two Nominal sections in Part-2 and all three sections in Part-1. This study shows that a model can be trained on a dataset with induced defects and effectively applied to a separate test piece, including sections without induced defects.

3.1 Classification Metrics

The performance of a classifier can be very high on individual measures but very low on others which may lead to false conclusions. Take, for example, a baseball player that gets one hit in ten at-bats. A classifier would have high accuracy (90%) with a model that guesses the player always strikes out, but the model would be useless if the goal was to predict the batter's successes.

A metric was utilized to measure the predicted array of labels found by the developed classifier against the actual labels of the corresponding array. This metric is based on a 4x4 confusion matrix counting pixels in the predicted array (0 or 1) that are True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). The classification metric found Recall, Precision, f1-score, Accuracy, and Receiver Operating Characteristic - Area Under the Curve (ROC-AUC) score. Each measure in the classification metric is shown below.

$$Accuracy = \frac{TN+TP}{TN+FP+FN+TP} \quad (4)$$

$$Recall = TPR = \frac{TP}{TP+FN} \quad (5)$$

$$Precision = \frac{TP}{TP+FP} \quad (6)$$

$$Fallout = FPR = \frac{FP}{FP+TN} \quad (7)$$

$$f1\text{-}score = \frac{2 \times Recall \times Precision}{Recall + Precision} \quad (8)$$

$$ROC\text{-}AUC = \int_{x=0}^{1} TPR(FPR^{-1}(x)dx \quad (9)$$

Accuracy is simply the percentage of true values in the set. Recall, a synonym for True Positive Rate (TPR), represents a classifier's ability to find all the positive samples. Precision is the ability to not label a sample positive if it is negative. False Positive Rate (FPR) is also known as fallout. F1-score is particularly useful for this study because recall and precision are measured at the same time while the ROC-AUC score represents the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one. All the metrics are important and tracked here; this study focuses on performance on the f1-score because it shows how the model performs overall in predicting the occurrence of Class 1 (the void class).

3.2 Logistic Regression Classification Model

This section outlines the development of the logistic regression classifier with k-Fold cross validation. Logistic Regression was implemented from Sci-kit learn with the lbfgs solver by `sklearn.linear_model.LogsiticRegression` and `sklearn.linear_model.LogisticRegressionCV`. As an optimization problem, the logistic regression implementation minimizes the logistic cost function [21]:

$$\min_{w,c} \frac{1}{2} w^T w \; + \; C \sum_{i=1}^{n} \log\left(\exp\left(-y_i(X_i^T w + c)\right) + 1\right) \quad (10)$$

Where w is the real-valued coefficient to be learned and c is the cost penalty.

In this model, the probabilities of possible outcomes are modeled using a logistic function with equation:

$$f(x) = \frac{L}{1 + e^{-k(x-x_0)}} \quad (11)$$

Where x_0 is the x value of the midpoint, L is the curve's maximum value, and k is the logistic growth rate of the curve. A standard logistic function, also known as a sigmoid function, where L=1, k=1, and x_0=0 is shown in figure 8.

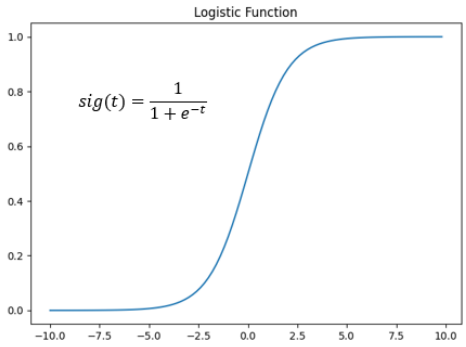

Figure 8: Standard logistic function, also known as the sigmoid function

The logistic regression classifier makes a prediction, given a new x, by estimating the probability that y = 1 on the input x. The output hypothesis function is:

$$h_\theta(x) = \frac{1}{1+ e^{-\theta^T x}} \tag{12}$$

Where:

$$h_\theta(x) = g(\theta^T x) \tag{13}$$

$$g(z) = \frac{1}{1+ e^{-x}} \tag{14}$$

For binary classes, $y \in \{0, 1\}$, the classification threshold is typically $p = 0.5$ where the classifier will:

$$y \in \{0,1\} \begin{cases} predict\ y = 1\ if\ h_\theta(x) \geq 0.5 \\ predict\ y = 0\ if\ h_\theta(x) < 0.5 \end{cases} \tag{15}$$

A method to choose an optimal selection threshold for this model is outlined in section 3.5.

Using Part-2 Center, the logistic regression classifier was trained on a random sample of 70% of the voxels while 30% of voxels were reserved for testing. The model inputs were the three in situ metrics (time above threshold, maximum radiance, and maximum radiance decrease rate) and laser power. The model output labels were 0/1 class labels from the XCT, created in Section 2.3. Class 0 represents nominal density and Class 1 represents void space; Class 1 is the "interesting" samples for this study so precision and recall performance, measured together with f1-score, on Class 1 is of particular interest.

All six datasets have a highly unbalanced class distribution; there are less than 2% of the data points in the two Center sections in Class 1 and approaching 0% in the four Nominal sections. With such an imbalance, a classifier would have very high accuracy simply by predicting '0' for all voxels (better than 98% accuracy on the Center sections and close to 100% accuracy on the Nominal sections); this would be entirely useless because this model needs to predict occurrence of Class 1 voxels while maintaining a relatively low false positive rate.

The Center section of Part-2, which was used for model development in this section, has 325,147 voxels with class distribution of {0: 320,001, 1: 6,146}. A logistic regression model trained on this dataset scored better than 98% accuracy, but the f1-score was 0.025 and the ROC-AUC was 0.51, or only slightly better performance than flipping a coin. This demonstrates the need to prioritize f1-score and ROC-AUC over accuracy in scoring the developed models.

