2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

Morphological Image Analysis and Feature Extraction for Reasoning with AI-based Defect Detection and Classification Models

Jiajun Zhang
Dept of Computer Science, School of Science
Loughborough University, UK
Email: j.zhang8@lboro.ac.uk

Georgina Cosma*
Dept of Computer Science, School of Science
Loughborough University, UK
Email: g.cosma@lboro.ac.uk
* Corresponding author

Sarah Bugby
Dept of Physics, School of Science
Loughborough University, UK
Email: s.bugby@lboro.ac.uk

Axel Finke
Dept of Mathematical Sciences, School of Science
Loughborough University, UK
Email: a.finke@lboro.ac.uk

Jason Watkins
Railston & Co. Ltd.
Nottingham, UK
Email: jason@railstons.com

***Abstract*—As the use of artificial intelligence (AI) models becomes more prevalent in industries such as engineering and manufacturing, it is essential that these models provide transparent reasoning behind their predictions. This paper proposes the AI-Reasoner, which extracts the morphological characteristics of defects (DefChars) from images and utilises decision trees to reason with the DefChar values. Thereafter, the AI-Reasoner exports visualisations (i.e. charts) and textual explanations to provide insights into outputs made by masked-based defect detection and classification models. It also provides effective mitigation strategies to enhance data pre-processing and overall model performance. The AI-Reasoner was tested on explaining the outputs of an IE Mask R-CNN model using a set of 366 images containing defects. The results demonstrated its effectiveness in explaining the IE Mask R-CNN model's predictions. Overall, the proposed AI-Reasoner provides a solution for improving the performance of AI models in industrial applications that require defect analysis.**

***Index Terms*—Morphological analysis, AI-Reasoner, defect characteristics, explainable AI.**

I. Introduction

As *artificial intelligence (AI)* models become increasingly prevalent in industries such as manufacturing, there is a growing need to explain the reasoning behind *machine learning (ML)* model outputs. Developing a comprehensive understanding of ML model outputs empowers end-users to make informed decisions based on the results [1]. Moreover, this understanding equips developers with vital information to effectively address and overcome any limitations inherent in the model. To address this challenge, researchers have developed a field of study known as *explainable AI (XAI)*. XAI techniques use various strategies to provide interpretations for the AI model and its outputs, allowing non-experts to understand how the models arrive at their predictions. These strategies may involve generating explanations that can be understood by humans, visualising the model's decision-making process, or identifying the key features that the model relies on to make its predictions. By providing explanations of an AI model's outputs, XAI can help to increase the transparency and trustworthiness of the model and enable non-experts to make informed decisions based on the model's predictions. This can ultimately lead to improved safety, quality, and efficiency in industrial applications.

In the application of ML to defect detection [2]–[6] and analysis [7]–[9], there have been some attempts to provide explainability of outputs. Fanci *et al.* [10] used image morphology analysis and human visual characteristics (i.e. area, roundness, ratio, brightness, and texture) to identify image defects, and demonstrated that their method could successfully distinguish defects. Lafor and Peansupap [11] proposed a defect detection system that quantifies defects into a feature list (i.e. region, edges, scale, interest points, and texture) and which achieved 94 % detection accuracy in a tiling defect detection task, and effectively reduced engineers' subjective judgements of aesthetic faults. Dong *et al.* [12] extracted geometry-, texture- and vision-based features from images to perform weld defect analysis using a support vector machine model, showing that defects could be identified with high accuracy over 90 % in a pipeline weld defect detection task. Yan and Gao [13] extracted a set of optical image features, including colour-, texture- and shape-based features, from steel surface defect images and used those for a classification task for an engineering enterprise, and their model's performance reached an average of 98 % accuracy.

One approach to XAI is through the use of saliency map methods [14]–[18], that can explain neural network models by highlighting interesting regions in a defect image. Another

This research is funded through joint funding by the School of Science at Loughborough University with industrial support from Railston & Co Ltd.

978-0-7381-4408-5/23/$31.00 ©2023 IEEE

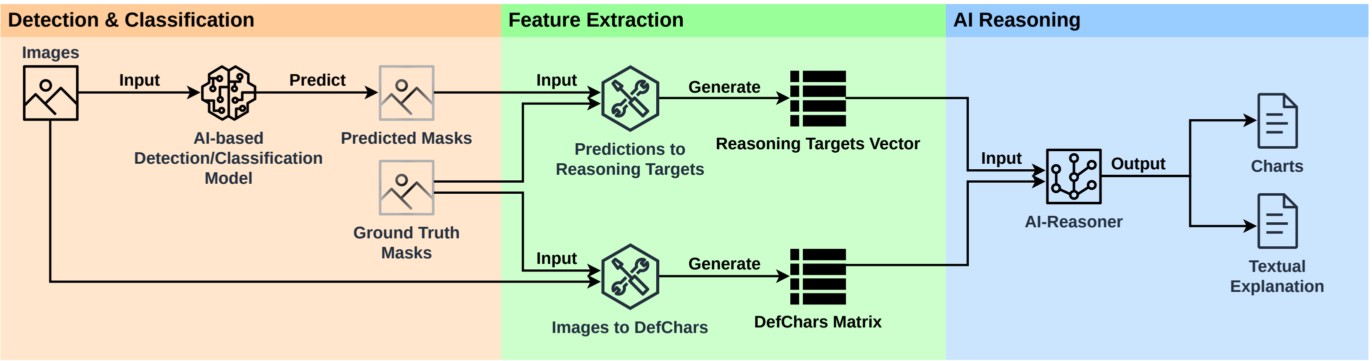


Fig. 1. AI-Reasoner architecture.