To handle the imbalance, the Synthetic Minority Oversampling Technique (SMOTE) was implemented on Class 1 labels. SMOTE is a method to create synthetic data points of the minority class that has been shown to outperform multiplying the minority class or down sampling the majority class [22]. A range of SMOTE sampling strategies were tested and a 2:1 sampling strategy was shown to perform the best on this dataset. Before SMOTE, the class distribution of the training set was {0: 223,986, 1: 4,316}. With SMOTE, the class distribution of the training set was {0: 223,986, 1: 111,993}. The test set, which held out 30% of the samples {0: 96,015, 1: 1,830}, was not adjusted because model performance on synthetic points is not of interest. Simply by implementing SMOTE, the f1-score was increased to 0.18 and the ROC-AUC increased to 0.81 although the accuracy decreased to 88%. Even though the accuracy decreased, this is an improvement in model performance for this study. Without SMOTE the classifier correctly predicted 24 Class-1 voxels in the test set; with SMOTE the number of True Positives went up to 1,350.

Cross validation was implemented to optimize the hyperparameter value of C in equation (10) for the maximum value of the f1-score. Stratified k-Fold cross validation splits the training set into k equal-size sets and the resulting model is validated on the remaining section of data. This holdout process is repeated k times, and the best performing C value is chosen on an average of test results [23]. Results from the model trained on a range of $n=k$-1 values are shown in Figure 9. Cross validation improves the f1-score only slightly and a final value of n=20 was chosen because both model training time increases, and performance gains decrease as the n value is increased. In addition, ROC-AUC performance, which was not selected as the scoring target, degrades beyond n=20.

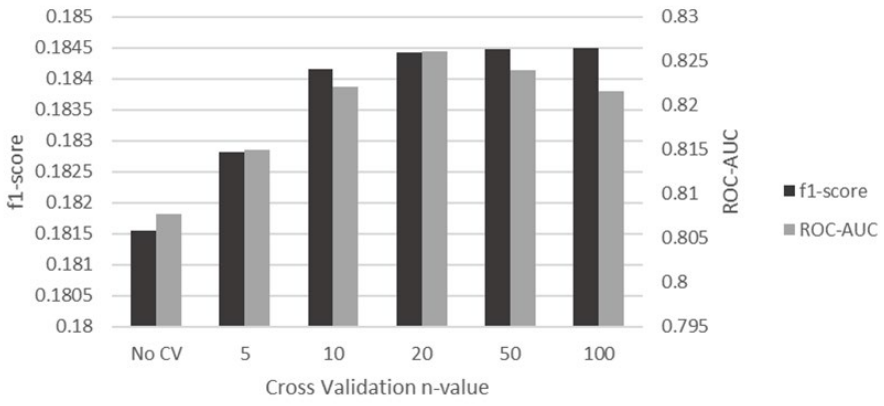

Figure 9: Cross validation performance at increasing n-values

The normalized coefficients of this logistic regression model, trained on the Center section of Part-2 with SMOTE and n=20 k-Fold cross validation, are:

Time above threshold: 3.02e-01 seconds
Maximum radiance: 3.11e-04
Maximum radiance decrease rate: -2.39e-04
Laser Power: -3.25e-02 W

Positive coefficients have a positive correlation with the probability of occurrence of Class 1 label while negative coefficients have a negative correlation with the probability of occurrence of Class 2 label. This means that increased time above threshold, which is the most significant feature in the model, shows an increase in probability of a voxel being in Class 1 (the void class) while the decreased laser power increases the probability of a voxel being in Class 1.

Although the time above threshold and laser power inputs are the most significant, the logistic regression model performed better with the presence of maximum radiance and maximum radiance decrease rate inputs. Figure 10 shows example models trained with different input features. The model, using only the time above threshold, had an f1-score of 0.081 and the model with all three in situ metrics increased f1-score to 0.093. Adding just laser power to time above threshold resulted in an f1-score of 0.177 and including all three in situ metrics with laser power resulted in an f1-score of 0.184. Time above threshold and laser power are certainly the most significant features, but the other two in situ features are still beneficial to model performance.

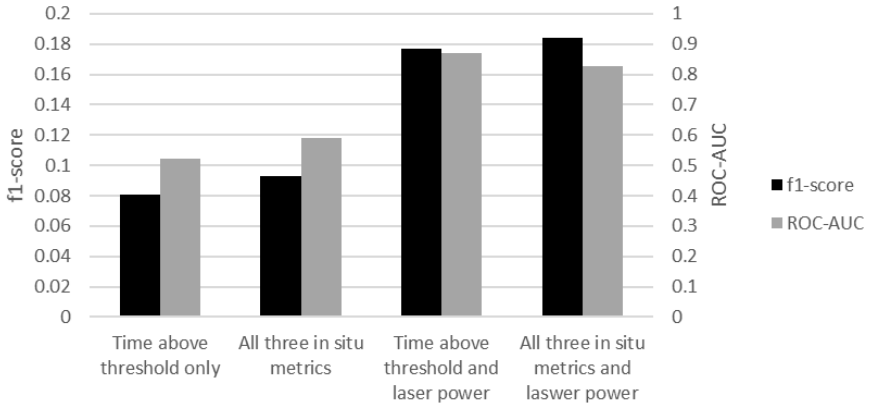

Figure 10: Model performance with different input features

The logistic regression model, on this size dataset and number of input features, did not have trouble converging in a reasonable amount of time (a matter of seconds).

3.3 Other Classifiers

The input and output features of the model from Section 3.2 were used to train five other classifiers. As with the previous section, the dataset used in this section is the Center section of Part-2 with a random sample of 70% of the voxels used for model training and 30% were reserved for testing. SMOTE, with a 2:1 sampling strategy, was used to synthetically oversample Class 1 in the training set. Scikit-learn was used to implement all models in this section.

A neural network multi-layer perceptron was implemented with `sklearn.neural_network.MLPClassifier`. Multi-layer Perceptron (MLP) is a supervised learning

algorithm that can learn a non-linear function approximator for either classification or regression [24]. It is different from logistic regression, in that between the input and the output layer, there can be one or more non-linear layers, called hidden layers. The activation function for the hidden layer used in the MLP was the logistic sigmoid function; other parameters used default settings including stochastic gradient-based log-loss function optimizer and 100 hidden layers.