approach is through the use of simplification and feature-relevance methods [19]–[21] that can provide ML model explanations by extracting and analysing defect features from images and tabular data. These methods can help identify the most important features that a model relies on to make its predictions, enabling non-experts to better understand how the model derived its decisions.

This paper proposes the AI-Reasoner that facilitates reasoning with AI outputs. The contributions are:

- A new set of *defect characteristics (DefChars)* for describing defects based on their morphological characteristics. DefChar values represent quantitative information about defects, including their colour, shape, and meta aspects. DefChars can be utilised in AI-based models for reasoning and defect analysis tasks.
- A novel AI-Reasoner that extracts DefChars from images and generates a set of charts and textual descriptions to provide visualisation and reasoning with AI outputs.
- Empirical demonstration of the AI-Reasoner on the outputs of a Mask *region-based convolutional neural network (R-CNN)* model using a set of 366 images containing defects. The results demonstrate the usefulness of the proposed DefChars and AI-Reasoner in explaining the model's predictions. The AI-Reasoner offers effective data pre-processing mitigation strategies aimed at enhancing model performance, and which can also save experimental time. Furthermore, this paper discusses the suitability of shapley additive explanations (SHAP) for Reasoning with Defect Predictions.

II. Proposed AI-Reasoner

This section presents the AI-Reasoner's components, as illustrated in Fig. 1.

A. Detection and Classification

A mask-based *deep learning (DL)* model can be utilised to detect or classify defect regions. Such a model can be a Mask R-CNN that takes images and predicts the presence or categories of the defects.

B. Feature Extraction: Predictions to Reasoning Targets

The AI-Reasoner is able to reason the AI predictions for a detection task or a classification task. Let X be an image. Assume that this image shows I defects in pairwise disjoint (i.e. non-overlapping) regions. In other words, $X \setminus (X_1 \cup \cdots \cup X_I)$ is the remaining part of the image which does not contain any defects. Furthermore, let $L_i = L(X_i) \in \{1, \ldots, K\}$ denote the type (category) of the defect in the ith region.

An AI model for defect detection will then output a prediction consisting of J pairs $(\widehat{X}_1, \widehat{L}_1), \ldots, (\widehat{X}_J, \widehat{L}_J)$, where $\widehat{X}_j \subseteq X$ is a predicted defect region and $\widehat{L}_j = \widehat{L}(\widehat{X}_j) \in \{1, \ldots, K\}$ is the predicted type of defect in that region.

Throughout this work, it is assumed that a predicted defect can always be matched to at most one true defect; and that a true defect can always be matched to at most one predicted defect, e.g., via the *intersection over union (IoU)* value.

For the remainder of this work, reasoning targets (i.e. true positives and false negatives) will be defined as follows.

- **Detection.** If only *detection* (but not classification) of defects is of interest, one can consider which of the true defects have been detected (or not) and set $C = (C_1, \ldots, C_I)$ and $D = (D_1, \ldots, D_I)$, where

$$C_i = \begin{cases} 1, & \text{if } X_i \text{ matches some } \widehat{X}_j, \\ 0, & \text{otherwise}, \end{cases} \tag{1}$$

$$D_i = \begin{cases} 1, & \text{if } X_i \text{ does not match any } \widehat{X}_j, \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

  for $i = 1, \ldots, I$.
- **Classification.** If only *classification* (of already known/correctly detected) defects is of interest (in which case $I = J$), one can set $C' = (C'_1, \ldots, C'_I)$ and $D' = (D'_1, \ldots, D'_I)$, where

$$C'_i = \begin{cases} 1, & \text{if } \widehat{L}_i = L_i, \\ 0, & \text{otherwise}, \end{cases} \qquad D'_i = \begin{cases} 1, & \text{if } \widehat{L}_i \neq L_i, \\ 0, & \text{otherwise}, \end{cases}$$

  for $i = 1, \ldots, I$.

- **Joint detection and classification.** If both *detection* and *classification* of defects are of interest, one can specify C_i and D_i as in (1) and (2) and then set

$$C'_i = \begin{cases} 1, & \text{if } C_i = 1 \text{ and } \widehat{L}(X_i) = L_i, \\ 0, & \text{otherwise,} \end{cases}$$

$$D'_i = \begin{cases} 1, & \text{if } C_i = 1 \text{ and } \widehat{L}(X_i) \neq L_i, \\ 0, & \text{otherwise,} \end{cases}$$

for $i = 1, \ldots, I$, where $\widehat{L}(X_i)$ is the predicted category of the ith true defect. Finally, set $C = (C_1, \ldots, C_I)$, $C' = (C'_1, \ldots, C'_I)$, $D = (D_1, \ldots, D_I)$, $D' = (D'_1, \ldots, D'_I)$.