A Support-vector machine (SVM) classifies training examples by maximizing the width of the gap between two classes [25]. Support vector machines do not directly provide probability estimates, these are calculated using a computationally expensive five-fold cross-validation, so a linear kernel was used to significantly reduce training time. SVC was implemented with `sklearn.svm.LinearSVC` and the 'l1' penalty, leading to sparce coefficient vectors, with default parameters on other settings.

Finally, three ensemble classifiers were used: Random Forest, Gradient Boosting, and AdaBoost. In random forest, decision tree classifiers are fit on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting [26]. In Gradient Boosting (GB), an additive model is built in a forward stage-wise fashion allowing for the optimization of arbitrary differentiable loss functions [27]. Binary GB classification, implemented here, uses only a single regression tree that is fit on the negative gradient of the loss function. An AdaBoost classifier begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases [28]. The implementation of all three ensemble classifiers used default settings in `sklearn.ensemble.RandomForestClassifer`, `sklearn.ensemble.GradientBoostingClassifier`, and `sklearn.ensemble.AdaBoostClassifier`.

Figure 11 shows the f1-score and ROC-AUC performance for all five classifiers discussed in this section along with the Logistic Regression classifier with 20 k-Fold cross validation from Section 3.2. On this dataset, based on the f1-score, the Logistic Regression model performs in the mid-range of all the classifiers. The two ensemble classifiers, Gradient Boosting and Ada Boost, are the highest performing as they returned the highest f1-score.

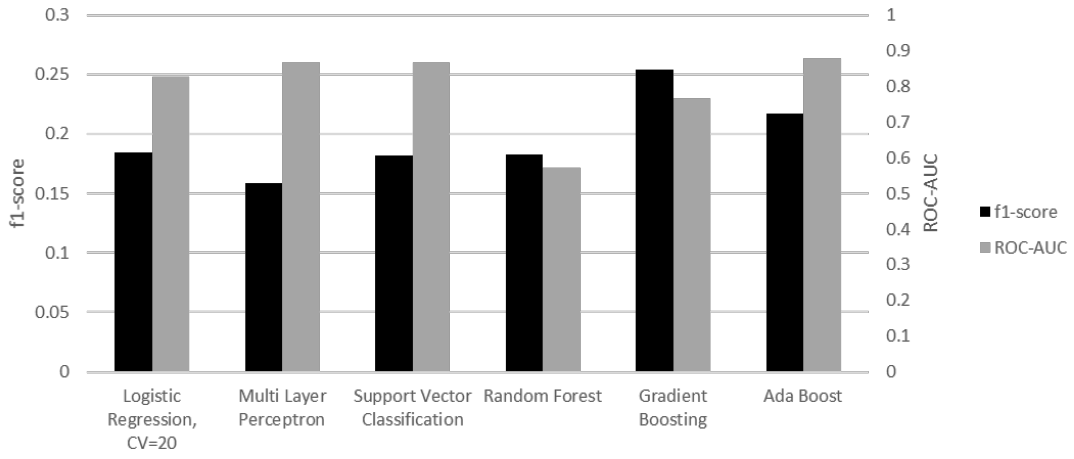

Figure 11: Model performance with different classifiers

3.4 Creation of Synthetic Input Features

This section demonstrates how synthetic features can be created from the original three in situ metrics to improve model performance. Synthetic features have been created in DREAM.3D using the Insight Toolkit (ITK), an open-source toolkit developed by the National Library of Medicine and several partners to support the creation of a public resource in high-dimension data processing tools and image analysis [29].

A total of 36 synthetic in situ features were created using ten filters listed in Table 2. Nine median arrays were created, three from each of the three in situ metrics. To create the median arrays, a given voxel was assigned the median value of the neighboring voxels at a median of 1, 2, and 3 voxels. 27 other synthetic arrays were created in similar fashion with the ITK filters in DREAM.3D, nine from each of the three in situ metrics. Detailed explanation of each filter can be found in the filter documentation at simpleitk.org and dream3d.io.

Table 2: Filters used to create 36 synthetic in situ features

ITK::Binomial Blur Image Filter	ITK::H Maxima Image Filter
ITK::Black Top Hat Image Filter	ITK::H Minima Image Filter
ITK::Bounded Reciprocal Image Filter	ITK::Log10 Image Filter
ITK::Box Mean Image Filter	ITK::Median Image Filter
ITK::H Convex Image Filter	ITK::White Top Hat Image Filter

Each of the six models were retrained on the same training dataset from sections 3.2 and 3.3 with 40 feature inputs in the model (three in situ metrics, laser power, and 36 synthetic features developed in this section) instead of just the four initial feature inputs. The other model features (training/test size, SMOTE, output labels, and model parameters) were kept the same. Results of each model are shown in Figure 12 and can be compared to results of models trained with four feature inputs in Figure 11. Including the synthetic in situ features increased the f1 performance of all six models except the MLP. ROC-AUC performance increased for the Logistic Regression, SVC, and Random Forest classifiers but not the MLP, Gradient Boosting, or Ada Boost.

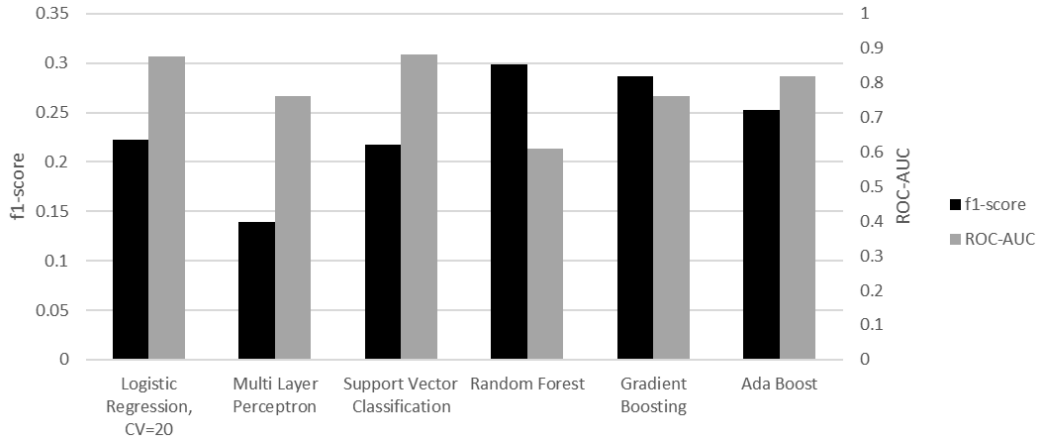

Figure 12: Model performance with different classifiers using 40 input features, including the 36 additional synthetic input features