C. Feature Extraction: Images to DefChar

DefChars are a new set of defect characteristics that are extracted from images by analysing the *red, green and blue (RGB)* values and the polygon-shaped defect regions/masks in an image. Table I presents the complete set of DefChars.

Color characteristics: extract colour information using the *hue, saturation, and brightness (HSV)* values, converted from RGB values. HSV values can provide more intuitive colour properties than RGB values. The colour complexity features capture the frequency distribution differences of the HSV values between the defect and background areas.

Shape characteristics: extract shape information from polygon annotations, including bounding boxes, vertices and edges. Shape complexity indicates the shape irregularity of the defect by calculating four values: the edge ratio, follow turning, reverse turning and small turning.

Meta characteristics: 'Defect size' provides an indication of the severity of the defect. 'Distance to the nearest neighbour defect' provides information about a defect's environment.

D. AI Reasoning

The AI-Reasoner uses an ensemble *decision tree (DT)* to reason with the outputs of an AI-based defect detection model. Algorithm 1 introduces the pseudocode for the proposed AI-Reasoner, which comprises: `PlantForest`, `ValidateForest`, `ClimbForest`, `AnalyseForest`, `SummariseForest` and `ExplainForest`.

Empirical evaluations were carried out to determine the optimal parameters for the ensemble DTs. These are the recommended parameters when using the proposed AI-Reasoner.

`PlantForest` takes the DefChar matrix E and reasoning target vector T as inputs where T is the reasoning target C, D, C' or D' (described in Section II-B). Then, 200 DTs are created with the untrimmed setting (i.e. `max depth = -1`: unrestricted tree depth, `min split = 2`: at least two samples in a node when splitting it, and `min leaf = 1`: at least one sample contained in a leaf node) for exploring reasons behind the predictions of an AI-based defect detection model. All trained DTs are stored in a list M for the next step.

`ValidateForest` evaluates the learning performance of each trained DT model M using the metrics (*true positive rate (TPR)* and *true negative rate (TNR)*) defined in Section II-E.

Algorithm 1 AI-Reasoner.

```
Input: DefChar matrix $E$ of size $m \times 38$, where $m$ represents the number of defects in the dataset.
       Reasoning target vector $T = C$ or $T = D$ or $T = C'$ or $T = D'$, where $C$, $D$, $C'$ and $D'$ are as in Section II-B
Output: Reasoning result $R$ (charts and descriptions)

$M \leftarrow$ PlantForest($E$,$T$)
$V \leftarrow$ ValidateForest($M$,$E$,$T$)
print $V$ % defects have been correctly reasoned
$O \leftarrow$ ClimbForest($M$)
$A \leftarrow$ AnalyseForest($O$)
$S \leftarrow$ SummariseForest($A$)
$R \leftarrow$ ExplainForest($S$)
return $R$
```

The input data (DefChar matrix E and reasoning target vector T) are utilised to assess how many defects were correctly learnt by each trained DT. Then, a learning score V provides the overall learning performance of the AI-Reasoner and the equation is shown in (3).

`ClimbForest` parses all trained DTs and recursively reads the nodes in each tree and extracts tree information (i.e. DefChar features, split condition, the sample distributions in the node and its child nodes). The extracted information of every node is stored in a list O for further steps.

`AnalyseForest` analyses the extracted list of DTs' nodes O and computes a set of values to determine the importance of each node in each DT. A *decision score*, DS, is the average split ratio in which the parent node divides the samples into its two child nodes:

$$\text{DS} = \frac{1}{2}\left(\left|\frac{\text{TC}_1 - \text{FC}_1}{N_1}\right| + \left|\frac{\text{TC}_0 - \text{FC}_0}{N_0}\right|\right),$$

where

- N_1 is the number of samples whose reasoning target values are 1 in a node;
- N_0 is the number of samples whose reasoning target values are 0 in a node;
- TC_1 is the number of samples whose reasoning target values are 1 in its true child node;
- TC_0 is the number of samples whose reasoning target values are 0 in its true child node;
- FC_1 is the number of samples whose reasoning target values are 1 in its false child node;
- FC_0 is the number of samples whose reasoning target values are 0 in its false child node.

Furthermore, the *distinguish score*, TS, measures the percentage in which the parent node isolates the samples in its two child nodes:

$$\text{TS} = \left|\frac{\text{TC}_1}{N_1} - \frac{\text{TC}_0}{N_0}\right| = \left|\frac{\text{FC}_1}{N_1} - \frac{\text{FC}_0}{N_0}\right|.$$

TABLE I
PROPOSED DEFCHARS.

Colour Information extracted and stored separately for the defect and background areas		
DefChar Name	**Value Range**	**Description**
Average Hue	$\{0, 1, \dots, 359\}$	Average hue value
Mode of Hue	$\{0, 1, \dots, 359\}$	Most frequent hue value
Unique number of Hue values	$\{1, 2, \dots, 360\}$	Number of unique hue values
Hue Range	$\{0, 1, \dots, 180\}$	Difference of maximum and minimum hue value
Average Saturation	$\{0, 1, \dots, 254\}$	Average saturation value
Mode of Saturation	$\{0, 1, \dots, 254\}$	Most frequent saturation value
Unique number of Saturation	$\{1, 2, \dots, 255\}$	Number of unique saturation values
Saturation Range	$\{0, 1, \dots, 254\}$	Difference of maximum and minimum saturation values
Average Brightness	$\{0, 1, \dots, 254\}$	Average brightness value
Mode of Brightness	$\{0, 1, \dots, 254\}$	Most frequent brightness value
Unique number of Brightness	$\{1, 2, \dots, 255\}$	Unique brightness values
Brightness Range	$\{0, 1, \dots, 254\}$	Difference of maximum and minimum brightness value
Colour Complexity		
DefChar Name	**Value Range**	**Description**
Hue Difference	$[0, 1]$	Hue frequency distribution difference between the defect and background areas
Saturation Difference	$[0, 1]$	Saturation frequency distribution difference between the defect and background areas
Brightness Difference	$[0, 1]$	Brightness frequency distribution difference between the defect and background areas
Shape Information		
DefChar Name	**Value Range**	**Description**
Number of Edges	$\{3, 4, \dots\}$	Number of edges of the defect polygon areas
Coverage	$[0, 1]$	Percentage of the defect polygon area covered by its bounding box
Aspect Ratio	$[0, 1]$	Ratio between the width and height of defect bounding box
Average Turning Angles	$\{1, 2, \dots, 180\}$	Average value of vertex angles of the defect polygon area
Mode of Turning Angle	$\{1, 2, \dots, 180\}$	Value of vertex angles that appears the most often in the defect polygon
Shape Complexity		
DefChar Name	**Value Range**	**Description**
Edge Ratio	$[0, 1]$	Average length ratio between two adjacent edges in the defect polygon area
Followed Turns	$[0, 1]$	Proportion of two adjacent vertices which turn to the same direction in the defect polygon area
Small Turns	$[0, 1]$	Percentage of vertices which are smaller than 90° in the defect polygon area
Reversed Turns	$[0, 1]$	Proportion of two adjacent vertices which turn to a different direction in the defect polygon area
Meta Information		
DefChar Name	**Value Range**	**Description**
Defect Size	$\{1, 2, \dots\}$	Number of pixels in the defect polygon area
Neighbour Distance	$\{0, 1, 2\}$	Categorised distances to the nearest neighbour, 0→Short ($\leq$100 px); 1→Long; 2→No Neighbour.

Finally, *usage of samples*, U, is the percentage of samples contained in a node:

$$U = \frac{N_1 + N_0}{N},$$

where N is the total number of samples, i.e. at the root node.

A set of additional values are computed. The *node importance index*, IDX, is calculated to state a node's capability in distinguishing samples that have different reasoning target values:

$$\mathrm{IDX} = U \times ((1 + \mathrm{DS}) \times \mathrm{TS} + B_{\mathrm{deg}} + B_{\mathrm{sta}}).$$

Decision degree bonus (B_{deg}) and decision status bonus (B_{sta}) are calculated based on decision degree (DEG) and decision status (STA) (shown in Table II), and these two values are used to amplify the IDX value. DEG expresses the sample isolation degree (i.e. empty, weak, middle, strong and full) according to TS. STA expresses the node's splitting status (i.e. confirmation, half reduction, and reduction) according to

TABLE II
INFORMATION ABOUT A NODE'S DECISION DEGREE, STATUS AND BONUS.

DEG			**STA**		
TS threshold	**Degree**	$\mathbf{B_{deg}}$	**Condition**	**Status**	$\mathbf{B_{sta}}$
$\mathrm{TS} = 0$	empty	0	$\mathrm{TC}_1 = 0$ or $\mathrm{TC}_0 = 0$	confirmation	0.5
$0 < \mathrm{TS} < 0.25$	weak	0.1	$\mathrm{TC}_1 = \mathrm{FC}_1$ or $\mathrm{TC}_0 = \mathrm{FC}_0$	half reduction	0.2
$0.25 \leq \mathrm{TS} < 0.5$	middle	0.2	all other conditions	reduction	0
$0.5 \leq \mathrm{TS} < 1$	strong	0.3			
$\mathrm{TS} = 1$	full	0.5			

the values of TC_1, TC_0, FC_1, FC_0. Decision direction (DIR) is a boolean signal that indicates which child node contains more samples that reasoning target values are 1. The output is an extended list A, which contains the list O and these analysed values.