The normalized coefficients for the logistic regression model are show in Figure 13. As was the case in Section 3.2, the time above threshold metric (labeled "TAT" in the figure), the associated synthetic features and laser power are the most significant features of the model. Testing a range of models trained on a variety of synthetic features showed that other two in situ metrics (labeled Tp and CR in the figure) did benefit the model just like in Section 3.2 The presence of all 40 of these features had the best model performance based on the f1-score.

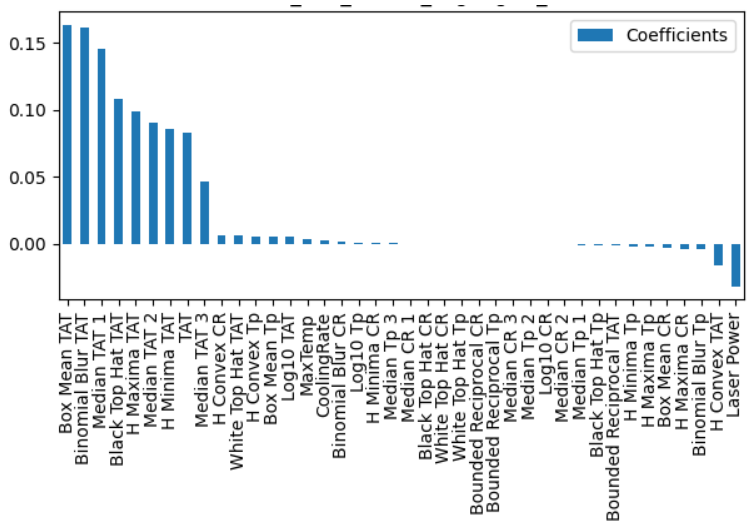

Figure 13: Normalized coefficients of the logistic regression model with n = 20 k-Fold cross validation and 40 input features, including the 36 additional synthetic input features

3.5 Choosing an Optimal Selection Criterion

In logistic regression, the classification threshold, p, can be selected based on the needs of the specific application. The datasets in this study are severely imbalanced, so model classification can be improved by adjusting the classification threshold. In this section an optimal classification threshold is selected, utilizing false positive and false negative rates, and the logistic regression model performance is improved.

Modifying equation (15) for a general classification threshold, p:

$$y \in \{0,1\} \begin{cases} predict\ y = 1\ if\ h_\theta(x) \geq p \\ predict\ y = 0\ if\ h_\theta(x) < p \end{cases} \tag{16}$$

The confusion matrix for the logistic regression model, from Section 3.5, is shown in Figure 15 (A). The model has relatively high false positives with a lower number of false negatives. Essentially, the model is over labeling predictions to correctly label a large percentage (86%) of the Class 1 voxels.

To reduce this over labeling, the selection threshold, p, can be increased which will reduce the false positives. This will also increase the false negatives so there is a tradeoff to find the optimal p. The TPR and the FPR graphed at different selection thresholds, $0 < p < 1$, is shown in

Figure 14 (a). The figure clearly shows the tradeoff between false positives and false negatives predicted in the model. A selection threshold of $p = 0.85$ was found by optimizing FPR and FNR for f1-score and is represented at the green line in Figure 14 (a).

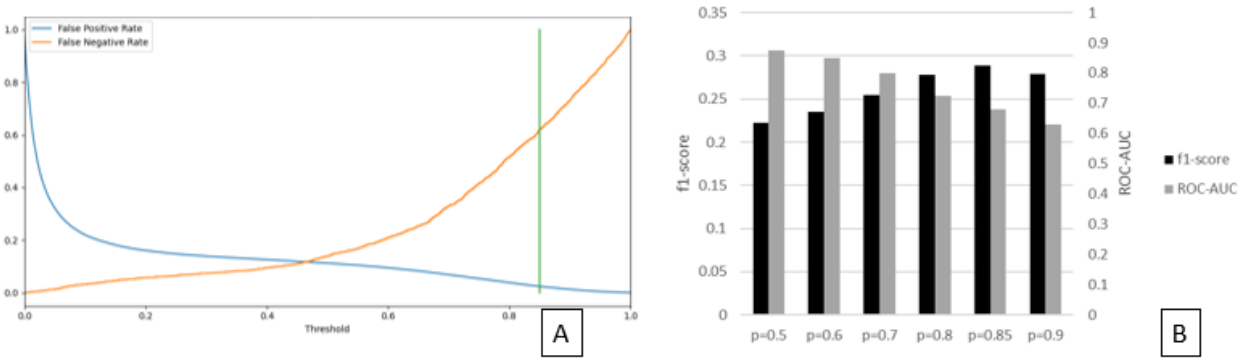

Figure 14: False Positive Rate and False Negative Rate plotted at different selection thresholds (A) and model results at different selection thresholds (B)

Using the optimal selection threshold, $p = 0.85$, the model's performance, based on the f1-score, was increased from 0.22 to 0.29. Figure 14 (b) also shows that the ROC-AUC performance degrades as p is increased. The confusion matrix for both $p = 0.50$ and $p = 0.85$ is shown in Figure 15. Both false positives and true positives decreased with the increase in p and false negatives increased. The model, with the selection threshold optimized for maximum f1-score, correctly labeled 56% less Class 1 voxels.