`SummariseForest` computes the importance of each DefChar by summarising the list A. Firstly, all analysed nodes are divided into groups by DefChar; hence 38 groups of nodes are constructed where each node group corresponds to a unique DefChar. Next, DefChar Importance Index (DIS), DefChar Usage Frequency (DUF), DefChar Overall Score (DOS), DefChar Effective Range (DER) are calculated to quantify the importance of each DefChar as follows.

- DIS is calculated by averaging the node importance indices IDX in the grouped nodes; a higher value indicates that the DefChar can easily affect the AI model's detection/classification performance.
- DUF is calculated by averaging the occurrence of the DefChar used in the ensemble DT; a higher value implies that the DefChar is essential to split the reasoning targets.
- DOS is the overall score for each DefChar calculated by multiplying the DIS and DUF with a 3:1 ratio.
- DER is a value range to show the DefChar value interval that can affect the AI model's predictions.

The metrics defined in this section are utilised by `ExplainForest` to create the AI-Reasoner's output (i.e. charts and textual descriptions) for enabling the end-user to reason with a model's outputs. `ExplainForest` visualises the array-format reasoning result S using charts and provides textual descriptions that include mitigation strategies R to help users gain reasons behind the AI results and to take steps in improving the dataset that was used to train the models. Sample outputs are shown in the paper's Github repository[1].

E. Metrics for Evaluating the AI-Reasoner's Learning

TPR and TNR evaluate the learning performance of each DT and are defined as

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \quad \mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}},$$

where

- TP is the total number of defects whose reasoning target values are 1 and were correctly learnt by the DT;
- TN is the total number of defects whose reasoning target values are 0 and were correctly learnt by the DT;
- FP is the total number of defects whose reasoning target values are 1 and were not correctly learnt by DT;
- FN is the total number of defects whose reasoning target values are 0 and were not correctly learnt by DT.

TPR measures the ability of a model to correctly identify positive instances. TNR measures the ability of a model to correctly identify negative instances.

The learning score evaluates the overall learning performance of the reasoning model:

$$\text{Learning Score} = \frac{1}{n} \sum_{i=1}^{n} \frac{\mathrm{TPR}_i + \mathrm{TNR}_i}{2}, \qquad (3)$$

where n is the number (i.e. $n = 200$) of DTs that comprise the reasoning model; i is the index of a DT; TPR_i and TNR_i are the TPR and the TNR of the ith DT.

III. EXPERIMENT METHODOLOGY – UTILISING DEFCHARS TO REASON WITH AI MODELS

The proposed AI-Reasoner is applied to reason with the outputs of the image-enhanced mask R-CNN (IE-MRCNN) model proposed by Zhang *et al.* [22] which can detect the presence and types of wind turbine blade defects. Their paper evaluates the model's defect detection and type classification performance on an augmented dataset of defects (v1), the augmented dataset greyscaled (v2), and the augmented dataset after image enhancement (but not greyscaled) (v3). Their prediction results are shown in Table III.

A. Dataset

The dataset utilised for the experiments was provided by Zhang *et al.* [22]. Each defect is represented as a set of DefChar features stored in a 366 × 38 matrix E. The ground truth label (i.e. region and type) of each defect is stored in a set of pairs (X, L) where each pair (X_i, L_i) corresponds to the ith row of matrix E. The predicted label of each defect is stored in a set of pairs $(\widehat{X}, \widehat{L})$.

[1] https://github.com/edgetrier/AI-Reasoner

TABLE III
OVERVIEW OF PREDICTION RESULTS FROM IMAGE-ENHANCED MASK R-CNN [22].

Prediction Results	**Number (Percentage) of Detected Defects**		
	Augmented	**Greyscaled**	**Image-Enhanced**
Detected C	311 (84.97 %)	298 (81.42 %)	308 (84.15 %)
Undetected D	55 (15.03 %)	68 (18.58 %)	58 (15.85 %)
Correctly Classified C'	296 (80.87 %)	287 (78.42 %)	295 (80.60 %)
Misclassified D'	15 (4.10 %)	11 (3.00 %)	13 (3.55 %)
Total		366 (100.00 %)	

TABLE IV
COMBINATIONS OF DEFCHARS.

Combination	**Color DefChar**	**Shape DefChar**	**Meta DefChar**
Color	✓		
Shape		✓	
Meta			✓
Color-Shape	✓	✓	
All DefChars	✓	✓	✓