	Predicted 0	Predicted 1
Actual 0	TN	FP
Actual 1	FN	TP

A

	Predicted 0	Predicted 1
Actual 0	85,190	10,825
Actual 1	248	1,582

$p = 0.50$

B

	Predicted 0	Predicted 1
Actual 0	93,703	2,312
Actual 1	1,131	699

$p = 0.85$

Figure 15: Confusion matrix for the logistic regression model, trained on the Center section of Part-2, with $n = 20$ k-Fold cross validation and 40 input features. The model results with $p = 0.50$ (A) and the model results with $p = 0.85$ (B)

4.0 Predictive Model

In this section, the logistic regression model developed in Section 3 is trained on the full dataset from the Center section of Part-2. There is no need for a train/test split where the model is applied and tested against other datasets. The trained model from Part-2 Center is applied to the Nominal sections in Part-2, as well as Center and Nominal sections in Part-1, to generate prediction arrays for all five test sections. Performance will be measured by comparing each prediction array to its respective output labels as described in Section 3.0. The four Nominal sections are printed at nominal laser power (200W) and voids were not intentionally induced by any means. The two

Center sections had a section of lettering printed at low laser power (100W) to intentionally induce defects.

Figure 16 demonstrates each of the six arrays used for model training and testing in this section. For each array, the time above threshold is shown in blue, the Class 1 output labels (the classified XCT arrays) are shown in red, and the class distribution of the output labels is at the bottom.

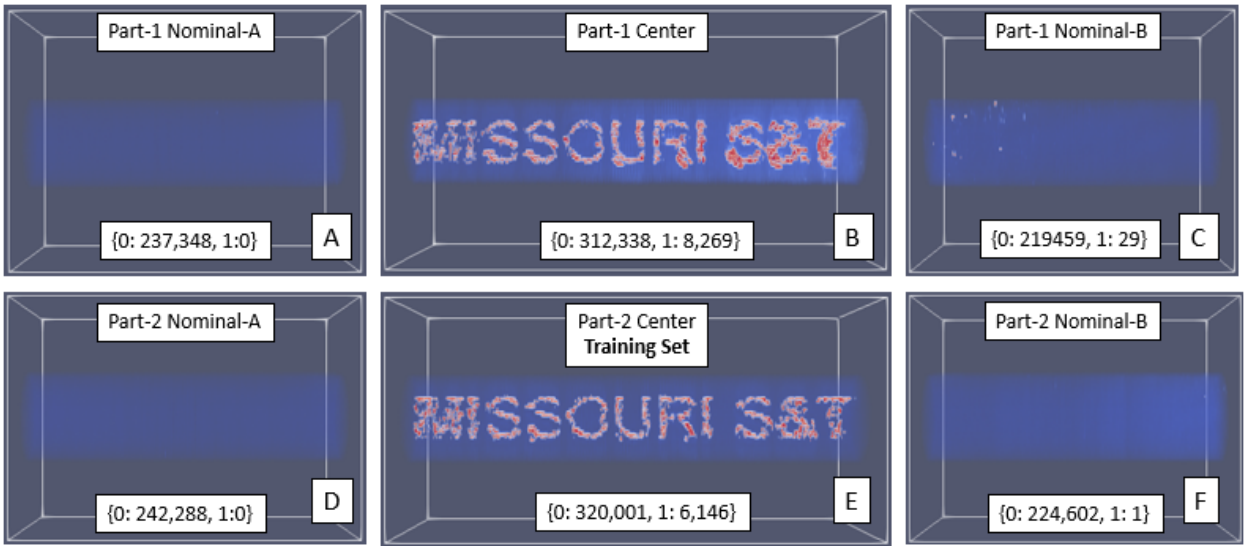

Figure 16: The six independent data arrays used for this section: Part-1 Nominal-A (A), Part-1 Center (B), Part-1 Nominal-B (C), Part-2 Nominal-A (D), Part-2 Center (E), Part-2 Nominal-B (F). Class distribution sizes for each array are included.

Both Nominal-A arrays do not have any Class 1 voxels and the Part-2 Nominal-B has one Class 1 voxel. Part-1 Nominal-B has 29 that are loosely located within two centimeters of each other. Large quantities of Class 1 voxels are evident in both center sections where the part was printed at low laser power. Section 4 shows that the model can be trained on one array, Part-2 Center, and perform similarly on a separate test piece with similar geometry, Part-1 Center. Section 4 also shows that the model can regionally predict the occurrence of Class 1 voxels in the Nominal arrays, but the voxel-by-voxel performance Nominal arrays is extremely low.

4.1 Apply Trained Model to Separate Test Piece

The in situ and laser power input features from the fingerprint of Part-1 Center were input into the models trained with data from Part-2 Center. Each model returns an array of prediction labels (Class 1 or Class 2) that can be tested against the XCT labels for Part-1 Center providing a performance score for each model. The performance, as indicated by f1-score, for the highest performing models outlined in Section 3 (Logistic Regression, Random Forest, and Gradient Boosting) is shown in Figure 17 showing the model performance on the Center section of both parts. The results shown for Part-2 Center have a 70/30 train/test split and the results shown for Part-1 Center were trained on 100% of Part-2 Center and tested on 100% of Part-1 Center.