B. Methodology

Step 1: Apply the Mask R-CNN model to the dataset containing 366 defects with ground truth labels (i.e. regions X, types L) and obtain predicted labels (i.e. regions $\widehat{X}$, types $\widehat{L}$) (described in Section III-A).
Step 2: Extract the DefChar matrix E (366×38) of the images and rescale the DefChar values using the min-max scaling method (described in Section II-C).
Step 3: Convert and merge each image's ground truth labels (X, L) and predicted labels $(\widehat{X}, \widehat{L})$ of the Mask R-CNN model to four separate reasoning target vectors C, D, C' and D'. Vectors C and D hold the reasoning targets of the correct and incorrect model outputs of the detection task, and vectors C' and D' hold the reasoning targets of the correct and incorrect outputs of the classification task (see Section II-B).
Step 4: Apply the AI-Reasoner (see Section II-D) to the DefChar matrix E and each reasoning target vector $T \in \{C, D, C', D'\}$. Conduct four experiments, each with a different reasoning target: 1) reasoning with the outputs that were correctly detected; 2) reasoning with the outputs that were not detected; 3) reasoning with the outputs whose types were correctly classified; and 4) reasoning with the outputs whose types were not correctly classified.
Step 5: Evaluate the learning performance of the AI-Reasoner across each experiment using the learning score metric described in Section II-E. The learning scores are averaged across four reasoning targets and presented in Table V that also contains the results when tuning the DT with different parameter settings (`max depth`, `min split` and `min leaf`) and using different DefChar combinations (see Table IV).
Step 6: AI-Reasoner interprets the outputs, presents charts, textual explanations, and suggests mitigation strategies to the user for improving prediction performance. These strategies are presented to the end-user in textual format (see Section IV-B). The user can follow the proposed mitigation strategies to improve their dataset and the model's performance.

IV. RESULTS

A. Reasoning Performance when using DefChars

This section describes the learning performance of the AI-Reasoner when using different parameters and combinations of DefChars. The learning scores for each parameter setting and combination are computed by averaging the learning scores across the four reasoning targets (see Section III-B step 4) and are shown in Table V. The highest learning score of each model is marked in bold text. The AI-Reasoner achieves a higher learning score if the DT is set with a deep tree depth, a small number of splits, and a small number of leaves. The learning score gradually increased from 50.31 % (`max depth = 1`, `min split = 2`, and `min leaf = 1`) to 100 % (`max depth = infinity`, `min split = 2`, and `min leaf = 1`) when applying all DefChars. The AI-Reasoner achieved the highest learning score of 100 % when using all DefChars and the DT setting (`max depth = infinity`, `min split = 2`, and `min leaf = 1`); hence, this DefChar combination and DT settings are applied to the DTs that comprise AI-Reasoner.

B. Interpretation of the AI-Reasoner's Outputs

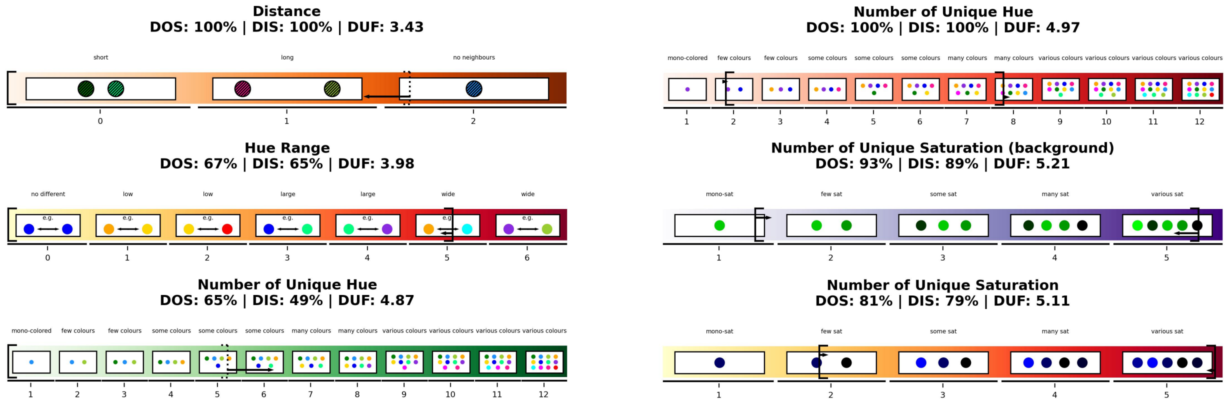
The outputs of AI-Reasoner contain 38 charts in total for a reasoning target. Sample outputs are shown in the project's Github[2] and in Fig. 2. Each chart illustrates the value range (i.e. DER) of a DefChar for a defect that was undetected or misclassified by the IE-MRCNN model. The charts provide the calculated scores (i.e. DIS, DUF and DOS) indicating the importance of a DefChar. These scores are shown at the top of each chart. The reasons for undetected cases include low hue range, neighbouring defects, and low saturation; and the reasons for misclassified cases include low hue range and strong saturation in the background, and narrow hue differences between the inside and outside of the defected area. Based on these findings, the AI-Reasoner suggests the following mitigation strategies: Greyscaling the defects may reduce misclassified but increase the undetected cases. Enhancing the

[2] https://github.com/edgetrier/AI-Reasoner

TABLE V

AVERAGE LEARNING SCORES FOR EACH DEFCHAR COMBINATION ACROSS FOUR REASONING TARGETS (SEE SECTION III-B STEP 4).