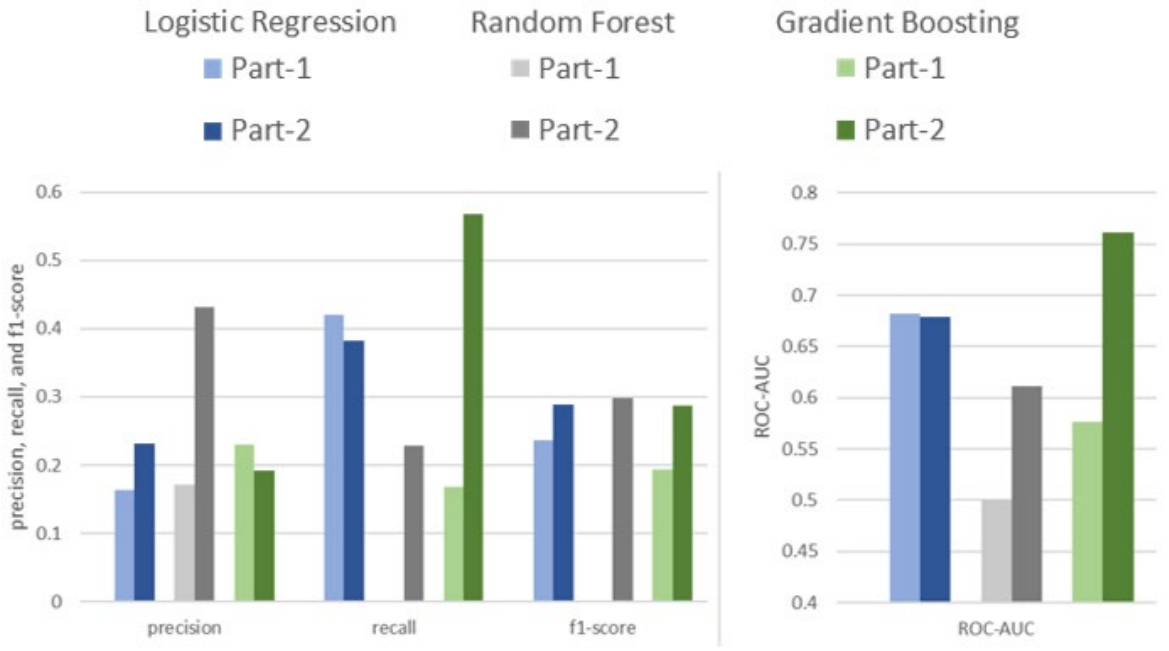

Figure 17: Model performance trained on Part-2 Center and applied to the Center section for both Part-1 and Part-2

The performance measures on Part-1 show how effectively the models, trained on Part-2 Center, performed on inputs in Part-1 Center. The Logistic Regression classifier had the best inter-part performance based on both f1-score and ROC-AUC performance. The f1-scores of both the Logistic Regression and the Gradient Boosting models decreased by 18% and 32%, but the Random Forest f1-performance went to nearly-zero across parts.

The Random Forest model had a nearly-zero recall (the ability to find positive samples); this model only identified six Class 1 samples in Part-1 Center out 8,269. Precision, the ability to not label a sample positive if it is negative, also decreased by more than half when the Random Forest model was applied to Part-1 Center. This resulted in a nearly-zero f1-score and ROC-AUC performance showing that this Random Forest classifier is not effective at making predictions on separate parts.

The Gradient Boosting model had relatively high recall on Part-2 Center, where over one thousand Class 1 samples were correctly identified in the test set of 1,830. Recall in the Gradient Boosting classifier was much lower when the model was applied to Part-1 Center due to a high number of False Negatives. The high FN count contributed to the lower f1-score and ROC-AUC performance.

4.2 Performance in Nominal Build Volume

All classifiers scored very low, by every metric, in the nominal build volume. This volume of each part was printed at nominal laser power and defects were not intentionally seeded in these sections as they were in the Center sections. As discussed in Section 4.0, the Class 1 distribution

size is very low in the nominal build space; none of the models correctly identified a Class 1 voxel in any of the four nominal build volumes. As in Section 4.1, the models were trained on data from Part-2 Center. Prediction arrays were generated for each of the four nominally printed arrays (Part-1 Nominal-A, Part-1 Nominal-B, Part-2 Nominal-A, and Part-2, Nominal-B) using the in situ and laser power inputs.

Voxel-wise performance was so low in these sections that there were no f1-scores for any section or any model as, for each model, precision was zero and recall was either zero or non-divisible by zero. The most effective way to objectively discuss voxel-wise performance is to interrogate the confusion matrix results of each section.

Table 3: Confusion matrix results in each of the four nominal volumes with the three highest performing models

Test Part	Test Section	Model	TN	FP	FN	TP
Part-1	Nominal-A	Logistic Regression	237343	5	0	0
Part-1	Nominal-B	Logistic Regression	218039	1420	29	0
Part-2	Nominal-A	Logistic Regression	242284	4	0	0
Part-2	Nominal-B	Logistic Regression	224517	85	1	0
Part-1	Nominal-A	Random Forest	237348	0	0	0
Part-1	Nominal-B	Random Forest	219459	0	29	0
Part-2	Nominal-A	Random Forest	242288	0	0	0
Part-2	Nominal-B	Random Forest	224602	0	1	0
Part-1	Nominal-A	Gradient Boosting	237348	0	0	0
Part-1	Nominal-B	Gradient Boosting	219459	0	29	0
Part-2	Nominal-A	Gradient Boosting	242288	0	0	0
Part-2	Nominal-B	Gradient Boosting	224602	0	1	0

Table 3 shows the results of each of the four nominal volumes based on prediction arrays created with three different models. There are 29 Class 1 voxels in Part-1 Nominal-B and 1 Class 1 voxel in Part-2 Nominal-B. These voids were not intentionally seeded and the researchers performing this study do not know why they occurred. The Random Forest and Gradient Boosting models did not have False Positives in these arrays, but the Logistic Regression classifier did. Interestingly, the Logistic Regression model has a high False Positive Rate in both Nominal-B sections.

None of the classifiers correctly identified a Class-1 voxel in the nominal space but the Logistic Regression classifier shows a high number of predictions in the test sections where the Class-1 voxels exist. Three-dimensional representations of the model predictions are shown in the next section to demonstrate that the Logistic Regression classifier is off target and over-guessing but in the correct zone.

4.3 Visualization of Predictions

Three-dimensional representations of the Class-1 voxels for each array were created with DREAM.3D. Figure 18 (a) and (b) show the XCT labeling, outlined in Section 2.3, for both Part-

1 and Part-2, respectively. The prediction labels from the Logistic Regression classifier are shown in (c) and (d) and the prediction labels from the Gradient Boosting classifier are shown in (e) and (f). The inter-part performance of other classifiers, Random Forest and others, was very low so they are not shown. The Center of Part-2 was used to train the models so there were not prediction labels generated for Part-2 Center; this section is greyed-out in (d) and (f).

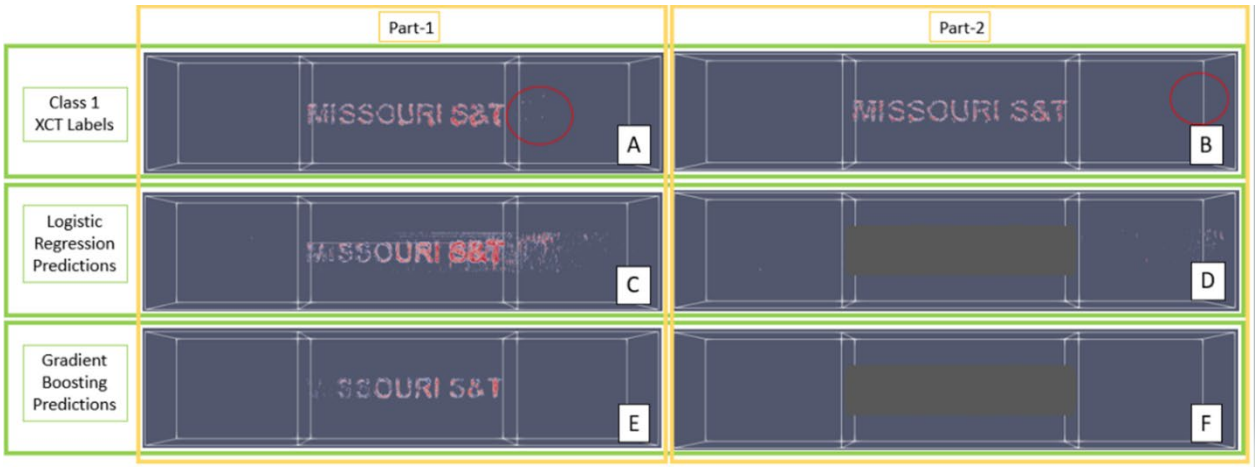

Figure 18: Visualization of Class 1 predictions and labels in all sections for highest performing models. All sections were trained on the output labels from Part-2 Center. Red circles indicate presence of Class-1 voxels in the nominally printed

In the nominally printed sections, there are 29 Class-1 voxels in Part-1 Nominal-B and one Class-1 voxel in Part-2 Nominal-B. The approximate center of these clusters are circled in red in Figure 18 (a) and (b). Neither Nominal-A section, in Part-1 or Part-2, had any Class-1 voxels.

As shown in the confusion matrix in Table 3, the Gradient Boosting classifier did not predict any Class-1 voxels in any of the four nominal sections. This model is not over-labeling predictions in the nominal sections, but it did entirely fail to detect the existence of Class-1 voxels in the nominal sections.

The Logistic Regression model over predicted the occurrence of Class-1 voxels in the nominal sections, but the location of those predictions loosely aligned with the location of the actual XCT labels. Figure 18 (c) shows that most of the 1,420 predictions in Part-1 Nominal-B are clustered to the left and slightly above center which lines up with the cluster of 29 labels seen in Figure 18 (c). Additionally, in Part-2 Nominal-B, where there is only one XCT labeled Class-1 voxel, the 85 predicted Class-1 voxels are seen, in Figure 18 (b), clustered at the far right, in the same region as the actual label.

The two Nominal-A sections did not have any XCT labeled Class-1 voxels, but the Logistic Regression model predicted 5 and 4 voxels in Part-1 and Part-2, respectively. These false positives are spread, seemingly randomly, across the regions and not clustered in central areas of the part volume. In addition to the location spread of the Class-1 predictions, the count of predictions appears correlated with the count of true labels in the nominal space.

5.0 Discussion of Results and Conclusions

This study demonstrated that a logistic regression machine learning model, trained on one test piece, can predict anomalies on a separate test piece. Voxel-wise precision and recall were low, especially in volumes of the part printed at nominal conditions, where defects were not seeded. This result shows that the model, and data used to train it, is not particularly effective at predicting specifically which voxels will be anomalous. The three-dimensional locations of predictions did, however, show that the logistic regression model, with this input data, could predict the occurrence of anomalies in regional clusters. This is a subjective observation as the metrics used were based on voxel-wise performance, not regional clustering. There is opportunity for future models to be optimized by incorporating a metric suite based on regional cluster performance.

This study outlined four dimensions of a build fingerprint: measured in situ signals, machine parameters, process modeling, and post-process product evaluation. Three of the four dimensions of a build fingerprint were represented in this study and used for machine learning with SWIR imaging in situ, laser power, and product labels generated with XCT imaging. As additional model inputs were used for training, including a suite of synthetic in situ features, the performance of the model improved. Incorporation of additional elements of the part fingerprint into model training may further improve performance.

This dataset, along with many others in this area of study, is highly unbalanced. The minority class in the training data was supplemented with synthetic Class-1 labels. The final logistic regression model used cross validation with n=20 k-folds, and an optimal selection threshold of p=0.85 was found. Based on objective voxel-wise performance and subjective analysis of prediction location, the logistic regression classifier outperformed other common machine learning models like a multi-layer perceptron, support vector classifier, and three different boosted classifiers.

There were only two test pieces used for this study. The logistic regression classifier, trained on an area with a large set of induced defects, was a reliable indicator of the occurrence of void space in sections of the other part. Further work is needed to demonstrate the scalability of model transfer across parts. This is a step to show that, to predict defects in a part where there should not be defects, a model may be trained on a separate part with a high number of defects.

The results of this study show an over prediction of Class 1 voxels in nominally printed space, but those predictions are clustered around the true labels. The method developed here, with its recall and precision deficiencies, effectively flags the occurrence of void space. With further work to demonstrate the applicability on a larger scale, this method may be used to highlight where voids are likely to occur or flag if a printing process is not performing as expected. The results can also be used to guide further analysis on regions of interest in the printed part, based on predictive labels, with either destructive or non-destructive investigation.

6.0 Acknowledgements

This work was funded by the Center for Aerospace Manufacturing Technologies and The Boeing Company. The author is particularly grateful to Jeremey Eagan, at The Boeing Company, for general guidance on machine learning applications.