Parameters			DefChar Combination				
Max Depth	**Min Split**	**Min Leaf**	**Color**	**Shape**	**Meta**	**Color-Shape**	**All DefChars**
1	2	1	50.23 %	50.00 %	50.09 %	50.20 %	**50.31** %
5	2	1	68.75 %	55.69 %	52.24 %	69.52 %	**71.88** %
10	2	1	91.67 %	75.57 %	71.52 %	93.14 %	**94.72** %
Infinity	2	1	100.00 %	96.60 %	96.60 %	100.00 %	**100.00** %
Infinity	2	3	79.24 %	67.48 %	60.33 %	81.02 %	**84.62** %
Infinity	2	5	69.16 %	56.61 %	53.54 %	70.57 %	**75.88** %
Infinity	5	1	88.84 %	79.45 %	74.61 %	89.06 %	**91.28** %
Infinity	5	3	79.26 %	67.52 %	60.37 %	80.87 %	**84.55** %



(a) Undetected Cases D.

(b) Misclassified Cases D'.

Fig. 2. Top-three reasoning results of undetected and misclassified cases.

images by normalising the colours of the defects may increase the detected cases and reduce the misclassified cases.

Zhang *et al.*'s [22] experiments revealed that using a greyscaled dataset (v2) reduced the misclassified and increased the undetected cases compared to when using the augmented dataset (v1); and when using the image-enhanced dataset (v3), the detected cases were slightly decreased but the misclassified cases were reduced (see Table III). These results are mostly consistent with the mitigation strategies proposed by the AI-Reasoner.

C. Discussion on the suitability of SHAP for Reasoning with Defect Predictions

This section explains the outputs of a pre-trained CNN model using the SHAP algorithm [23], an XAI technique for reasoning with model outputs. SHAP cannot support object-detection models and hence a simple CNN model was utilised for the type classification task instead of Zhang et al.'s [22] IE-MRCNN model. Fig. 3 shows SHAP's output when given four defect images. The important regions are highlighted with blue and red colours based on the analysis of the hidden layers of the CNN. The main observations when comparing SHAP with the proposed AI-Reasoner for defect detection, SHAP is not compatible with masked-based models that provide multi-channel outputs (i.e. matrices that indicate the presence or absence of specific objects or regions within an image) or

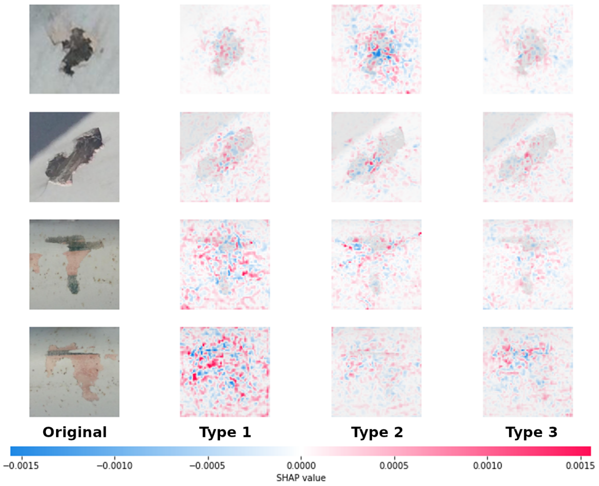


Fig. 3. SHAP output of four defect images. Important regions are highlighted with blue and red based on the analysis of the hidden layers of the CNN.

have complex-structured designs (e.g. YOLO, Faster R-CNN, and Mask R-CNN). Whereas, the AI-Reasoner is compatible with masked-based models. SHAP does not provide details about a defect's characteristics that lead to its misclassification, whereas the AI-Reasoner captures that information in the form of DefChar charts and provides reasoning in the form of charts and textual explanations with mitigation strategies for improving the performance of the model.

V. Conclusion

This paper proposes DefChars and an AI-Reasoner that extracts DefChars from images and utilises DTs to reason with AI outputs. The AI-Reasoner provides the end-user with charts, textual explanations and recommendations on improving the dataset pre-processing in order to improve a model's performance. Future work includes experiments with additional datasets, considering other prediction errors (e.g. false positive, duplicated predictions, etc.), DefChar applications, and exploring the AI-Reasoner's capability in saving experimental time via the mitigation strategies it provides.

Acknowledgment

The authors acknowledge the expert guidance and datasets provided by Jason Watkins, Chris Gibson, and Andrew Rattray of *Railston & Co Ltd*.

References

[1] I. Ahmed, G. Jeon, and F. Piccialli, "From Artificial Intelligence to Explainable Artificial Intelligence in Industry 4.0: A Survey on What, How, and Where," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 8, pp. 5031–5042, 2022.

[2] Y. Cheng, D. HongGui, and F. YuXin, "Effects of Faster Region-based Convolutional Neural Network on the Detection Efficiency of Rail Defects under Machine Vision," in *2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*, 2020, pp. 1377–1380.

[3] J. Lian, W. Jia, M. Zareapoor, Y. Zheng, R. Luo, D. K. Jain, and N. Kumar, "Deep-Learning-Based Small Surface Defect Detection via an Exaggerated Local Variation-Based Generative Adversarial Network," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 2, pp. 1343–1351, 2020.