7.0 References

[1] ISO / ASTM52900-15, Standard Terminology for Additive Manufacturing – General Principles – Terminology, ASTM International, West Conshohocken, PA, 2015, www.astm.org

[2] C. Wang, X.P. Tan, S.B. Tor, C.S. Lim, Machine learning in additive manufacturing: State-of-the-art and perspectives, Additive Manufacturing, 2020.

[3] Meng, L., McWilliams, B., Jarosinski, W. et al. Machine Learning in Additive Manufacturing: A Review. JOM 72, 2363–2377 (2020). https://doi.org/10.1007/s11837-020-04155-y

[4] S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, 3rd ed. Pearson Education Inc., 2010.

[5] Pedregosa et al., Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825-230, 2011.

[6] P. Wang, Y.Yang, N. S. Moghaddam, Process modeling in laser powder bed fusion towards defect detection and quality control via machine learning: The state-of-the-art and research challenges, Journal of Manufacturing Processes, Volume 73, 2022, Pages 961-984, ISSN 1526-6125, https://doi.org/10.1016/j.jmapro.2021.11.037

[7] C. S. Lough, T. Liu, X. Wang, B. Brown, R. Landers, D. Bristow, J. Drallmeier, E. Kinzel, Local prediction of Laser Powder Bed Fusion porosity by short-wave infrared imaging thermal feature porosity probabiliy maps

[8] Z. Snow, B. Diehl, E. Reutzel, A. Nassar, Toward in-situ flaw detection in laser powder bed fusion additive manufacturing through layerwise imagery and machine learning, Journal of Manufacturing Systems, Volume 59, 2021, p12-26, ISSN 0278-6125, https://doi.org/10.1016/j.jmsy.2021.01.008.

[9] G. Mohr, S.J. Altenburg, A. Ulbricht, P. Heinrich, D. Baum, C. Maierhofer, K. Hilgenberg, In-Situ Defect Detection in Laser Powder Bed Fusion by Using Thermography and Optical Tomography—Comparison to Computed Tomography. Metals 2020, 10, 103. https://doi.org/10.3390/met10010103

[10] J. Mitchell, T. Ivanoff, D. Dagel, J. Madison, B. Jared, Linking pyrometry to porosity in additively manufactured metals, Additive Manufacturing, Volume 31, 2020, 100946, ISSN 2214-8604, https://doi.org/10.1016/j.addma.2019.100946.

[11] J.B. Forien, N. Calta, P. DePond, G. Guss, T. Roehling, M. Matthews, Detecting keyhole pore defects and monitoring process signatures during laser powder bed fusion: A correlation between in situ pyrometry and ex situ X-ray radiography, Additive Manufacturing, Volume 35, 2020, 101336, ISSN 2214-8604, https://doi.org/10.1016/j.addma.2020.101336.

[12] M. Grasso *et al.,* In-Situ measurement and monitoring methods for metal powder bed fusion: an updated review, 2021 *Meas. Sci. Technol.* 32 112001

[13] A.S.T.M. E8/E8M, Standard Test Methods for Tension Testing of Metallic Materials, ASTM International, West Conshohocken, PA, 2016.

[14] C.S. Lough, X. Wang, C. Smith, R. Landers, D. Bristow, J. Drallmeier, B. Brown, E. Kinzel, Correlation of SWIR imaging with LPBF 304L stainless steel part properties, Additive Manufacturing, Volume 35, 2020, 101359, ISSN 2214-8604, https://doi.org/10.1016/j.addma.2020.101359.

[15] A. Lang, C. O. Rios, J. Newkirk, R. Landers, J. Castle, D. Bristow, Unsupervised Defect Classification of 2D SEM and 3D X-Ray CT Images from Laser Powder Bed Fusion, Proceedings of Solid Freeform Fabrication Symposium, 2021.

[16] N. Otsu, A Threshold Selection Method from Gray-Level Histograms, IEEE Transactions on Systems, Man, and Cybernetics, vol. 9, no. 1, pp. 62-66, Jan. 1979, doi: 10.1109/TSMC.1979.4310076.

[17] M.A. Groeber, M.A. Jackson, DREAM.3D: A Digital Representation Environment for the Analysis of Microstructure in 3D, Integrating Materials 3, 56–72 (2014). https://doi.org/10.1186/2193-9772-3-5

[18] R. Szeliski, Computer Vision: Algorithms and Applications, 2011, ISBN: 978-1-84882-934-3

[19] P. Besl, N.D. McKay, A Method for Registration of 3-D Shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1992, 14 (2): 239–256. doi:10.1109/34.121791.

[20] A. Lang, C. O. Rios, J. Newkirk, R. Landers, J. Castle, D. Bristow, Image Registration and Matching Error in 2D and 3D for Laser Powder Bed Fusion, Proceedings of Solid Freeform Fabrication Symposium, 2021.

[21] Linear Models, Scikit-Learn, https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression. Accessed 18 June 2022

[22] N. V. Chawla, K. W. Bowyer, L. O.Hall, W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," Journal of artificial intelligence research, 321-357, 2002.

[23] Cross Validation, Scikit-Learn, https://scikit-learn.org/stable/modules/cross_validation.html. Accessed 18 June 2022

[24] Neural Networks Supervised, Scikit-Learn, https://scikit-learn.org/stable/modules/neural_networks_supervised.html. Accessed 18 June 2022

[25] Support Vector Machines, Scitkit-Learn, https://scikit-learn.org/stable/modules/svm.html. Accessed 18 June 2022

[26] Random Forest Classifier, Scikit-Learn, https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. Accessed 18 June 2022

[27] Gradient Boosting, Scikit-Learn, https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html. Accessed 18 June 2022

[28] AdaBoost Classifier, Scikit-Learn, https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html. Accessed 18 June 2022

[29] H. Johnson, M. McCormick, L. Ibanez. The ITK Software Guide: Design and Functionality. Fourth Edition.2015. ISBN: 9781-930934-28-3