[4] H. Dong, K. Song, Y. He, J. Xu, Y. Yan, and Q. Meng, "PGA-Net: Pyramid Feature Fusion and Global Context Attention Network for Automated Surface Defect Detection," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 12, pp. 7448–7458, 2020.

[5] B. Yang, Z. Liu, G. Duan, and J. Tan, "Mask2Defect: A Prior Knowledge-Based Data Augmentation Method for Metal Surface Defect Inspection," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 10, pp. 6743–6755, 2022.

[6] X. Ni, Z. Ma, J. Liu, B. Shi, and H. Liu, "Attention Network for Rail Surface Defect Detection via Consistency of Intersection-over-Union(IoU)-Guided Center-Point Estimation," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, pp. 1694–1705, 2022.

[7] G. Acciani, G. Brunetti, and G. Fornarelli, "Application of neural networks in optical inspection and classification of solder joints in surface mount technology," *IEEE Transactions on Industrial Informatics*, vol. 2, no. 3, pp. 200–209, 2006.

[8] C. Phua and L. B. Theng, "Semiconductor Wafer Surface: Automatic Defect Classification with Deep CNN," in *2020 IEEE REGION 10 CONFERENCE (TENCON)*, 2020, pp. 714–719.

[9] L. Wen, Y. Wang, and X. Li, "A New Cycle-consistent Adversarial Networks With Attention Mechanism for Surface Defect Classification With Small Samples," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 12, pp. 8988–8998, 2022.

[10] G. Fanci, Z. Chune, and X. Ke, "Image defect inspection based on human visual characteristics," *IET Conference Proceedings*, pp. 377–380(3), January 2011. [Online]. Available: https://digital-library.theiet.org/content/conferences/10.1049/cp.2011.1028

[11] C. Laofor and V. Peansupap, "Defect detection and quantification system to support subjective visual quality inspection via a digital image processing: A tiling work case study," *Automation in Construction*, vol. 24, pp. 160–174, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0926580512000271

[12] S. Dong, X. Sun, S. Xie, and M. Wang, "Automatic defect identification technology of digital image of pipeline weld," *Natural Gas Industry B*, vol. 6, no. 4, pp. 399–403, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352854019300749

[13] X. Yan and L. Gao, "A feature extraction and classification algorithm based on improved sparse auto-encoder for round steel surface defects," *Mathematical Biosciences and Engineering*, vol. 17, no. 5, pp. 5369–5394, 2020. [Online]. Available: https://www.aimspress.com/article/doi/10.3934/mbe.2020290

[14] M. Lee, J. Jeon, and H. Lee, "Explainable AI for domain experts: a post Hoc analysis of deep learning for defect classification of TFT–LCD panels," *Journal of Intelligent Manufacturing*, 2021, publisher Copyright: © 2021, The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature.

[15] J. Lorentz, T. Hartmann, A. Moawad, F. Fouquet, and D. Aouada, "Explaining Defect Detection with Saliency Maps," in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems.* Springer, 2021, pp. 506–518.

[16] V. Bento, M. Kohler, P. Diaz, L. Mendoza, and M. A. Pacheco, "Improving deep learning performance by using Explainable Artificial Intelligence (XAI) approaches," *Discover Artificial Intelligence*, vol. 1, no. 1, pp. 1–11, 2021.

[17] R.-K. Sheu, L.-C. Chen, M. S. Pardeshi, K.-C. Pai, and C.-Y. Chen, "AI Landing for Sheet Metal-Based Drawer Box Defect Detection Using Deep Learning (ALDB-DL)," *Processes*, vol. 9, no. 5, 2021. [Online]. Available: https://www.mdpi.com/2227-9717/9/5/768

[18] C. Seiffer, H. Ziekow, U. Schreier, and A. Gerling, "Detection of Concept Drift in Manufacturing Data with SHAP Values to Improve Error Prediction," in *DATA ANALYTICS 2021: The Tenth International Conference on Data Analytics, October 3-7, 2021, Barcelona, Spain*, 2021, pp. 51–60.

[19] G. Kolappan Geetha and S.-H. Sim, "Fast identification of concrete cracks using 1D deep learning and explainable artificial intelligence-based analysis," *Automation in Construction*, vol. 143, p. 104572, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0926580522004423

[20] S. Zhou, H. Liu, K. Cui, and Z. Hao, "JCS: An Explainable Surface Defects Detection Method for Steel Sheet by Joint Classification and Segmentation," *IEEE Access*, vol. PP, pp. 1–1, 10 2021.

[21] A. S. Barnard, "Explainable prediction of N-V-related defects in nanodiamond using neural networks and Shapley values," *Cell Reports Physical Science*, vol. 3, no. 1, p. 100696, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666386421004215

[22] J. Zhang, G. Cosma, and J. Watkins, "Image Enhanced Mask R-CNN: A Deep Learning Pipeline with New Evaluation Measures for Wind Turbine Blade Defect Detection and Classification," *Journal of Imaging*, vol. 7, no. 3, 2021. [Online]. Available: https://www.mdpi.com/2313-433X/7/3/46

[23] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 4765–4774. [Online]. Available: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf